Connectivity Processor with Cache, DSP, and Bluetooth®, USB, and Dual CAN Interfaces

Check for Samples: CP3SP33

1 Introduction

1.1 Features

- CPU Features
  - Fully static RISC processor core, capable of operating from 0 to 96 MHz with zero wait/hold state
  - Minimum 10.4 ns instruction cycle time with a 96-MHz internal clock frequency, based on a 12-MHz external input
  - 4K-byte, 4-way set-associative instruction cache
  - 69 independently vectored peripheral interrupts
- DSP Features
  - Capable of operating up to 96 MHz
  - 16-bit fixed-point arithmetic, dual-MAC architecture
  - 32-bit interface to 4K-byte RAM shared with CPU
  - 32-bit external bus interface
  - Bus master interface to audio peripherals and I/O
- Memory
  - 4K bytes CPU instruction cache
  - 32K bytes CPU data RAM
  - 4K bytes CPU/DSP shared RAM
  - 24K bytes DSP program RAM
  - 24K bytes DSP data RAM
  - 8K bytes Bluetooth® sequencer and data RAM
  - Addresses up to 96M bytes (FBGA-224 package) or 8M bytes (FBGA-144 package) of external memory
- Broad Range of Hardware Communications Peripherals
  - Bluetooth Lower Link Controller (LLC) including a shared 7K byte Bluetooth data RAM and 1K byte Bluetooth Sequencer RAM
  - Universal Serial Bus (USB) 2.0 On-The-Go
  - Audio/telematics codec with dual ADC inputs and high quality stereo DAC output
  - Two CAN interfaces with 15 message buffers conforming to CAN specification 2.0B active
  - Two ACCESS.bus serial bus interfaces (I^2C compatible)
  - Two 8/16-bit SPI, Microwire/Plus serial interfaces
  - I^2S digital audio bus interface
  - Four Universal Asynchronous Receiver/Transmitter (UART) channels, one channel has USART capability
  - Advanced Audio Interface (AAI) to connect to external 8/ 13-bit PCM Codecs as well as to ISDN-Controllers through the IOM-2 interface (slave only)
  - Two CVSD/PCM converters, for supporting two bidirectional audio connectionsM
- External Bus Interface Shared Between CPU and DSP
  - 16/32-bit data busbus interface
  - 23-bit address bus
  - 3 programmable chip select outputs
  - Up to 96M bytes external memory
  - 8-level write buffer
- General-Purpose Hardware Peripherals
  - 10-channel, 10-bit A/D Converter (ADC)
  - 16-channel DMA controller
  - Dual 16-bit Multi-Function Timer (MFT)
  - Dual Versatile Timer Units (VTU), each with four independent timers
  - Timing and Watchdog Unit
- Extensive Power and Clock Management Support
  - Two Phase Locked Loops (PLL) for synthesizing independent system and audio peripheral clocks
  - Two independent oscillators for Active mode (12 MHz) and Power Save mode (32.768 kHz) clocks
  - Low-power modes (Power Save, Idle, and Halt) for slowing or stopping clocks to optimize power consumption while meeting application needs

- **Flexible I/O**
  - **Up to 64 general-purpose I/O pins (shared with on-chip peripheral I/O)**
  - **Programmable I/O pin characteristics: TRI-STATE output, push-pull output, weak pullup/pulldown input, high impedance input, high-speed drive capability**
  - **Schmitt triggers on general-purpose inputs**
  - **Multi-Input Wake-Up (MIWU) capability…**
- **Power Supply**
  - **I/O port operation at 3.0–3.3V**
  - **Core logic operation at 1.8V**
  - **On-chip power-on reset**
- **Temperature Range**
  - **–40°C to +85°C (Industrial)**
- **Packages**
  - **FBGA-224, FBGA-144**
- **Complete Development Environment**
  - **Pre-integrated hardware and software support for rapid prototyping and production**
  - **Multi-file C source editor, source debugger, and project manager**
  - **Comprehensive, integrated, one-stop technical support**
- **Bluetooth Protocol Stack**
  - **Applications can interface to the high-level protocols or directly to the low-level Host Controller Interface (HCI)**
  - **Transport layer support allows HCI command-based interface over UART port**
  - **Baseband (Link Controller) hardware minimizes the bandwidth demand on the CPU**
  - **Link Manager (LM)**
  - **Logical Link Control and Adaptation Protocol (L2CAP)**
  - **Service Discovery Protocol (SDP)**
  - **RFCOMM Serial Port Emulation Protocol**
  - **All packet types, piconet, and scatternet functionality**

1.2 CP3SP33 Connectivity Processor Selection Guide

NSID	Speed (MHz)	Temp. Range	On-Chip RAM	Maximum External Memory	External Data Bus Width	General Purpose I/O	Package Type
CP3SP33SMS	96	–40°C to +85°C	32K	96M	32	64	FBGA-224
CP3SP33SMR	96	–40°C to +85°C	32K	8M	16	36	FBGA-144

2 Description

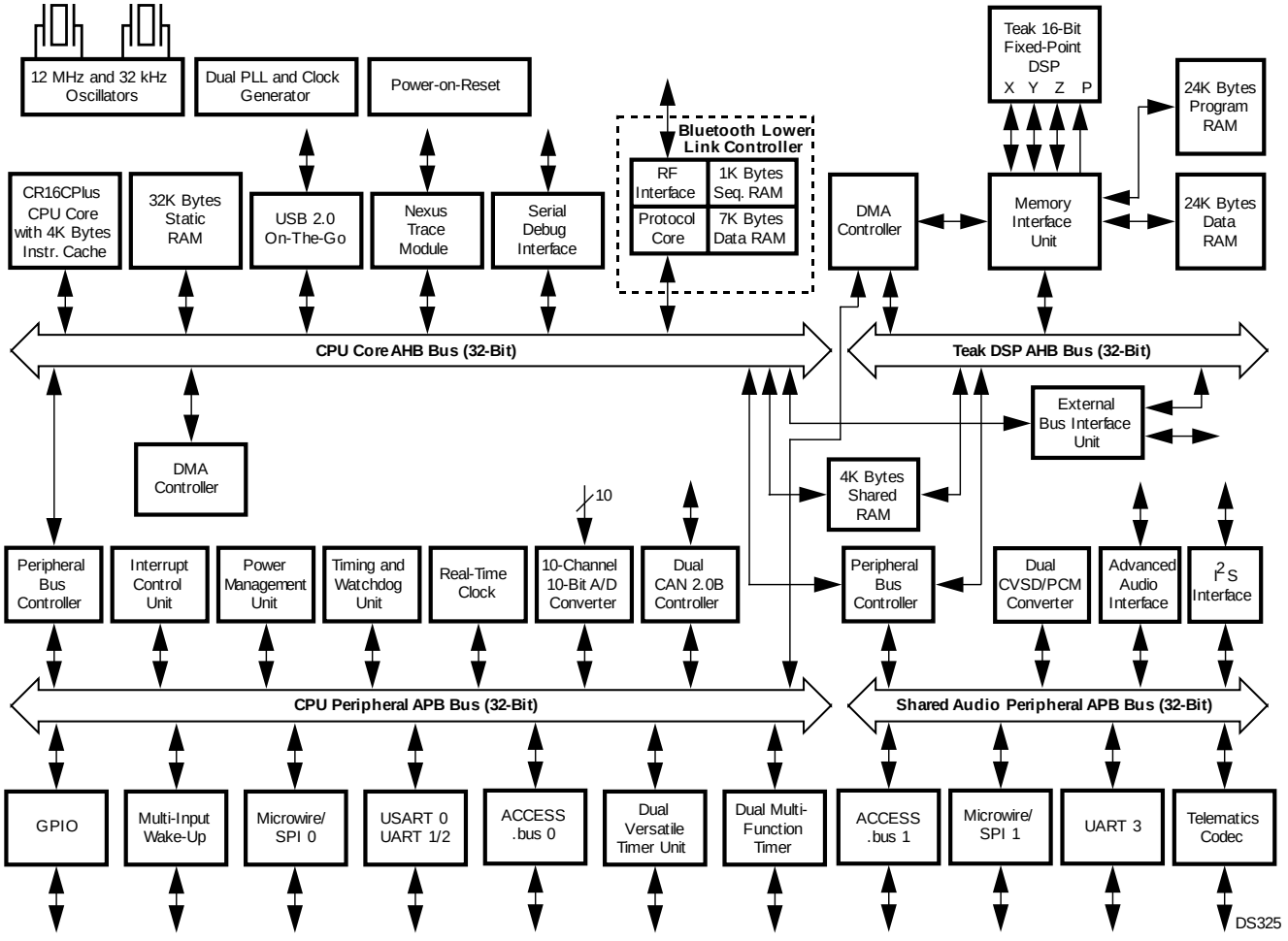
The CP3SP33 connectivity processor combines high performance with the massive integration needed for embedded Bluetooth applications. A powerful RISC core with 4Kbyte instruction cache and a Teak® DSP coprocessor provides high computing bandwidth, DMA-driven hardware communications peripherals provide high I/O bandwidth, and an external bus provides system expandability.

On-chip communications peripherals include: Bluetooth Lower Link Controller, Universal Serial Bus (2.0) OTG node and host controller, dual CAN, dual Microwire/Plus/SPI, dual ACCESS.bus, quad UART, 10-bit A/D converter, and telematics/audio codec. Additional on-chip peripherals include DMA controller, dual CVSD/PCM conversion module, I^2S and AAI digital audio bus interfaces, Timing and Watchdog Unit, dual Versatile Timer Unit, dual Multi-Function Timer, and Multi-Input Wake-Up (MIWU) unit.

In addition to providing the features needed for the next generation of embedded Bluetooth products, the CP3SP33 is backed up by the software resources that designers need for rapid time-to-market, including an operating system, Bluetooth protocol stack implementation, peripheral drivers, reference designs, and an integrated development environment. Combined with an external program memory and a Bluetooth radio transceiver such as National's LMX5252, the CP3SP33 provides a complete Bluetooth system solution.

National Semiconductor offers a complete and industry proven application development environment for CP3SP33 applications, including the IAR Embedded Workbench, iSYSTEM winIDEA and iC3000 Active Emulator, Bluetooth Development Board, Bluetooth protocol stack, and application examples.

2.1 Block Diagram



3 Device Overview

The CP3SP33 connectivity processor is an advanced microcomputer with system timing, interrupt logic, instruction cache, data memory, and I/O ports included on-chip, making it well-suited to a wide range of embedded applications. The Section 2.1 shows the major on-chip components of the CP3SP33.

3.1 CR16CPlus CPU Core

The CP3SP33 contains a CR16CPlus CPU core. This core improves upon the performance of previous CP3000 devices by adding a 4-Kbyte instruction cache and doubling the CPU core data bus bandwidth. The cache greatly reduces instruction-fetch bandwidth on the 32-bit system bus, which leaves more bus bandwidth available for DMA-based I/O. The cache moves the average execution rate closer to the peak rate of one instruction per clock cycle, especially when executing from off-chip program memory. The DMA controller provides efficient sharing of the CPU core bus between the CPU and high-bandwidth peripherals such as wired and wireless communication interfaces.

For information on the instruction set architecture, refer to the *CR16C Programmer's Reference Manual* (document number 424521772-101, which may be downloaded from Texas Instruments web site at http://www.ti.com).

3.2 Teak DSP Core

The Teak 16-bit fixed-point DSP core is designed for lowpower, high-speed digital signal processing applications, including acoustic echo cancellation, noise reduction, and MP3/WMA decoding. It features a four-bus, dual-MAC, enhanced Harvard architecture. The DSP has 24K bytes of dedicated program RAM, 24K bytes of data RAM, and a 4K-byte RAM shared with the CPU. The DSP has a bus master interface to the 4K-byte shared RAM and an external memory bus. It also has a bus master interface to a shared audio peripheral bus. The DSP is slave on the CPU peripheral bus, for downloading software to the program RAM.

The DSP has its own DMA controller for I/O and memory access.

3.3 AMBA Bus Architecture

The CPU and DSP core buses implement AMBA-compatible AHB high-performance 32-bit buses with bursting and split transactions. The CPU peripheral bus and CPU/DSP shared audio peripheral bus implement AMBA-compatible 32-bit APB buses. The CPU and DSP buses operate at independent rates up to 96 MHz. The APB buses operate at a rate which is a factor of 1, 2, or 4 slower than the CPU AHB bus.

3.4 External Bus Interface Unit

The External Bus Interface Unit (EBIU) provides programmable timing, memory type, base address, size, and bus width (8, 16, or 32 bits) for three regions of up to 32M bytes. An 8-level write buffer releases the bus master to continue execution without waiting for write cycles to complete.

3.5 Memory

The CP3SP33 devices support a uniform linear address space. Three types of on-chip memory occupy specific regions within this address space, along with any external memory:

- 32K bytes of CPU RAM
- 4K bytes of CPU/DSP shared RAM
- 8K bytes of Bluetooth sequencer and data RAM
- Up to 96M bytes of external memory

A non-volatile external program memory is used to store the application program, Bluetooth protocol stack, and real-time operating system.

The 32K bytes of CPU RAM are used for temporary storage of data and for the program stack and interrupt stack. Read and write operations can be byte-wide or word-wide, depending on the instruction executed by the CPU.

3.6 Bluetooth LLC

The integrated hardware Bluetooth Lower Link Controller (LLC) complies to the Bluetooth Specification Version 1.2 and integrates the following functions:

- 7K-byte dedicated Bluetooth data RAM
- 1K-byte dedicated Bluetooth sequencer RAM
- Support of all Bluetooth 1.2 packet types and extended Synchronous Connection-Oriented (eSCO) links
- Support for fast frequency hopping of 1600 hops/s
- Access code correlation and slot timing recovery circuit
- Power Management Control Logic
- BlueRF-compatible interface (mode 2/3) to connect with National's LMX5252 and other RF transceiver chips

3.7 USB

The full-speed Universal Serial Bus (USB) node and host controller is compatible with USB Specification 2.0 and USB On-The-Go. It integrates the required USB transceiver, the Serial Interface Engine (SIE), and USB endpoint FIFOs. A total of seven endpoint pipes are supported: one bidirectional pipe for the mandatory control EP0 and an additional six pipes for unidirectional endpoints to support USB interrupt, bulk, and isochronous data transfers.

The on-chip USB transceiver features an integrated pullup resistor on the D+ line to UVCC. This pullup resistor can be switched in or out by the USB VBUS sense input (VBUS), which eliminates the need for external components.

3.8 CAN Interface

The two CAN modules support Full CAN 2.0B class, CAN serial bus interfaces for applications that require a highspeed (up to 1 Mbits per second) or a low-speed interface with CAN bus master capability. The data transfer between CAN and the CPU is established by 15 memory-mapped message buffers, which can be individually configured as receive or transmit buffers. An incoming message is filtered by two masks, one for the first 14 message buffers and another one for the 15th message buffer to provide a basic CAN path. A priority decoder allows any buffer to have the highest or lowest transmit priority. Remote transmission requests can be processed automatically by automatic reconfiguration to a receiver after transmission or by automated transmit scheduling upon reception. In addition, a 16-bit time stamp counter supports real-time applications.

The CAN modules allow single-cycle byte or word read/ write access. A set of diagnostic features (such as loopback, listen only, and error identification) support the development with the CAN modules and provide a sophisticated error management tool.

The CAN receivers can trigger a wake-up condition out of low-power modes through the Multi-Input Wake-Up unit.

3.9 Audio/Telematics Codec

The on-chip codec is designed for voice input and stereo audio playback. It includes dual mono ADC channels operating at a sample rate of 8–24 kHz (125× oversampling clock required). A stereo DAC operates at selected sample rates from a 125× or 128× oversampling clock, driving two configurable, gain-programmable differential line driver outputs. The DAC features click and pop reduction, zero-crossing detection, tone/compensation filter, sidetone injection from the ADC, and internal power management. The ADCs accept differential or single-ended analog microphone inputs. The DAC employs fully differential signaling for high PSRR and low crosstalk. DMA transfers are supported to allow for fast CPU-independent receive and transmit.

3.10 CVSD/PCM Conversion Modules

The two CVSD/PCM modules perform conversion between CVSD data and PCM data, in which the CVSD encoding is as defined in the Bluetooth specification and the PCM data can be 8-bit µ-Law, 8-bit A-Law, or 13-bit to 16-bit Linear.

3.11 I^2S Digital Audio Bus

The Inter-IC Sound (I^2S) interface is a synchronous serial interface intended for the transfer of digital audio data. The I^2S interface can be configured as a master or a slave, and it supports all three common data formats: I^2S-mode, left-justified, and right-justified. It has programmable word length from 8 to 32 bits and programmable valid data resolution from 8 to 24 bits.

3.12 Advanced Audio Interface

The Advanced Audio Interface (AAI) provides a serial synchronous, full-duplex interface to codecs and similar serial devices. Transmit and receive paths operate asynchronously with respect to each other. Each path uses three signals for communication: shift clock, frame synchronization, and data.

When the receiver and transmitter use external shift clocks and frame sync signals, the interface operates in its asynchronous mode. Alternatively, the transmit and receive path can share the same shift clock and frame sync signals for synchronous mode operation.

3.13 Analog to Digital Converter

This device contains a 10-channel, multiplexed input, successive approximation, 10-bit Analog-to-Digital Converter. It supports both single-ended and differential modes of operation.

The integrated 10-bit ADC provides the following features:

- 10-channel, multiplexed input
- 5 differential channels
- Single-ended and differential external filtering capability
- 12-bit resolution; 10-bit accuracy
- Sign bit
- 10-microsecond conversion time
- External start trigger
- Programmable start delay after start trigger
- Poll or interrupt on conversion completion

The ADC provides several options for the voltage reference source. The positive reference can be ADVCC (internal), VREF, ADC0, or ADC1. The negative reference can be AD-VCC (internal), ADC2, or ADC3.

Two specific analog channel selection modes are supported. These are as follows:

- Allow any specific channel to be selected at one time. The A/D Converter performs the specific conversion requested and stops.

- Allow any differential channel pair to be selected at one time. The A/D Converter performs the specific differential conversion requested and stops.

In both single-ended and differential modes, there is the capability to connect the analog multiplexer output and A/D converter input to external pins. This provides the ability to externally connect a common filter/signal conditioning circuit for the A/D Converter.

3.14 Quad UART

Four UART modules support a wide range of programmable baud rates and data formats, parity generation, and several error detection schemes. The baud rate is generated on-chip, under software control. All UART modules support DMA and hardware flow control. One module has USART capability (synchronous mode). The maximum speed is 3.072 Mbaud in either synchronous or asynchronous mode.

The UARTs offer a wake-up condition from the low-power modes using the Multi-Input Wake-Up module.

3.15 Microwire/SPI

The two Microwire/SPI (MWSPI) interface modules support synchronous serial communications with other devices that conform to Microwire or Serial Peripheral Interface (SPI) specifications. It supports 8-bit and 16-bit data transfers. The maximum bus clock frequency is 12 MHz.

The Microwire interfaces allows several devices to communicate over a single system consisting of four wires: serial in, serial out, shift clock, and slave enable. At any given time, the Microwire interfaces operate as a master or a slave. The Microwire interfaces supports the full set of slave select for multi-slave implementation.

In master mode, the shift clock is generated on-chip under software control. In slave mode, a wake-up out of a low-power mode may be triggered using the Multi-Input WakeUp module.

3.16 Dual ACCESS.BUS Interface

The two ACCESS.bus (ACB) interface modules support a two-wire serial interface compatible with the ACCESS.bus physical layer. It is also compatible with Intel's System Management Bus (SMBus) and Philips' I^2C bus. The ACB modules can be configured as a bus master or slave, and they can maintain bidirectional communications with both multiple master and slave devices. The maximum bus clock frequency is 400 kHz (Fast-mode).

The ACCESS.bus receivers can trigger a wake-up condition out of the low-power modes through the Multi-Input Wake-Up module.

3.17 Dual Multi-Function Timer

The two Multi-Function Timer (MFT) modules each contain a pair of 16-bit timer/counter registers. Each timer/counter unit can be configured to operate in any of the following modes:

- *Processor-Independent Pulse Width Modulation (PWM) mode:* Generates pulses of a specified width and duty cycle and provides a general-purpose timer/ counter.
- *Dual Input Capture mode:* Measures the elapsed time between occurrences of external event and provides a general-purpose timer/counter.
- *Dual Independent Timer mode:* Generates system timing signals or counts occurrences of external events.
- *Single Input Capture and Single Timer mode:* Provides one external event counter and one system timer.

3.18 Versatile Timer Units

The two Versatile Timer Unit (VTU) modules each contain four independent timer subsystems, which operate as a dual 8-bit PWM configuration, a single 16-bit PWM timer, or a 16-bit counter with two input capture channels. Each of the timer subsystems offer an 8-bit clock prescaler to accommodate a wide range of frequencies.

3.19 Timing and Watchdog Module

The Timing and Watchdog Module (TWM) contains a Real-Time timer and a Watchdog unit. The Real-Time Clock Timing function can be used to generate periodic real-time based system interrupts. The timer output is one of 16 inputs to the Multi-Input Wake-Up module which can be used to exit from a low-power mode. The Watchdog unit is designed to detect the application program getting stuck in an infinite loop resulting in loss of program control or “runaway” programs. When the watchdog triggers, it resets the device. The TWM is clocked by the low-speed Slow Clock.

3.20 Power Management

The Power Management Module (PMM) improves the efficiency of the device by changing the operating mode and power consumption to match the required level of activity.

The device can operate in any of four power modes:

- *Active:* The device operates at full speed using the high-frequency clock. All device functions are fully operational.
- *Power Save:* The device operates at reduced speed using the Slow Clock. The CPU and some modules can continue to operate at this low speed.
- *Idle:* The device is inactive except for the Power Management Module and Timing and Watchdog Module, which continue to operate using the Slow Clock.
- *Halt:* The device is inactive but still retains its internal state (RAM and register contents).

The PMM provides a mechanism to handle Bluetooth-specific power management modes, for optimizing power consumption during special Bluetooth states, like Park, Page Scan, Inquiry Scan, etc.

3.21 Multi-Input Wake-up

The Multi-Input Wake-Up (MIWU) feature is used to return (wake-up) the device from low-power modes to the active mode. The 64-channel MIWU unit receives wake-up signals from various internal and external sources. In addition to the wake-up function, the MIWU unit can generate up to eight interrupt requests. Each MIWU channel can be individually programmed to activate one of the interrupt requests.

3.22 Input/Output Ports

The device has 64 software-configurable I/O pins (36 in the FBGA-144 package), organized into four ports called Port E, Port F, Port G, and Port H. Each pin can be configured to operate as a general-purpose input or general-purpose output. In addition, many I/O pins can be configured to operate as inputs or outputs for on-chip peripheral modules such as the UARTs or timers.

The I/O pin characteristics are fully programmable. Each pin can be configured to operate as a TRI-STATE output, push-pull output, weak pullup/pulldown input, high-speed drive, or high-impedance input.

3.23 Clock and Reset Module

The Clock and Reset module generates a 12-MHz Main Clock from an external crystal network or external clock in-put. Main Clock may be used as a reference clock for two PLL-based clock multipliers available for generating higher-speed clocks.

Most modules operate from clocks derived from Main Clock or a PLL clock. Modules on the CPU core AHB bus operate from HCLK Clock, while modules on the peripheral APB buses operate from PCLK Clock. PCLK Clock is generated by dividing HCLK Clock by 1, 2, or 4. Some peripheral modules may use one of several auxiliary clocks, which also are derived from Main Clock or a PLL clock using 12-bit programmable prescalers.

In Power-Save mode, HCLK Clock is driven by Slow Clock, which is typically a 32.768 kHz signal generated from an external clock network or a prescaled Main Clock may be used to eliminate the 32.768 kHz crystal network, for the most cost-sensitive applications. In the most power-sensitive applications, operation from an external 32.768 kHz crystal network allows the high-frequency oscillator and PLLs to be shut down.

In addition, the Clock and Reset module generates the device reset by using reset input signals coming from an external reset, the watchdog timer, or the SDI debugging interface. A power-on reset (POR) circuit eliminates the need for an external RC network. The POR circuit generates an internal reset of sufficient length if the power supply rise time specification is met.

3.24 DMA Controller

The Direct Memory Access Controller (DMAC) can speed up data transfer between memory and I/O devices or between two regions of memory, as compared to data transfers performed directly by the CPU. Cycle stealing allows the CPU and the DMAC to interleave access to the CPU core bus for greater utilization of the available bandwidth. The following on-chip modules can assert a DMA request to the DMA controller:

- USART 0 (2 request channels)
- UART 1/2/3 (6 request channels
- Advanced Audio Interface (6 request channels)
- CVSD/PCM Converter 0/1 (8 request channels)
- Microwire/SPI 0/1 (4 request channels)
- ACCESS.bus 0/1 (2 request channels)
- Codec (4 request channels)
- I2S Interface (4 request channels)

The DSP has its own DMA controller which can be configured to accept DMA requests from peripherals on the shared audio peripheral APB bus.

3.25 Serial Debug Interface

The Serial Debug Interface module (SDI module) provides a JTAG-based serial link to an external debugger, for example running on a PC. In addition, the SDI module integrates an on-chip debug module, which allows the user to set up to eight hardware breakpoints on instruction execution and data transfer. The SDI module can act as a CPU bus master to access all memory-mapped resources, such as RAM and peripherals. Therefore it also allows for fast program code download using the JTAG interface.

3.26 Development Support

In addition to providing the features needed for the next generation of embedded Bluetooth products, the CP3SP33 devices are backed up by the software resources that designers need for rapid product development, including an operating system, Bluetooth protocol stack implementation, peripheral drivers, reference designs, and an integrated development environment. Combined with National's LMX5252 Bluetooth radio transceiver, the CP3SP33 devices provide a total Bluetooth system solution.

Texas Instruments offers a complete and industry-proven application development environment for CP3SP33 applications, including the IAR Embedded Workbench, iSYSTEM winIDEA and iC3000 Active Emulator, Bluetooth Development Board, Bluetooth Protocol Stack, and Application Software. See your Texas Instruments sales representative for current information on availability and features of emulation equipment and evaluation boards.

4 Signal Descriptions

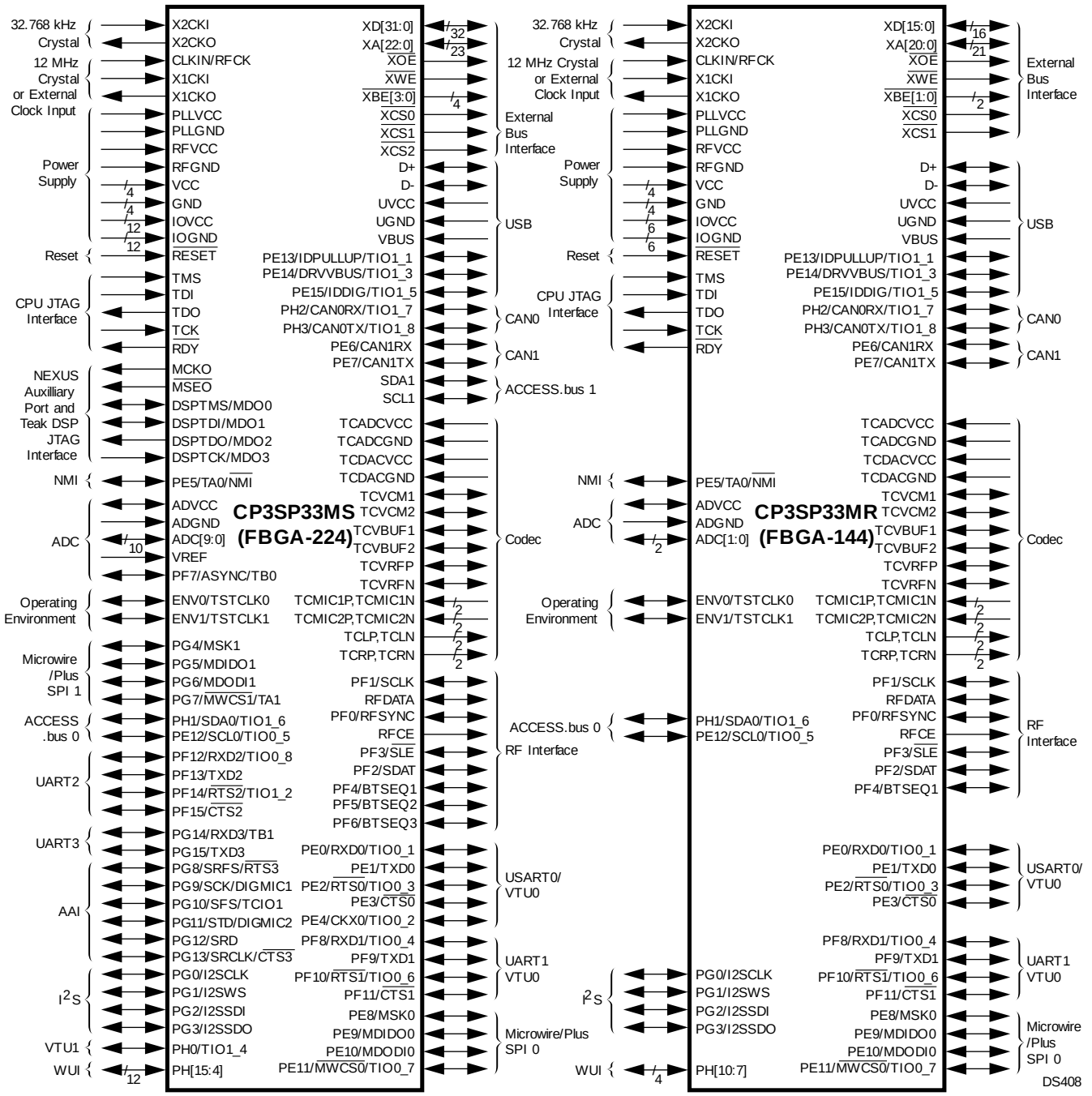


Figure 4-1. CP3SP33 Device Signals

Some pins may be enabled as general-purpose I/O-port pins or as alternate functions associated with specific peripherals or interfaces. These pins may be individually configured as port pins, even when the associated peripheral or interface is enabled.

Table 4-1 describes the device signals. When the number of signals differs between packages, the first number in the Pins column is for the FBGA-224 package and the second number is for the FBGA-144 package.

Table 4-1. CP3SP33 Signal Descriptions

NAME	PINS	I/O	PRIMARY FUNCTION	ALTERNATE NAME	I/O	ALTERNATE FUNCTION
CLKIN	1	Input	12 MHz Clock Input	RFCK	Input	RF Interface Clock
X1CKI	1	Input	12 MHz Oscillator Input	None	None	None
X1CKO	1	Output	12 MHz Oscillator Output	None	None	None
X2CKI	1	Input	32 kHz Oscillator Input	None	None	None
X2CKO	1	Output	32 kHz Oscillator Output	None	None	None
$\overline{\text{RESET}}$	1	Input	Chip general reset	None	Input	None
ENV0	1	Input	Special mode select input with internal pullup during reset	TSTCLK0	Output	Internal Clock Output
ENV1	1	Input	Special mode select input with internal pullup during reset	TSTCLK1	Output	Internal Clock Output
VCC	4	Input	Core Logic, 1.8V Power Supply	None	None	None
GND	4	Input	Core Ground	None	None	None
IOVCC	12/6	Input	I/O 3.3V Power Supply	None	None	None
IOGND	12/6	Input	I/O Ground	None	None	None
PLLVCC	1	Input	PLL 1.8V Power Supply	None	None	None
PLLGND	1	Input	PLL Ground	None	None	None
ADVCC	1	Input	ADC 1.8V Power Supply	None	None	None
ADGND	1	Input	ADC Ground	None	None	None
TCDACVCC	1	Input	Codec 3.3V Power Supply	None	None	None
TCDACGND	1	Input	Codec 3.3V Ground	None	None	None
TCADCVCC	1	Input	Codec 1.8V Power Supply	None	None	None
TCADCGND	1	Input	Codec 1.8V Ground	None	None	None
RFVCC	1	Input	RF Interface Power Supply	None	None	None
RFGND	1	Input	RF Interface Ground	None	None	None
UVCC	1	Input	USB 3.3V Transceiver Supply	None	None	None
UGND	1	Input	USB 3.3V Transceiver Ground	None	None	None
TMS	1	Input	CPU JTAG Test Mode Select (with internal weak pullup)	None	None	None
TCK	1	Input	CPU JTAG Test Clock Input (with internal weak pullup)	None	None	None
TDI	1	Input	CPU JTAG Test Data Input (with internal weak pullup)	None	None	None
TDO	1	Output	CPU JTAG Test Data Output	None	None	None
$\overline{\text{RDY}}$	1	Output	NEXUS Ready Output	None	None	None
MCKO	1/0	Output	NEXUS Trace Clock Output	None	None	None
$\overline{\text{MSEO}}$	1/0	Output	NEXUS Trace End of Message Output	None	None	None
DSPTMS	1/0	Input	DSP JTAG Test Mode Select (with internal weak pullup)	MDO0	Output	NEXUS Trace Data Output
DSPTDI	1/0	Input	DSP JTAG Test Data Input (with internal weak pullup)	MDO1	Output	NEXUS Trace Data Output
DSPTDO	1/0	Output	DSP JTAG Test Data Output	MDO2	Output	NEXUS Trace Data Output
DSPTCK	1/0	Input	DSP JTAG Test Clock Input (with internal weak pullup)	MDO3	Output	NEXUS Trace Data Output
RFDATA	1	I/O	RF Interface Data	None	None	None
RFCE	1	Output	RF Interface Chip Enable	None	None	None
VBUS	1	Input	USB VSENSE (5V tol.)	None	None	None

Table 4-1. CP3SP33 Signal Descriptions (continued)

NAME	PINS	I/O	PRIMARY FUNCTION	ALTERNATE NAME	I/O	ALTERNATE FUNCTION
D+	1	I/O	USB D+ Upstream Port	None	None	None
D–	1	I/O	USB D– Upstream Port	None	None	None
ADC0	1	Input	ADC Input Channel 0	None	None	None
ADC1	1	Input	ADC Input Channel 1	None	None	None
ADC2	1/0	Input	ADC Input Channel 2	None	None	None
ADC3	1/0	Input	ADC Input Channel 3	None	None	None
ADC4	1/0	Input	ADC Input Channel 4	MUXOUT0	Output	Analog Multiplexer Output 0
ADC5	1/0	Input	ADC Input Channel 5	MUXOUT1	Output	Analog Multiplexer Output 1
ADC6	1/0	Input	ADC Input Channel 6	None	None	None
ADC7	1/0	Input	ADC Input Channel 7	ADCIN	Input	ADC Input (in MUX mode)
ADC8	1/0	Input	ADC Input Channel 8	None	None	None
ADC9	1/0	Input	ADC Input Channel 9	None	None	None
VREF	1/0	Input	ADC Voltage Reference	None	None	None
TCVBUF1	1	Input	Codec ADC1 External Filter Capacitor Pin (1.5V)	None	None	None
TCVBUF2	1	Input	Codec ADC2 External Filter Capacitor Pin (1.5V)	None	None	None
TCVCM1	1	Input	Codec ADC1 External Filter Capacitor Pin (0.9V)	None	None	None
TCVCM2	1	Input	Codec ADC2 External Filter Capacitor Pin (0.9V)	None	None	None
TCVRFP	1	Output	Codec ADC1/ADC2 1.5V Reference (buffered version to provide microphone bias)	None	None	None
TCVRFN	1	Input	Codec ADC1/ADC2 Reference Ground (connect to 0V)	None	None	None
TCMIC1P	1	Input	Codec Microphone 1 Input (+)	None	None	None
TCMIC1N	1	Input	Codec Microphone 1 Input (–)	None	None	None
TCMIC2P	1	Input	Codec Microphone 2 Input (+)	None	None	None
TCMIC2N	1	Input	Codec Microphone 2 Input (–)	None	None	None
TCRP	1	Output	Codec Right Channel Output (+)	None	None	None
TCRN	1	Output	Codec Right Channel Output (–)	None	None	None
TCLP	1	Output	Codec Left Channel Output (+)	None	None	None
TCLN	1	Output	Codec Left Channel Output (–)	None	None	None
SCL1	1	I/O	ACCESS.bus 1 Clock	None	None	None
SDA1	1	I/O	ACCESS.bus 1 Serial Data	None	None	None
PE0	1	I/O	Generic I/O	RXD0	Input	UART 0 Receive Data Input
				TIO0_1	I/O	Versatile Timer Unit 0 Input 1
PE1	1	I/O	Generic I/O	TXD0	Output	UART 0 Transmit Data Output
PE2	1	I/O	Generic I/O	$\overline{\text{RTS0}}$	Output	UART 0 Ready-To-Send Output
				TIO0_3	I/O	Versatile Timer Unit 0 Input 3
PE3	1	I/O	Generic I/O	$\overline{\text{CTS0}}$	Input	UART 0 Clear-To-Send Input
PE4	1/0	I/O	Generic I/O	CKX0	I/O	UART 0 Clock Input
				TIO0_2	I/O	Versatile Timer Unit 0 Input 2
PE5	1	I/O	Generic I/O	TA0	I/O	Multi-Function Timer 0 Port A
				$\overline{\text{NMI}}$	Input	Non-Maskable Interrupt
PE6	1	I/O	Generic I/O	CAN1RX	Input	CAN 1 Receive Input
PE7	1	I/O	Generic I/O	CAN1TX	Output	CAN 1 Transmit Output
PE8	1	I/O	Generic I/O	MSK0	I/O	Microwire/SPI 0 Shift Clock

Table 4-1. CP3SP33 Signal Descriptions (continued)

NAME	PINS	I/O	PRIMARY FUNCTION	ALTERNATE NAME	I/O	ALTERNATE FUNCTION
PE9	1	I/O	Generic I/O	MDIDO0	I/O	Microwire/SPI 0 Master In Slave Out
PE10	1	I/O	Generic I/O	MDODI0	I/O	Microwire/SPI 0 Master Out Slave In
PE11	1	I/O	Generic I/O	$\overline{\text{MWCS0}}$	I/O	Microwire/SPI 0 Slave Select Input
				TIO0_7	I/O	Versatile Timer Unit 0 Input 7
PE12	1	I/O	Generic I/O	SCL0	I/O	ACCESS.bus 0 Clock
				TIO0_5	I/O	Versatile Timer Unit 0 Input 5
PE13	1	I/O	Generic I/O	IDPULLUP	Output	USB OTG ID Pullup Enable
				TIO1_1	I/O	Versatile Timer Unit 1 Input 6
PE14	1	I/O	Generic I/O	DRVVBUS	Output	USB OTG VBUS Driver Enable
				TIO1_3	I/O	Versatile Timer Unit 1 Input 3
PE15	1	I/O	Generic I/O	IDDIG	Input	USB OTG ID Input
				TIO1_5	I/O	Versatile Timer Unit 1 Input 5
PF0	1	I/O	Generic I/O	RFSYNC	Output	RF Interface Frequency Correlation/DC Compensation Output
PF1	1	I/O	Generic I/O	SCLK	Output	RF Interface Shift Clock Output
PF2	1	I/O	Generic I/O	SDAT	I/O	RF Interface Data
PF3	1	I/O	Generic I/O	$\overline{\text{SLE}}$	Output	RF Interface Load Enable
PF4	1	I/O	Generic I/O	BTSEQ1	Output	Bluetooth Sequencer Status
PF5	1/0	I/O	Generic I/O	BTSEQ2	Output	Bluetooth Sequencer Status
PF6	1/0	I/O	Generic I/O	BTSEQ3	Output	Bluetooth Sequencer Status
PF7	1/0	I/O	Generic I/O	ASYNC	Input	ADC External Start Trigger
				TB0	Input	Multi-Function Timer 0 Port B
PF8	1	I/O	Generic I/O	RXD1	Input	UART 1 Receive Data Input
				TIO0_4	I/O	Versatile Timer Unit 0 Input 4
PF9	1	I/O	Generic I/O	TXD1	Output	UART 1 Transmit Data Output
PF10	1	I/O	Generic I/O	$\overline{\text{RTS1}}$	Output	UART 1 Ready-To-Send Output
				TIO0_6	I/O	Versatile Timer Unit 6 Input 6
PF11	1	I/O	Generic I/O	$\overline{\text{CTS1}}$	Input	UART 1 Clear-To-Send Input
PF12	1/0	I/O	Generic I/O	RXD2	Input	UART 2 Receive Data Input
				TIO0_8	I/O	Versatile Timer Unit 0 Input 8
PF13	1/0	I/O	Generic I/O	TXD2	Output	UART 2 Transmit Data Output
PF14	1/0	I/O	Generic I/O	$\overline{\text{RTS2}}$	Output	UART 2 Ready-To-Send Output
				TIO1_2	I/O	Versatile Timer Unit 1 Input 2
PF15	1/0	I/O	Generic I/O	$\overline{\text{CTS2}}$	Input	UART 2 Clear-To-Send Input
PG0	1	I/O	Generic I/O	I2SCLK	I/O	I^2S Serial Clock
PG1	1	I/O	Generic I/O	I2SWS	I/O	I^2S Word Select
PG2	1	I/O	Generic I/O	I2SSDI	Input	I^2S Serial Data Input
PG3	1	I/O	Generic I/O	I2SSDO	Output	I^2S Serial Data Output
PG4	1/0	I/O	Generic I/O	MSK1	I/O	Microwire/SPI 1 Shift Clock
PG5	1/0	I/O	Generic I/O	MDIDO1	I/O	Microwire/SPI 1 Master In Slave Out
PG6	1/0	I/O	Generic I/O	MDODI1	I/O	Microwire/SPI 1 Master Out Slave In

Table 4-1. CP3SP33 Signal Descriptions (continued)

NAME	PINS	I/O	PRIMARY FUNCTION	ALTERNATE NAME	I/O	ALTERNATE FUNCTION
PG7	1/0	I/O	Generic I/O	$\overline{\text{MWCS1}}$	I/O	Microwire/SPI 1 Slave Select Input
				TA1	I/O	Multi-Function Timer 1 Port A
PG8	1/0	I/O	Generic I/O	SRFS	I/O	AAI Serial Receive Frame Sync
				$\overline{\text{RTS3}}$	Output	UART 3 Ready-To-Send Output
PG9	1/0	I/O	Generic I/O	SCK	I/O	AAI Serial Transmit Clock
				DIGMIC1	Input	Digital Microphone Input 1
PG10	1/0	I/O	Generic I/O	SFS	I/O	AAI Serial Transmit Frame Sync
				TCIO1	I/O	Codec GPIO
PG11	1/0	I/O	Generic I/O	STD	Output	AAI Serial Transmit Data
				DIGMIC2	Input	Digital Microphone Input 2
PG12	1/0	I/O	Generic I/O	SRD	Input	AAI Serial Receive Data
PG13	1/0	I/O	Generic I/O	SRCLK	I/O	AAI Serial Receive Clock
				$\overline{\text{CTS3}}$	Input	UART 3 Clear-To-Send Input
PG14	1/0	I/O	Generic I/O	RXD3	Input	UART 3 Receive Data Input
				TB1	Input	Multi-Function Timer 1 Port B
PG15	1/0	I/O	Generic I/O	TXD3	Output	UART 3 Transmit Data Output
PH0	1/0	I/O	Generic I/O	TIO1_4	I/O	Versatile Timer Unit 1 Input 4
PH1	1	I/O	Generic I/O	SDA0	I/O	ACCESS.bus 0 Serial Data
				TIO1_6	I/O	Versatile Timer Unit 1 Input 6
PH2	1	I/O	Generic I/O	CAN0RX	Input	CAN 0 Receive Input
				TIO1_7	I/O	Versatile Timer Unit 1 Input 7
PH3	1	I/O	Generic I/O	CAN0TX	Output	CAN 0 Transmit Output
				TIO1_8	I/O	Versatile Timer Unit 1 Input 8
PH4	1/0	I/O	Generic I/O	None	None	None
PH5	1/0	I/O	Generic I/O	None	None	None
PH6	1/0	I/O	Generic I/O	None	None	None
PH7	1	I/O	Generic I/O	None	None	None
PH8	1	I/O	Generic I/O	None	None	None
PH9	1	I/O	Generic I/O	None	None	None
PH10	1	I/O	Generic I/O	None	None	None
PH11	1/0	I/O	Generic I/O	None	None	None
PH12	1/0	I/O	Generic I/O	None	None	None
PH13	1/0	I/O	Generic I/O	None	None	None
PH14	1/0	I/O	Generic I/O	None	None	None
PH15	1/0	I/O	Generic I/O	None	None	None
XWE	1	Output	External Bus Write Enable	None	Output	None
XOE	1	Output	External Bus Output Enable	None	Output	None
XBE[3:0]	4/2	Output	External Bus Byte Enables	None	Output	None
XCS[2:0]	3/2	Output	External Bus Chip Selects	None	Output	None
XA[22:0]	23/21	Output	External Address Bus	None	Output	None
XD[31:0]	32/16	I/O	External Data Bus	None	I/O	None

5 CPU Architecture

The CP3SP33 uses the CR16CPlus third-generation 16-bit CompactRISC processor core. The CPU implements a Reduced Instruction Set Computer (RISC) architecture that allows an effective execution rate of up to one instruction per clock cycle. For a detailed description of the CR16CPlus architecture, see the *CompactRISC CR16C Programmer's Reference Manual* which is available on the Texas Instruments web site (http://www.ti.com).

The CR16CPlus CPU core includes these internal registers:

- General-purpose registers (R0-R13, RA, and SP)
- Dedicated address registers (PC, ISP, USP, and INTBASE)
- Processor Status Register (PSR)
- Configuration Register (CFG)

The R0-R11, PSR, and CFG registers are 16 bits wide. The R12, R13, RA, SP, ISP and USP registers are 32 bits wide. The PC register is 24 bits wide. Figure 5-1 shows the CPU registers.

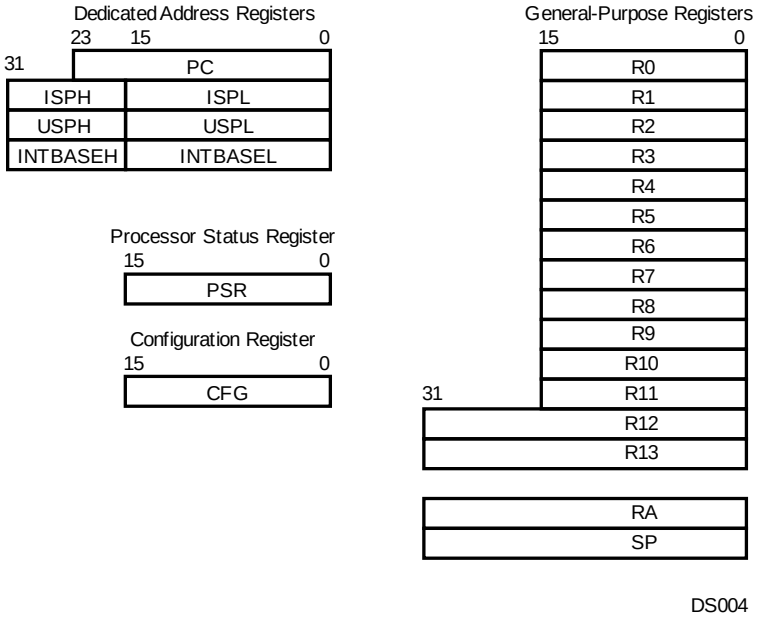


Figure 5-1. CPU Registers

Some register bits are designated as "reserved." Software must write a zero to these bit locations when it writes to the register. Read operations from reserved bit locations return undefined values.

5.1 General-Purpose Registers

The CompactRISC CPU features 16 general-purpose registers. These registers are used individually as 16-bit operands or as register pairs for operations on addresses greater than 16 bits.

- General-purpose registers are defined as R0 through R13, RA, and SP.
- Registers are grouped into pairs based on the setting of the Short Register bit in the Configuration Register (CFG.SR). When the CFG.SR bit is set, the grouping of register pairs is upward-compatible with the architecture of the earlier CR16A/B CPU cores: (R1,R0), (R2,R1) ... (R11,R10), (R12_L, R11), (R13_L, R12_L), (R14_L, R13_L) and SP. (R14_L, R13_L) is the same as (RA,ERA).
- When the CFG.SR bit is clear, register pairs are grouped in the manner used by native CR16CPlus software: (R1,R0), (R2,R1) ... (R11,R10), (R12_L, R11), R12, R13, RA, SP. R12, R13, RA, and SP are 32-bit registers for holding addresses greater than 16 bits.

With the recommended calling convention for the architecture, some of these registers are assigned special hardware and software functions. Registers R0 to R13 are for general-purpose use, such as holding variables, addresses, or index values. The SP register holds a pointer to the program run-time stack. The RA register holds a subroutine return address. The R12 and R13 registers are available to hold base addresses used in the index addressing mode.

If a general-purpose register is specified by an operation that is 8 bits long, only the lower byte of the register is used; the upper part is not referenced or modified. Similarly, for word operations on register pairs, only the lower word is used. The upper word is not referenced or modified.

5.2 Dedicated Address Registers

The CR16CPlus has four dedicated address registers to implement specific functions: the PC, ISP, USP, and INTBASE registers.

5.2.1 Program Counter (PC) Register

The 24-bit value in the PC register points to the first byte of the instruction currently being executed. CR16CPlus instructions are aligned to even addresses, therefore the least significant bit of the PC is always 0. At reset, the PC is initialized to 0 or an optional predetermined value. When a warm reset occurs, value of the PC prior to reset is saved in the (R1,R0) general-purpose register pair.

5.2.2 Interrupt Stack Pointer (ISP)

The 32-bit ISP register points to the top of the interrupt stack. This stack is used by hardware to service exceptions (interrupts and traps). The stack pointer may be accessed as the ISP register for initialization. The interrupt stack can be located anywhere in the CPU address space. The ISP cannot be used for any purpose other than the interrupt stack, which is used for automatic storage of the CPU registers when an exception occurs and restoration of these registers when the exception handler returns. The interrupt stack grows downward in memory. The least significant bit and the 8 most significant bits of the ISP register are always 0.

5.2.3 User Stack Pointer (USP)

The USP register points to the top of the user-mode program stack. Separate stacks are available for user and supervisor modes, to support protection mechanisms for multitasking software. The processor mode is controlled by the U bit in the PSR register (which is called PSR.U in the shorthand convention). Stack grow downward in memory. If the USP register points to an illegal address (any address greater than 0x00FF_FFFF) and the USP is used for stack access, an IAD trap is taken.

5.2.4 Interrupt Base Register (INTBASE)

The INTBASE register holds the address of the dispatch table for exceptions. The dispatch table can be located any where in the CPU address space. When loading the INTBASE register, bits 31 to 24 and bit 0 must written with 0.

5.3 Processor Status Register (PSR)

15 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	I	P	E	0	N	Z	F	0	U	L	T	C

C The Carry bit indicates whether a carry or borrow occurred after addition or subtraction.
0 – No carry or borrow occurred.
1 – Carry or borrow occurred.

T The Trace bit enables execution tracing, in which a Trace trap (TRC) is taken after every instruction. Tracing is automatically disabled during the execution of an exception handler.
0 – Tracing disabled.
1 – Tracing enabled.

L The Low bit indicates the result of the last comparison operation, with the operands interpreted as unsigned integers.
0 – Second operand greater than or equal to first operand.
1 – Second operand less than first operand.

U The User Mode bit controls whether the CPU is in user or supervisor mode. In supervisor mode, the SP register is used for stack operations. In user mode, the USP register is used instead. User mode is entered by executing the Jump USR instruction. When an exception is taken, the exception handler automatically begins execution in supervisor mode. The USP register is accessible using the Load Processor Register (LPR/LPRD) instruction in supervisor mode. In user mode, an attempt to access the USP register generates a UND trap. 0 – CPU is executing in supervisor mode. 1 – CPU is executing in user mode.

F The Flag bit is a general condition flag for signaling exception conditions or distinguishing the results of an instruction, among other thing uses. For example, integer arithmetic instructions use the F bit to indicate an overflow condition after an addition or subtraction operation

Z The Zero bit is used by comparison operations. In a comparison of integers, the Z bit is set if the two operands are equal. If the operands are unequal, the Z bit is cleared.
0 – Source and destination operands unequal.
1 – Source and destination operands equal.

N The Negative bit indicates the result of the last comparison operation, with the operands interpreted as signed integers.
0 – Second operand greater than or equal to first operand.
1 – Second operand less than first operand.

E The Local Maskable Interrupt Enable bit enables or disables maskable interrupts. If this bit and the Global Maskable Interrupt Enable (I) bit are both set, all interrupts are enabled. If either of these bits is clear, only the nonmaskable interrupt is enabled. The E bit is set by the Enable Interrupts (EI) instruction and cleared by the Disable Interrupts (DI) instruction.
0 – Maskable interrupts disabled.
1 – Maskable interrupts enabled.

P The Trace Trap Pending bit is used together with the Trace (T) bit to prevent a Trace (TRC) trap from occurring more than once for one instruction. At the beginning of the execution of an instruction, the state of the T bit is copied into the P bit. If the P bit remains set at the end of the instruction execution, the TRC trap is taken.
0 – No trace trap pending.
1 – Trace trap pending.

I The Global Maskable Interrupt Enable bit is used to enable or disable maskable interrupts. If this bit and the Local Maskable Interrupt Enable (E) bit are both set, all maskable interrupts are taken. If either bit is clear, only the non-maskable interrupt is taken. Unlike the E bit, the I bit is automatically cleared when an interrupt occurs and automatically set upon completion of an interrupt handler.
0 – Maskable interrupts disabled.
1 – Maskable interrupts enabled.

Bits Z, C, L, N, and F of the PSR are referenced from assembly language by the condition code in conditional branch instructions. A conditional branch instruction may cause a branch in program execution, based on the value of one or more of these PSR bits. For example, one of the Bcond instructions, BEQ (Branch EQual), causes a branch if the PSR.Z bit is set.

On reset, bits 0 through 11 of the PSR are cleared, except for the PSR.E bit, which is set. On warm reset, the values of each bit before reset are copied into the R2 general-purpose register. Bits 4 and 8 of the PSR have a constant value of 0. Bits 12 through 15 are reserved. In general, status bits are modified only by specific instructions. Otherwise, status bits maintain their values throughout instructions which do not implicitly affect them.

5.4 Configuration Register (CFG)

The CFG register is used to enable or disable various operating modes and to control optional on-chip caches. All CFG bits are cleared on reset.

15 10	9	8	7	6	5	4	3 2	1	0
Reserved	SR	ED	0	0	LIC	IC	Reserved	0	0

IC The Instruction Cache bit controls whether the 4K-byte instruction cache is enabled for satifying instruction fetches by the CPU. When the instruction cache is disabled, every instruction fetch is propagated to the CPU core bus. Disabling the cache automatically invalidates all of the cache entries. When the cache is enabled, instruction fetches that are cache hits do not result in bus cycles on the CPU core bus.
0 – Instruction cache disabled.
1 – Instruction cache enabled.

LIC The Lock Instruction Cache bit controls whether the 4K-byte instruction cache is locked. When the instruction cache is locked, no new entries are allocated as a result of cache misses, however cache hits continue to be handled by the cache.
0 – Instruction cache is not locked.
1 – Instruction cache is locked.

ED The Extended Dispatch bit selects whether the size of an entry in the interrupt dispatch table (IDT) is 16 or 32 bits. Each entry holds the address of the appropriate exception handler. When the IDT has 16-bit entries, and all exception handlers must reside in the first 128K of the address space. The location of the IDT is held in the INTBASE register, which is not affected by the state of the ED bit.
0 – Interrupt dispatch table has 16-bit entries.
1 – Interrupt dispatch table has 32-bit entries.

SR The Short Register bit enables a compatibility mode for the CR16B large model. In the CR16CPlus core, registers R12, R13, and RA are extended to 32 bits. In the CR16B large model, only the lower 16 bits of these registers are used, and these “short registers” are paired together for 32-bit operations. In this mode, the (RA, R13) register pair is used as the extended RA register, and address displacements relative to a single register are supported with offsets of 0 and 14 bits in place of the index addressing with these displacements.
0 – 32-bit registers are used.
1 – 16-bit registers are used (CR16B mode).

5.5 Addressing Modes

The CR16CPlus CPU core implements a load/store architecture, in which arithmetic and logical instructions operate on register operands. Memory operands are made accessible in registers using load and store instructions. For effIcient implementation of I/O-intensive embedded applications, the architecture also provides a set of bit operations that operate on memory operands.

The load and store instructions support these addressing modes: register/pair, immediate, relative, absolute, and index addressing. When register pairs are used, the lower bits are in the lower index register and the upper bits are in the higher index register. When the CFG.SR bit is clear, the 32-bit registers R12, R13, RA, and SP are also treated as register pairs.

References to register pairs in assembly language use parentheses. With a register pair, the lower numbered register pair must be on the right. For example,

```
jump (r5, r4)
load $4(r4,r3), (r6,r5)
load $5(r12), (r13)
```

The instruction set supports the following addressing modes:

Register/Pair Mode In register/pair mode, the operand is held in a general-purpose register, or in a general-purpose register pair. For example, the following instruction adds the contents of the low byte of register r1 to the contents of the low byte of r2, and places the result in the low byte register r2. The high byte of register r2 is not modified.
ADDB R1, R2

Immediate Mode In immediate mode, the operand is a constant value which is encoded in the instruction. For example, the following instruction multiplies the value of r4 by 4 and places the result in r4.
MULW $4, R4

Relative Mode In relative mode, the operand is addressed using a relative value (displacement) encoded in the instruction. This displacement is relative to the current Program Counter (PC), a general-purpose register, or a register pair.

In branch instructions, the displacement is always relative to the current value of the PC Register. For example, the following instruction causes an unconditional branch to an address 10 ahead of the current PC.
BR *+10

In another example, the operand resides in memory. Its address is obtained by adding a displacement encoded in the instruction to the contents of register r5. The address calculation does not modify the contents of register r5.
LOADW 12(R5), R6

The following example calculates the address of a source operand by adding a displacement of 4 to the contents of a register pair (r5, r4) and loads this operand into the register pair (r7, r6). r7 receives the high word of the operand, and r6 receives the low word.
LOADD 4(r5, r4), (r7, r6)

Index Mode In index mode, the operand address is calculated with a base address held in either R12 or R13. The CFG.SR bit must be clear to use this mode.

- For relative mode operands, the memory address is calculated by adding the value of a register pair and a displacement to the base address. The displacement can be a 14 or 20-bit unsigned value, which is encoded in the instruction.
- For absolute mode operands, the memory address is calculated by adding a 20-bit absolute address encoded in the instruction to the base address.

In the following example, the operand address is the sum of the displacement 4, the contents of the register pair (r5,r4), and the base address held in register r12. The word at this address is loaded into register r6.
LOADW [r12]4(r5, r4), r6

Absolute Mode In absolute mode, the operand is located in memory, and its address is encoded in the instruction (normally 20 or 24 bits).

For example, the following instruction loads the byte at address 4000 into the lower 8 bits of register r6.
LOADB 4000, r6

For additional information on the addressing modes, see the *CompactRISC CR16C Programmer's Reference Manual*.

5.6 Stacks

A stack is a last-in, first-out data structure for dynamic storage of data and addresses. A stack consists of a block of memory used to hold the data and a pointer to the top of the stack. As more data is pushed onto a stack, the stack grows downward in memory. The CR16CPlus supports two types of stacks: the interrupt stack and program stacks.

5.6.1 Interrupt Stack

The processor uses the interrupt stack to save and restore the program state during the exception handling. Hardware automatically pushes this data onto the interrupt stack before entering an exception handler. When the exception handler returns, hardware restores the processor state with data popped from the interrupt stack. The interrupt stack pointer is held in the ISP register.

5.6.2 Program Stack

The program stack is normally used by software to save and restore register values on subroutine entry and exit, hold local and temporary variables, and hold parameters passed between the calling routine and the subroutine. The only hardware mechanisms which operate on the program stack are the PUSH, POP, and POPRET instructions.

5.6.3 User and Supervisor Stack Pointers

To support multitasking operating systems, support is provided for two program stack pointers: a user stack pointer and a supervisor stack pointer. When the PSR.U bit is clear, the SP register is used for all program stack operations. This is the default mode when the user/supervisor protection mechanism is not used, and it is the supervisor mode when protection is used.

When the PSR.U bit is set, the processor is in user mode, and the USP register is used as the program stack pointer. User mode can only be entered using the JUSR instruction, which performs a jump and sets the PSR.U bit. User mode is exited when an exception is taken and re-entered when the exception handler returns. In user mode, the LPRD instruction cannot be used to change the state of processor registers (such as the PSR).

5.7 Instruction Set

Table 5-1 lists the operand specifiers for the instruction set, and Table 3 is a summary of all instructions. For each instruction, the table shows the mnemonic and a brief description of the operation performed.

In the mnemonic column, the lower-case letter “i” is used to indicate the type of integer that the instruction operates on, either “B” for byte or “W” for word. For example, the notation ADDi for the “add” instruction means that there are two forms of this instruction, ADDB and ADDW, which operate on bytes and words, respectively.

Similarly, the lower-case string “cond” is used to indicate the type of condition tested by the instruction. For example, the notation Jcond represents a class of conditional jump instructions: JEQ for Jump on Equal, JNE for Jump on Not Equal, etc. For detailed information on all instructions, see the *CompactRISC CR16C Programmer's Reference Manual*.

Table 5-1. Key to Operand Specifiers

OPERAND SPECIFIER	DESCRIPTION
abs	Absolute address
disp	Displacement (numeric suffix indicates number of bits)
imm	Immediate operand (numeric suffix indicates number of bits)
Iposition	Bit position in memory
Rbase	Base register (relative mode)
Rdest	Destination register

Table 5-1. Key to Operand Specifiers (continued)

OPERAND SPECIFIER	DESCRIPTION
Rindex	Index register
RPbase, RPbasex	Base register pair (relative mode)
RPdest	Destination register pair
RPlink	Link register pair
Rposition	Bit position in register
Rproc	16-bit processor register
Rprocd	32-bit processor register
RPsrc	Source register pair
RPtarget	Target register pair
Rsrc, Rsrc1, Rsrc2	Source register

Table 5-2. Instruction Set Summary

MNEMONIC	OPERANDS	DESCRIPTION
MOVi	Rsrc/imm, Rdest	Move
MOVXB	Rsrc, Rdest	Move with sign extension
MOVZB	Rsrc, Rdest	Move with zero extension
MOVXW	Rsrc, RPdest	Move with sign extension
MOVZW	Rsrc, RPdest	Move with zero extension
MOVD	imm, RPdest	Move immediate to register-pair
	RPsrc, RPdest	Move between register-pairs
ADD[U]i	Rsrc/imm, Rdest	Add
ADDCi	Rsrc/imm, Rdest	Add with carry
ADDD	RPsrc/imm, RPdest	Add with RP or immediate.
MACQWa	Rsrc1, Rsrc2, RPdest	Multiply signed Q15: RPdest := RPdest + (Rsrc1 × Rsrc2)
MACSWa	Rsrc1, Rsrc2, RPdest	Multiply signed and add result: RPdest := RPdest + (Rsrc1 × Rsrc2)
MACUWa	Rsrc1, Rsrc2, RPdest	Multiply unsigned and add result: RPdest := RPdest + (Rsrc1 × Rsrc2)
MULi	Rsrc/imm, Rdest	Multiply: Rdest(8) := Rdest(8) × Rsrc(8)/imm Rdest(16) := Rdest(16) × Rsrc(16)/imm
MULSB	Rsrc, Rdest	Multiply: Rdest(16) := Rdest(8) × Rsrc(8)
MULSW	Rsrc, RPdest	Multiply: RPdest := RPdest(16) × Rsrc(16)
MULUW	Rsrc, RPdest	Multiply: RPdest := RPdest(16) × Rsrc(16);
SUBi	Rsrc/imm, Rdest	Subtract: (Rdest := Rdest Rsrc/imm)
SUBD	RPsrc/imm, RPdest	Subtract: (RPdest := RPdest RPsrc/imm)
SUBCi	Rsrc/imm, Rdest	Subtract with carry: (Rdest := Rdest Rsrc/imm)
CMPi	Rsrc/imm, Rdest	Compare Rdest Rsrc/imm
CMPD	RPsrc/imm, RPdest	Compare RPdest RPsrc/imm
BEQ0i	Rsrc, disp	Compare Rsrc to 0 and branch if EQUAL
BNE0i	Rsrc, disp	Compare Rsrc to 0 and branch if NOT EQUAL
ANDi	Rsrc/imm, Rdest	Logical AND: Rdest := Rdest & Rsrc/imm
ANDD	RPsrc/imm, RPdest	Logical AND: RPdest := RPsrc & RPsrc/imm
ORi	Rsrc/imm, Rdest	Logical OR: Rdest := Rdest | Rsrc/imm
ORD	RPsrc/imm, RPdest	Logical OR: Rdest := RPdest | RPsrc/imm
Scond	Rdest	Save condition code as boolean
XORi	Rsrc/imm, Rdest	Logical exclusive OR: Rdest := Rdest ^ Rsrc/imm
XORD	RPsrc/imm, RPdest	Logical exclusive OR: Rdest := RPdest ^ RPsrc/imm
ASHUi	Rsrc/imm, Rdest	Arithmetic left/right shift
ASHUD	Rsrc/imm, RPdest	Arithmetic left/right shift

Table 5-2. Instruction Set Summary (continued)

MNEMONIC	OPERANDS	DESCRIPTION
LSHi	Rsrc/imm, Rdest	Logical left/right shift
LSHD	Rsrc/imm, RPdest	Logical left/right shift
SBITi	Iposition, disp(Rbase)	Set a bit in memory (Because this instruction treats the destination as a readmodify-write operand, it not be used to set bits in write-only registers.)
	Iposition, disp(RPbase)	
	Iposition, (Rindex)disp(RPbasex)	
	Iposition, abs	
	Iposition, (Rindex)abs	
CBITi	Iposition, disp(Rbase)	Clear a bit in memory
	Iposition, disp(RPbase)	
	Iposition, (Rindex)disp(RPbasex)	
	Iposition, abs	
	Iposition, (Rindex)abs	
TBIT TBITi	Rposition/imm, Rsrc	Test a bit in a register Test a bit in memory
	Iposition, disp(Rbase)	
	Iposition, disp(RPbase)	
	Iposition, (Rindex)disp(RPbasex)	
	Iposition, abs	
	Iposition, (Rindex)abs	
LPR	Rsrc, Rproc	Load processor register
LPRD	RPsrc, Rprocd	Load double processor register
SPR	Rproc, Rdest	Store processor register
SPRD	Rprocd, RPdest	Store 32-bit processor register
Bcond	disp9	Conditional branch
	disp17	
	disp24	
BAL	RPlink, disp24	Branch and link
BR	disp9	Branch
	disp17	
	disp24	
EXCP	vector	Trap (vector)
Jcond	RPtarget	Conditional Jump to a large address
JAL	RA, RPtarget,	Jump and link to a large address
	RPlink, RPtarget	
JUMP	RPtarget	Jump
JUSR	RPtarget	Jump and set PSR.U
RETX		Return from exception
PUSH	imm, Rsrc, RA	Push “imm” number of registers on user stack, starting with Rsrc and possibly including RA
POP	imm, Rdest, RA	Restore “imm” number of registers from user stack, starting with Rdest and possibly including RA
POPRET	imm, Rdest, RA	Restore registers (similar to POP) and JUMP RA
LOADi	disp(Rbase), Rdest	Load (register relative)
	abs, Rdest	Load (absolute)
	(Rindex)abs, Rdest	Load (absolute index relative)
	(Rindex)disp(RPbasex), Rdest	Load (register relative index)
	disp(RPbase), Rdest	Load (register pair relative)

Table 5-2. Instruction Set Summary (continued)

MNEMONIC	OPERANDS	DESCRIPTION
LOADD	disp(Rbase), Rdest	Load (register relative)
	abs, Rdest	Load (absolute)
	(Rindex)abs, Rdest	Load (absolute index relative)
	(Rindex)disp(RPbasex), Rdest	Load (register pair relative index)
	disp(RPbase), Rdest	Load (register pair relative)
STORi	Rsrc, disp(Rbase)	Store (register relative)
	Rsrc, disp(RPbase)	Store (register pair relative)
	Rsrc, abs	Store (absolute)
	Rsrc, (Rindex)disp(RPbasex)	Store (register pair relative index)
	Rsrc, (Rindex)abs	Store (absolute index)
STORD	RPsrc, disp(Rbase)	Store (register relative)
	RPsrc, disp(RPbase)	Store (register pair relative)
	RPsrc, abs	Store (absolute)
	RPsrc, (Rindex)disp(RPbasex)	Store (register pair index relative)
	RPsrc, (Rindex)abs	Store (absolute index relative)
STOR IMM	imm4, disp(Rbase)	Store unsigned 4-bit immediate value extended to operand length in memory
	imm4, disp(RPbase)	
	imm4, (Rindex)disp(RPbasex)	
	imm4, abs	
	imm4, (Rindex)abs	
LOADM	imm3	Load 1 to 8 registers (R2-R5, R8-R11) from memory starting at (R0)
LOADMP	imm3	Load 1 to 8 registers (R2-R5, R8-R11) from memory starting at (R1, R0)
STORM	STORM imm3	Store 1 to 8 registers (R2-R5, R8-R11) to memory starting at (R2)
STORMP	imm3	Store 1 to 8 registers (R2-R5, R8-R11) to memory starting at (R7,R6)
DI		Disable maskable interrupts
EI		Enable maskable interrupts
EIWAIT		Enable maskable interrupts and wait for interrupt
NOP		No operation
WAIT		Wait for interrupt
CINV [i]		Invalidate instruction cache

6 Memory

The CP3SP33 supports a uniform 256M-byte linear address space. Program memory must reside in the first 16M bytes of the address space. Table 6-1 lists the types of memory and peripherals that occupy this memory space. Reserved addresses must not be read or written.

Table 6-1. CP3SP33 Memory Map

START ADDRESS	END ADDRESS	SIZE IN BYTES	DESCRIPTION
0000 0000h	0000 7FFFh	32K	System RAM
0000 8000h	0000 FFFFh	32K	Reserved
0001 0000h	0001 0FFFh	4K	CPU/DSP Shared RAM
0001 1000h	0001 1FFFh	4K	Reserved
0001 2000h	0001 207Fh	128	Bluetooth Registers
0001 2080h	0001 20FFh	128	Reserved
0001 2100h	0001 24FFh	1K	Bluetooth Sequencer RAM
0001 2500h	0001 3DBFh	6336	Bluetooth Data RAM (Shared RAM)
0001 3DC0h	0001 FFFFh	49,728	Reserved
0002 0000h	00FE FFFFh	16M 192K	Available for External Bus Devices
00FF 0000h	00FF 03FFh	1K	External Bus Interface Unit Registers
00FF 0400h	00FF 07FFh	1K	DMA Controller Registers
00FF 0800h	00FF 0FFFh	2K	USB Controller
00FF 1000h	00FF 3FFFh	12K	Reserved
00FF 4000h	00FF 43FFh	1K	I2S Digital Audio Interface
00FF 4400h	00FF 47FFh	1K	Audio Codec
00FF 4800h	00FF 4BFFh	1K	CVSD/PCM Converter 0
00FF 4C00h	00FF 4FFFh	1K	CVSD/PCM Converter 1
00FF 5000h	00FF 53FFh	1K	Advanced Audio Interface
00FF 5400h	00FF 57FFh	1K	ACCESS.bus 1 Interface
00FF 5800h	00FF 5BFFh	1K	Microwire/SPI 1 Interface
00FF 5C00h	00FF 5FFFh	1K	UART3
00FF 6000h	00FF 63FFh	1K	Multi-Function Timer 1
00FF 6400h	00FF 67FFh	1K	Port G
00FF 6800h	00FF 6BFFh	1K	Audio Subsystem Controller
00FF 6C00h	00FF 7FFFh	5K	Reserved
00FF 8000h	00FF 83FFh	1K	ACCESS.bus 0 Interface
00FF 8400h	00FF 87FFh	1K	Microwire/SPI 0 Interface
00FF 8800h	00FF 8BFFh	1K	Versatile Timer Unit 0
00FF 8C00h	00FF 8FFFh	1K	Versatile Timer Unit 1
00FF 9000h	00FF 93FFh	1K	Multi-Function Timer 0
00FF 9400h	00FF 97FFh	1K	USART0
00FF 9800h	00FF 9BFFh	1K	UART1
00FF 9C00h	00FF 9FFFh	1K	UART2
00FF A000h	00FF A3FFh	1K	Timing and Watchdog Module
00FF A400h	00FF A7FFh	1K	Power Management Module
00FF A800h	00FF ABFFh	1K	Real-Time Clock
00FF AC00h	00FF AFFFh	1K	Analog/Digital Converter
00FF B000h	00FF B7FFh	2K	Reserved
00FF B800h	00FF BBFFh	1K	CAN0 Buffers and Registers
00FF BC00h	00FF BFFFh	1K	CAN1 Buffers and Registers
00FF C000h	00FF C3FFh	1K	Multi-Input Wake-Up Unit

Table 6-1. CP3SP33 Memory Map (continued)

START ADDRESS	END ADDRESS	SIZE IN BYTES	DESCRIPTION
00FF C400h	00FF C7FFh	1K	Port E
00FF C800h	00FF CBFFh	1K	Port F
00FF CC00h	00FF CFFFh	1K	Port H
00FF D000h	00FF EBFFh	7K	Reserved
00FF EC00h	00FF EFFFh	1K	Teak Access Port
00FF F000h	00FF F3FFh	1K	Reserved
00FF F400h	00FF F7FFh	1K	System Configuration
00FF F800h	00FF FBFFh	1K	Reserved
00FF FC00h	00FF FFFFh	1K	Interrupt Control Unit (registers start at 00FF FE00h)
0100 0000h	FFFF FFFFh	4G 16M	Available for External Bus Devices (data space only)

7 Instruction Cache

The CPU instruction cache has the following features:

- 4K bytes data memory
- 4-way set-associative organization
- Critical word first, with wrapping cache entry fill
- Pseudo Least Recently Used (PLRU) allocation policy
- Cache locking support
- Cache invalidation support

The instruction cache is enabled by setting the IC bit in the CFG register. At reset, the cache is disabled by default. When caching is enabled, the instruction cache greatly accelerates program execution by satisfying most instruction fetches. It also greatly reduces bus traffic to external memory, which increases the bandwidth available for the DMA controllers and DSP.

The cache is organized as four ways of 64 entries each, as shown in Figure 7-1. Each entry holds 16 data bytes (8 instructions).

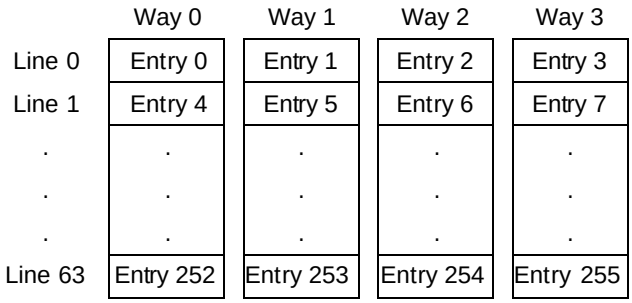


Figure 7-1. CPU Instruction Cache Organization

Each entry consists of:

- *Valid Bit*—Indicates whether the entry holds valid data. The valid bit is cleared when the cache is disabled, when the CINV [i] instruction is executed, and at reset. The valid bit is set when the entry is loaded.
- *Address Tag*—Holds address bits 31:10 associated with the entry. The address tag is loaded when a cache entry is allocated on a cache miss.
- *Data*—Eight bytes of data. The data is passed to the CPU on a cache hit, and it is loaded when a cache entry is allocated on a cache miss.

On an instruction fetch, address bits 9:4 select one of the lines. Any of the four ways may hold the target of the instruction fetch. The tags in each of the four ways are compared against bits 32:10 of the instruction address to determine which way (if any) holds the data. If one of the entries has matching address bits and its valid bit is set, the instruction fetch is a cache hit, and the cache data is passed to the CPU to satisfy the instruction fetch.

If none of the valid tags match the instruction address bits, the instruction fetch is a cache miss, and a bus cycle will be generated on the CPU core bus to read the memory location containing the target of the instruction fetch. When the memory data is received, it is loaded into the cache and passed to the CPU, which releases the CPU to continue execution. Three subsequent bus cycles read the remainder of an aligned 16-byte block to fill a cache entry.

A PLRU algorithm is used to allocate a cache entry in one of the four ways to receive the block. For each line, there are three bits B2:0 used by the algorithm. These bits are not directly visible to application software, but they may be indirectly visible by their effect on the execution speed of some programs. (It would be unusual for execution speed to be significantly affected.)

The way selected by the PLRU bits is shown in Table 7-1.

Table 7-1. Way Selected for Allocation on Cache Miss

B2:0	Way Selected for Allocation
0	Way 0
1	Way 2
10	Way 1
11	Way 2
100	Way 0
101	Way 3
110	Way 1
111	Way 3

The PLRU bits are cleared when the cache is disabled, and they are updated on a cache hit. Table 7-2 shows the next state of the PLRU bits when they are updated.

Table 7-2. Next State of PLRU Bits on Cache Hit

Way Selected by Cache Hit	B2	B1	B0
Way 0	Unchanged	1	1
Way 1	Unchanged	0	1
Way 2	1	Unchanged	0
Way 3	0	Unchanged	0

7.1 Cache Locking

Cache locking is typically used for performance-sensitive algorithms, to assure that the program memory will be in the cache during execution. It may also be used when deterministic behavior is required, to ensure that programs always execute in the same number of cycles.

When the cache is locked, the contents of the cache do not change. The cache is locked by setting the LIC bit in the CFG register.

When a cache miss occurs while the cache is locked, no cache entry is allocated on a cache miss, and the PLRU bits are not updated.

When a cache hit occurs while the cache is locked, the cache data is passed to the CPU, and the PLRU bits are up-dated. However, the PLRU bits have no effect while the cache is locked, because no new cache entries are allocated.

7.2 Cache Invalidation

The cache does not snoop any bus cycles. Software is responsible for invalidating the cache when the CPU or any other device writes to program memory. The cache is invalidated by executing the CINV [i] instruction. This clears the valid bits for all of the cache entries.

The cache is automatically invalidated at reset and when it is disabled.

8 CPU Core Bus Arbitration

The CPU core AHB bus can be controlled by either of two bus masters:

- *CPU Core*
- *CPU DMA Controller*

The bus arbiter implements two levels of priority arbitration among the potential bus masters:

- *Group Priority*—each potential bus master is assigned to one of four priority groups. The priority among groups is linear, with group A at highest priority and group D at lowest priority.
- *Priority within a Group*—either of two arbitration policies may be selected for a group: linear priority and round-robin. Linear priority gives bus ownership to the requesting device with the lowest master number. Round-robin assigns bus ownership in a cyclic sequence (for example, 2-3-4-2-3-4) among requesting devices.

Group priority is always considered before priority within a group. Priority within a group is only considered after the group is selected.

With only two potential bus masters, the bus arbitration mechanism may seem excessively complex, however this architecture is designed for scalability to future devices which may have a greater number of potential bus masters.

The default register settings for the bus arbiter provide an efficient arbitration policy across a wide range of applications:

- *One Group*—all potential bus masters are assigned to group A.
- *Round-Robin Arbitration*—a round-robin policy avoids starving any potential bus master of bandwidth.
- Default Bus Master is CPU—when no potential bus master asserts a request for control of the bus, the CPU is given ownership of the bus. This avoids stalling the CPU when it needs to access an idle bus.

When a linear priority policy is selected for a priority group, the master number is used to resolve arbitration among devices within that group, with lower numbers receiving higher priority. Table 8-1 shows the assignment of master numbers.

Table 8-1. Master Number Assignment

POTENTIAL BUS MASTER	MASTER NUMBER
CPU	2
CPU DMA Controller	3

Whether or not any default settings are changed, software should lock the bus arbiter configuration during system initialization by writing 1 to the LOCK bit in the ARBCFGLK register.

8.1 Bus Arbiter Registers

The bus arbiter registers control and provide status for certain aspects of the bus arbiter. The bus arbiter registers are listed in Table 8-2.

Table 8-2. Bus Arbiter Registers

NAME	ADDRESS	DESCRIPTION
MASTGP	FF F000h	Master Group Register
ARBALGO	FF F004h	Arbitration Algorithm Register
DFTMAST	FF F008h	Default Master Register
ARBCFGLK	FF F00ch	Arbiter Configuration Lock Register

8.1.1 Master Group Register (MASTGP)

The MASTGP register is a 32-bit, read/write register that selects the priority group for each potential bus master. After reset, this register is clear.

31 – 6	5 – 4	3 – 2	1 – 0
Reserved	MAS4	MAS3	MAS2

MASn The Master field selects the priority group for the corresponding potential bus master.
00 – Group A.
01 – Group B.
10 – Group C.
11 – Group D.

8.1.2 Arbitration Algorithm Register (ARBALGO)

The ARBALGO register is a 32-bit, read/write register that selects the arbitration policy used for each priority group. A ter reset, this register is initialized to 0000 0001h.31

31 – 4	3	2	1	0
Reserved	GRD	GRC	GRB	GRA

GRn The Group bit selects the arbitration policy for the corresponding group.
0 – Linear priority based on master number.
1 – Round-robin.

8.1.3 Default Master Register (DFTMAST)

The DFTMAST register is a 32-bit, read/write register that selects the master number of the default bus master. After reset, this register is initialized to 0000 0002h.

31 – 4	3 – 0
Reserved	DFTMAST

DFTMAST The Default Master field specifies the master number. There are two defined values:
2h – CPU.
3h – CPU DMA controller.

8.1.4 Arbiter Configuration Lock Register (ARBCFGLK)

The ARBALGO register is a 32-bit, read/write register that is used to lock the bus arbiter configuration. When the configuration is locked, writes to the bus arbiter registers are ignored. Reads are unaffected. Once locked, the configuration may not be unlocked until the device is reset. After reset, this register is initialized to 0000 0000h.

31	1	0
Reserved		LOCK

LOCK The Lock bit controls whether the bus arbiter registers are locked.
0 – Unlocked.
1 – Locked.

9 DSP and Audio Peripherals

The CP3SP33 includes a Teak DSP for supporting high-performance audio applications. DSP software packages are available from Texas Instruments for signal processing operations commonly used to process audio data.

The DSP implements 16-bit fixed-point arithmetic with a 4bus (X, Y, Z, and Program) Harvard architecture. The fast DSP core (96 MHz) and large on-chip 24K-byte program and 24K-byte data memories satisfy a wide range of compute-intensive applications. The DSP has a 32-bit high-bandwidth bus separate from that of the CPU, and it has an independent DMA controller, for efficient bus utilization.

The CPU host has a register-based interface for controlling the DSP and accessing the DSP program and data memory space. The CPU can reset the DSP, interrupt the DSP, and assert a DSP DMA request. The DSP can assert an interrupt request to the CPU, and it can wake the CPU from a low-power mode.

Figure 9-1 shows the architecture of the DSP and audio peripherals. The DSP DMA controller is a slave on the CPU APB bus, so that the CPU host can use a DMA channel for downloading software to the DSP program and data memories. Once released to begin execution, the DSP is bus master on its own 32-bit, high-bandwidth bus, which the DSP uses to access three types of slave devices:

- *4K-byte Shared RAM*—data memory which is mapped into CPU and DSP address spaces.
- *External Bus Interface*—interface to external (off-chip) memory devices.
- *Shared Audio Peripheral APB Bus*—peripherals shared between the CPU and DSP (codec, I^2S interface, etc.).

The Audio Subsystem Controller (ASC) provides multiplexing of interrupt and DMA signals between shared peripherals and the independent interrupt and DMA controllers of the CPU and DSP.

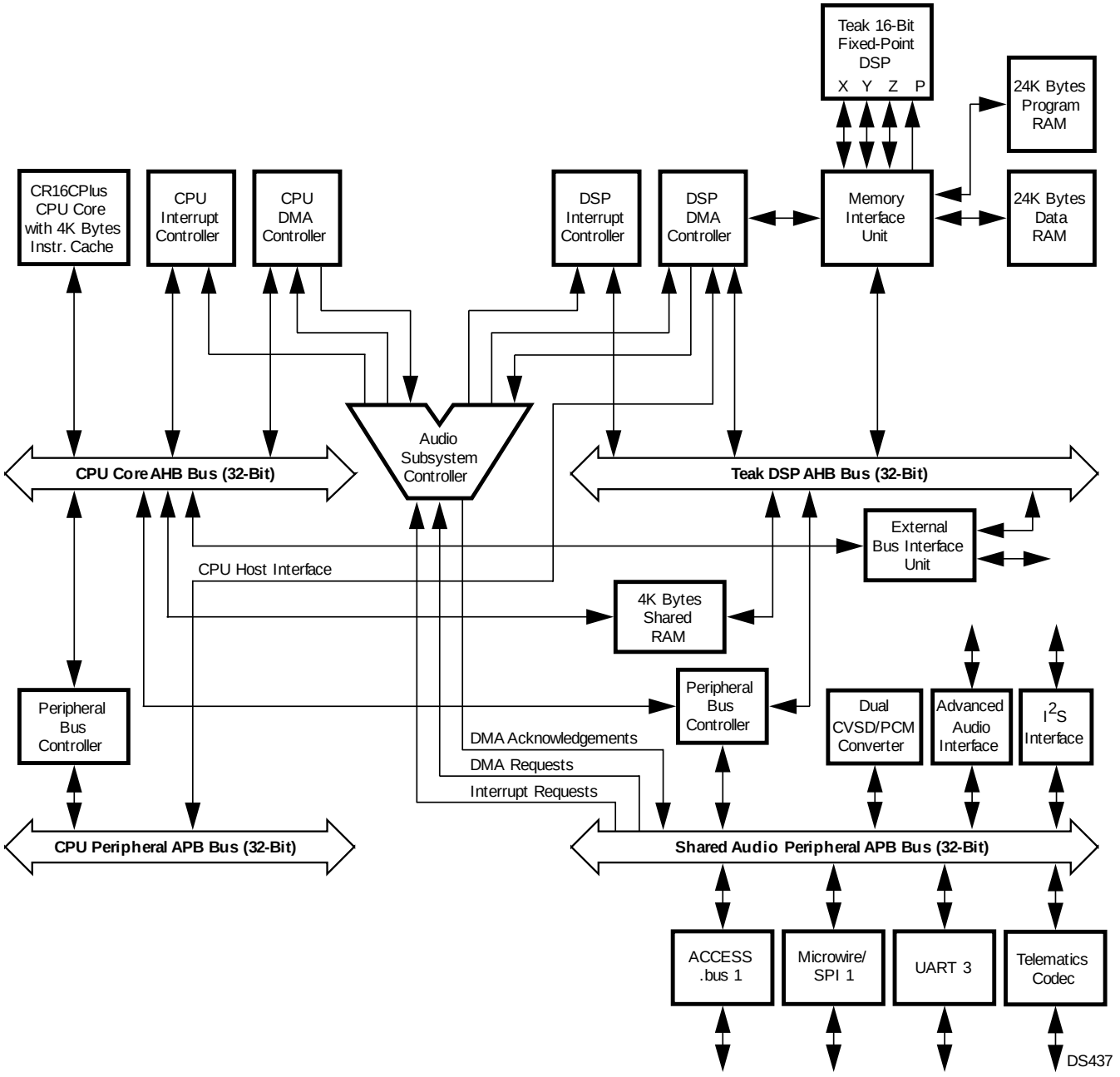


Figure 9-1. DSP and Audio Peripherals

9.1 DSP Memory Spaces

The DSP has a four-bus architecture. Instructions are received on the 16-bit P bus, the DSP data memory is accessed through the 16-bit X and Y buses, and external memory and DSP peripherals are accessed through the Z bus. Word addressing is used for all buses. Table 9-1 shows the mapping of the DSP data memory space.

Table 9-1. DSP Data Memory Map

ADDRESS RANGE	BUS	DESCRIPTION
0000h to 17FFh	X	On-chip 6K-word data memory
1800h to DFFFh	Z	Available for mapping to the external memory bus
8000h to 87FFh (Default address after reset)	Z	Memory-mapped I/O registers used by peripherals that are only directly accessible by the DSP. The base address of this region can be reprogrammed by the DSP software.
E000h to F7FFh	Y	On-chip 6K-word data memory
F800h to FFFFh	Z	Available for mapping to the external memory bus

The DSP program space is implemented as two blocks, as shown in Table 9-2. While the DSP executes from one block, the other block can be accessed by the DMA controller without degrading DSP performance.

Table 9-2. DSP Program Memory Map

ADDRESS RANGE	DESCRIPTION
0000h to 0FFFh	4K-word program memory block
1000h to 3FFFh	Reserved
4000h to 5FFFh	8K-word program memory block
6000h to FFFFh	Reserved

The DSP accesses peripherals shared with the CPU indirectly, through the DSP DMA controller or the memorymapped I/O registers (MMIO).

9.2 CPU/DSP Interface

The CPU has read and write access to the entire DSP data memory space (X, Y, Z), and write access to the DSP program memory space. The CPU/DSP interface makes use of the DSP DMA controller to avoid interrupting the DSP when servicing the CPU. The interface is hardcoded to use DSP DMA channel 0 for DMA transfers between its FIFOs and DSP memory. This DMA channel has a slave interface on the CPU peripheral data bus.

The CPU/DSP interface also includes hardware features to support interprocessor communications such as a command/reply interface and semaphores. It also provides mechanisms for the CPU and DSP to interrupt each other.

Data is transferred between the CPU and the DSP memory spaces through a set of indirect address and data registers. An auto-increment address function eliminates the need to resend the address for every word transferred. By default, the number of transactions is limited to 65535 (the default DMA transfer size). The interface contains a 16-word read FIFO and a 16-word write FIFO to buffer transfers. These FIFOs can be programmed to generate interrupts to the CPU on particular events (full, empty, etc.). Additionally, the PBUSY bit in the ADMAS register indicates when the interface cannot accept another access, for example if the write FIFO is full or a new operation is initiated before the current one completes.

The CPU/DSP interface cannot be used when HCLK Clock or PCLK Clock is faster than DSP clock.

9.2.1 DSP Reset

There are two reset bits for the DSP:

- *ADMAC.DSPRSTN*—this bit in the CPU/DSP interface asserts a reset signal that is external to the DSP module.
- *PCFG.DSPR*—this bit within the DSP module can be used to hold the DSP in reset.

The ADMAC.DSPRSTN bit is like a power-on reset, which should be used when complete reset of the DSP is required. The DSP memories cannot be accessed while the ADMAC.DSPRSTN bit is clear (reset asserted). The DSP clock (Auxiliary Clock 7) must be enabled for at least two cycles before the ADMAC.DSPRSTN bit is set (reset deasserted). After deasserting reset, the PSTS.PRST bit can be polled to determine when the DSP has completed its reset sequence.

The PCFG.DSPR bit can be used to hold the processor in reset while accessing the DSP block, for example while downloading software to the DSP program memory. When asserted, it must be held for eight DSP clock cycles before being deasserted.

9.2.2 Downloading to DSP Program Memory

The CPU interface can be used to download software to the DSP memories. After power-on-reset or other system resets, the DSP will be held in reset with the DSP clock (Auxiliary Clock 7) enabled. The DSP clock must be enabled during reset to properly reset the DSP.

1. Enable the clock to the DSP (if necessary).
2. Set the DSPR bit in the PCFG register. This will hold the DSP in reset.
3. Deassert the reset signal to the DSP module by setting the DSPRSTN bit in the ADMAC register. A delay of at least two DSP clocks must be maintained between enabling the DSP clock and deasserting the reset signal.
4. Poll the PRST bit in the PSTS register until a 1 to 0 transition is observed, which indicates that the MMIO registers that control a DMA transfer have been reset.
5. Set up and perform the transfer.
6. Clear the DSPR bit in the PCFG register, which restarts the DSP.

9.2.3 Audio Subsystem Controller (ASC)

As shown in Figure 9-2, the ASC multiplexes the interrupt and DMA signals between the independent interrupt and DMA controllers for the CPU and DSP. Peripherals capable of requesting interrupts from both the CPU and DSP must have their selections programmed in the ASCINTSEL register, in addition to having their requests enabled in the interrupt controllers. Peripherals capable of requesting DMA from both the CPU and DSP must have their selections programmed in the ASCDMASELn registers, in addition to having their requests enabled in the DMA controllers. By default after reset, all interrupt and DMA requests are routed to the CPU interrupt and DMA controllers.

The DSP DMA controller provides eight DMA request inputs and eight DMA acknowledge outputs. For each DSP DMA channel, a 5-bit field in the one of the ASCDDMASELn registers selects which of the 21 sources of DMA requests is serviced by that channel.

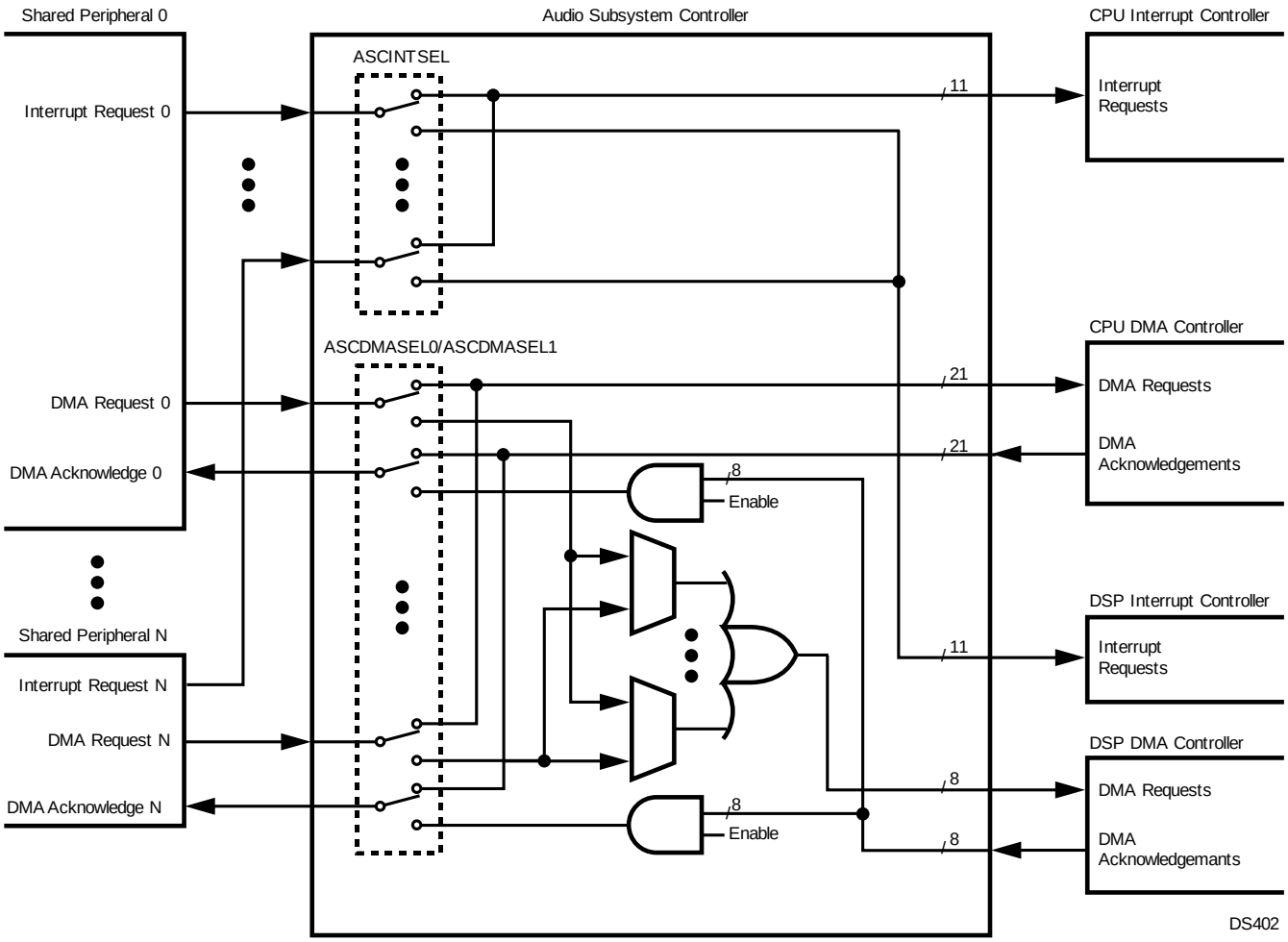


Figure 9-2. Audio Subsystem Controller

The ASC multiplexes interrupt request signals from devices capable of requesting interrupts from either of the CPU and DSP interrupt controllers. These requests are assigned to ASC interrupt channels 0 to 10, as shown in Table 9-3.

The ASC multiplexes DMA request/acknowledge signals from devices capable of requesting DMA from either of the CPU and DSP DMA controllers. These requests are assigned to ASC DMA channels 0 to 20, as shown in Table 9-4.

Table 9-3. ASC Interrupt Channel Assignment

CHANNEL	PERIPHERAL
0	Microwire/SPI 1
1	ACCESS.bus 1
2	UART3 TX
3	UART3 RX
4	CVSD/PCM 1
5	CVSD/PCM 0
6	I2S Interface
7	AAI
8	Telematics Codec
9	Multifunction Timer 1 TB1
10	Multifunction Timer 1 TA1

Table 9-4. ASC DMA Channel Assignment

CHANNEL	R/W	PERIPHERAL	REGISTER
0	R	UART3	RXBUF3
1	W	UART3	TXBUF3
2	R	CVSD/PCM0	PCMOUT0
3	W	CVSD/PCM0	PCMIN0
4	R	CVSD/PCM1	PCMOUT1
5	W	CVSD/PCM1	PCMIN1
6	R	AAI Slot 0	ARDR0
7	W	AAI Slot 0	ATDR0
8	R	AAI Slot 1	ARDR1
9	W	AAI Slot 1	ATDR1
10	R	Codec ADC1	TCDCADC1
11	R	Codec ADC2	TCDCADC2
12	W	Codec Left DAC	TCDCLEFT
13	W	Codec Right DAC	TCDCRIGHT
14	R	I2S Left	I2SRXDATALEFT
15	R	I2S Right	I2SRXDATARIGHT
16	W	I2S Left	I2STXDATALEFT
17	W	I2S Right	I2STXDATARIGHT
18	R	MWSPI1	MWDAT
19	W	MWSPI1	MWDAT
20	R/W	ACB1	ACB1SDA

9.3 DSP and ASC Registers

The DSP and ASC registers control and provide status for the interface between the CPU and the DSP. The DSP and ASC registers are listed in Table 9-5.

Table 9-5. DSP and ASC Registers

NAME	ADDRESS	DESCRIPTION
ADMAC	FF 6820h	ASC DSP Control Register
ADMAS	FF 681Ch	ASC DSP Status Register
PDATA	FF EC00h	DSP DMA Channel 0 Data Register
PADR	FF EC04h	DSP DMA Channel 0 Address Register
PCFG	FF EC08h	CPU/DSP Interface Configuration Register
PSTS	FF EC0Ch	CPU/DSP Interface Status Register
PSEM	FF EC10h	CPU-to-DSP Semaphore Register
PMASK	FF EC14h	CPU Interrupt Mask Register
PCLEAR	FF EC18h	CPU-to-DSP Semaphore Clear Register
APBP_SEM	FF EC1Ch	DSP-to-CPU Semaphore Register
APBP_COM0	FF EC20h	CPU-to-DSP Command Register 0
APBP_REP0	FF EC24h	DSP-to-CPU Reply Register 0
APBP_COM1	FF EC28h	CPU-to-DSP Command Register 1
APBP_REP1	FF EC2Ch	DSP-to-CPU Reply Register 1
APBP_COM2	FF EC30h	CPU-to-DSP Command Register 2
APBP_REP2	FF EC34h	DSP-to-CPU Reply Register 2
ASCDMASEL0	FF 6800h	ASC DMA Controller Select Register 0
ASCDMASEL1	FF 6804h	ASC DMA Controller Select Register 1

Table 9-5. DSP and ASC Registers (continued)

NAME	ADDRESS	DESCRIPTION
ASCDDMASEL0	FF 6808h	ASC DSP DMA Channel Select Register 0
ASCDDMASEL1	FF 680Ch	ASC DSP DMA Channel Select Register 1
ASCDDMASEL2	FF 6810h	ASC DSP DMA Channel Select Register 2
ASCDDMASEL3	FF 6814h	ASC DSP DMA Channel Select Register 3
ASCINTSEL	FF 6818h	ASC Interrupt Controller Select Register

9.3.1 ASC DSP Control Register (ADMAC)

The ADMAC register is a 16-bit, read/write register used by the CPU to reset the DSP. After reset, this register is 0000h.

15	1	0
Reserved		DSPRSTN

DSPRSTN The DSP Reset bit is used to assert reset to the DSP. The DSP reset signal is the OR function of this bit and the CPU reset signal. CPU software must set this bit to enable the DSP.
0 – DSP module reset.
1 – DSP module may run (if PCFG.DSPR is clear).

9.3.2 ASC DSP Status Register (ADMAS)

The ADMAS register is a 16-bit, read-only register that indicates whether the CPU host interface to the DSP DMA controller is busy and whether the DSP in reset. After reset, this register is 0000h.

15	2	1	0
Reserved		RRSTN	PBUSY

PBUSY The Port Busy bit indicates whether the CPU host interface to the DSP DMA controller is busy. If the DSP write FIFO is full, the DSP read FIFO is empty (and a read has been requested), or the PADR register has been written, this bit will be set. Software must poll the PBUSY bit before using the CPU host interface to access the DSP memory space.
0 – CPU host interface is ready.
1 – CPU host interface is busy.

RRSTN The Reflected Reset bit indicates whether the DSP is being held in reset.
0 – DSP is in reset.
1 – DSP is not in reset.

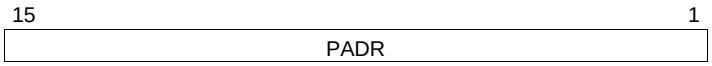
9.3.3 DSP DMA Channel 0 Data Register (PDATA)

The PDATA register is a 16-bit, read/write register that provides the CPU with access to the read and write FIFOs for DSP DMA channel 0. Writing the register loads the write FIFO, and reading the register unloads the read FIFO.

15	1
PDATA	

9.3.4 DSP DMA Channel 0 Address Register (PADR)

The PADR register is a 16-bit, write-only register that provides the lower 16 address bits for DSP DMA Channel 0. The page address (page 0) must be programmed in the DSP DMA controller. In autoincrement mode (PCFG.AIM = 1), this address is autoincremented on every DMA cycle. After reset, this register is clear.



9.3.5 CPU/DSP Interface Configuration Register (PCFG)

The PADR register is a 16-bit, read/write register that controls the configuration of the CPU/DSP interface. After reset, this register is clear.

15 12	11	10	9	8	7	6	5	4	3 2	1	0
MEMSEL	RRIE2	RRIE1	RRIE0	WFEIE	WFFIE	RFNEIE	RFFIE	RS	DRS	AIM	DSPR

DSPR — The DSP Reset bit, when set, resets the DSP. To properly reset the DSP, this bit must remain set for 8 DSP clock cycles.
0 – DSP reset is not asserted.
1 – DSP reset is asserted.

AIM — The Autoincrement Mode bit enables automatic increment of the address counter when the CPU uses DSP DMA channel 0 (the host interface used for downloading to the DSP).
0 – No increment of the DMA address.
1 – Automatic increment enabled.

DRS — The Data Read Select field selects the transfer mode used on the 32-bit high-performance DSP bus to read data for satisfying a CPU read request. This field is ignored if the RS bit is clear.
00 – Single read transfer.
01 – 8-word burst read transfer.
10 – 16-word burst read transfer.
11 – Free running (continuous).

RS — Setting the Read Start bit initiates a read transfer of the type specified in the DRS field. Clearing the bit flushes the read FIFO and terminates any read transfer in progress.
0 – No transfer is initiated or in progress.
1 – Write 1 to initiate a read transfer.

RFFIE — The Read FIFO Full Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller when the read FIFO becomes full.
0 – Interrupt disabled.
1 – Interrupt enabled

RFNEIE — The Read FIFO Not Empty Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller when the read FIFO holds valid data.
0 – Interrupt disabled.
1 – Interrupt enabled.

WFFIE — The Write FIFO Full Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller when the write FIFO is full.
0 – Interrupt disabled.
1 – Interrupt enabled.

WFEIE — The Write FIFO Empty Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller when the write FIFO is empty.
0 – Interrupt disabled.
1 – Interrupt enabled.

RRIE0 The Reply Register 0 Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller on DSP writes to the APBP_REP0 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RRIE1 The Reply Register 1 Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller on DSP writes to the APBP_REP1 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RRIE2 The Reply Register 2 Interrupt Enable bit asserts IRQ60 to the CPU interrupt controller on DSP writes to the APBP_REP2 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

MEMSEL The Memory Select field selects the DSP memory space accessed by the CPU. Values other than those listed below are reserved. 0000 – DSP data memory.
0001 – MMIO registers.
0101 – DSP program memory.
0110 – Expansion bus (16M EBIU region).
0111 – Expansion bus (240M EBIU region).

9.3.6 CPU/DSP Interface Status Register (PSTS)

The PSTS register is a 16-bit, read-only register that provides status bits for the CPU/DSP interface.

7	6	5	4	3	2	1	0
WFFI	RFNEI	RFFI	Reserved		PRST	WTIP	RTIP

15	14	13	12	11	10	9	8
RCOMIM2	RCOMIM1	RCOMIM0	RRI2	RRI1	RRI0	PSEMI	WFEI

RTIP The Read Transfer in Progress bit indicates when a read transfer is in progress. A read transfer can be terminated while it is in progress by writing 0 to the PCFG.RS bit.
0 – No transfer is in progress.
1 – A transfer is in progress.

WTIP The Write Transfer In Progress bit indicates when a write transfer is in progress.
0 – No transfer is in progress.
1 – A transfer is in progress.

PRST The Peripheral Reset Indicator bit is asserted from the start of DSP reset (PCFG.DSPR = 1) until the reset has completed. When the bit goes clear, the MMIO registers have received their reset values. The CPU must poll this bit before downloading software to the DSP.
0 – DSP is not in reset.
1 – DSP reset in progress.

RFFI The Read FIFO Full Indicator bit is asserted when the read FIFO is full.
0 – FIFO is not full.
1 – FIFO is full.

RFNEI The Read FIFO Not Empty Indicator bit is asserted when the read FIFO holds valid data.
0 – FIFO is empty.
1 – FIFO holds valid data.

WFFI The Write FIFO Full Indicator bit is asserted when the write FIFO is full.
0 – FIFO is not full.
1 – FIFO is full.

WFEI The Write FIFO Empty Indicator bit is asserted when the write FIFO is empty.
0 – FIFO is not empty.
1 – FIFO is empty.

PSEMI The Peripheral Semaphore Access Indicator bit is set when a bit in the APBP_SEM register has been set by the DSP and the corresponding bit in the PMASK register is clear.
0 – No unmasked semaphore bit is set.
1 – An unmasked semaphore bit is set.

PRI0 The Reply Register 0 Indicator bit is set when the DSP writes the APBP_REP0 register. The RRI0 bit is cleared when the CPU reads the APBP_REP0 register.
0 – APBP_REP0 has not been written.
1 – APBP_REP0 was written but not read.

RRI1 The Reply Register 1 Indicator bit is set when the DSP writes the APBP_REP1 register. The RRI1 bit is cleared when the CPU reads the APBP_REP1 register.
0 – APBP_REP1 has not been written.
1 – APBP_REP1 was written but not read.

RRI2 The Reply Register 2 Indicator bit is set when the DSP writes the APBP_REP2 register. The RRI2 bit is cleared when the CPU reads the APBP_REP2 register.
0 – APBP_REP2 has not been written.
1 – APBP_REP2 was written but not read.

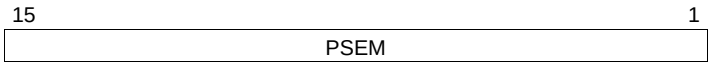
RCOMIM0 The Read Command Register 0 Indicator bit is set when the CPU writes the APBP_COM0 register. The RCOMIM0 bit is cleared when the DSP or CPU reads the APBP_COM0 register.
0 – APBP_COM0 has not been written.
1 – APBP_COM0 was written but not read.

RCOMIM1 The Read Command Register 1 Indicator bit is set when the CPU writes the APBP_COM1 register. The RCOMIM1 bit is cleared when the DSP or CPU reads the APBP_COM1 register.
0 – APBP_COM1 has not been written.
1 – APBP_COM1 was written but not read.

RCOMIM2 The Read Command Register 2 Indicator bit is set when the CPU writes the APBP_COM2 register. The RCOMIM2 bit is cleared when the DSP or CPU reads the APBP_COM2 register.
0 – APBP_COM2 has not been written.
1 – APBP_COM2 was written but not read.

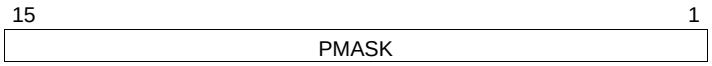
9.3.7 CPU-to-DSP Semaphore Register (PSEM)

The PSEM register is a 16-bit, read/write register used by the CPU to assert an interrupt to the DSP. If a bit in this register is set and the corresponding bit in the APBP_MASK register (only visible to the DSP) is clear, the DSP is interrupted. After reset, this register is clear.



9.3.8 CPU Interrupt Mask Register (PMASK)

The PMASK register is a 16-bit, read/write register used by the CPU to mask interrupt requests from the DSP. If the DSP sets a bit in the APBP_SEM register and the corresponding bit in the PMASK register is clear, the CPU is interrupted. After reset, this register is clear.

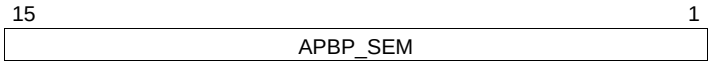


9.3.9 CPU-to-DSP Semaphore Clear Register (PCLEAR)

The PCLEAR register is a 16-bit, write-only register used by the CPU to clear interrupt requests from the DSP. Writing 1 to bits in the PCLEAR register clears the corresponding bits in the APBP_SEM register.

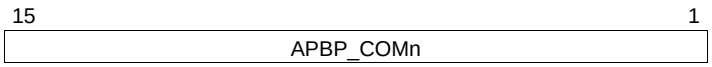
9.3.10 DSP-to-CPU Semaphore Register (APBP_SEM)

The APBP_SEM register is a 16-bit, read-only register which indicates interrupt requests from the DSP. When the DSP sets a bit in this register while the corresponding bit in the PMASK register is clear, an interrupt is asserted to the CPU. After reset, this register is clear.



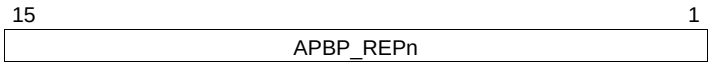
9.3.11 CPU-to-DSP Command Register n (APBP_COMn)

The APBP_COMn registers are 16-bit registers that are read/write to the CPU and read-only to the DSP (as MMIO registers). After reset, these registers are clear.



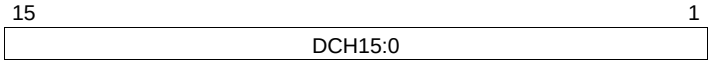
9.3.12 DSP-to-CPU Reply Register n (APBP_REPn)

The APBP_REPn registers are 16-bit registers that are read/write to the DSP (as MMIO registers) and read-only to the CPU. After reset, these registers are clear.



9.3.13 ASC DMA Controller Select Register n (ASCDMASELn)

The ASCDMASELn registers are 16-bit, read/write registers that select between the CPU and DSP DMA controllers for the ASC DMA channels (listed in Table 9-4). Bits 15:0 of ASCDMASEL0 correspond to ASC DMA channels 15:0, and bits 4:0 of ASCDMASEL1 correspond to ASC DMA channels 20:16. Clear bits select the CPU DMA controller, and set bits select the DSP DMA controller. After reset, these registers are clear.



15 – 5	4	3	2	1	0
Reserved	DCH20	DCH19	DCH18	DCH17	DCH16

DCHn The DMA Controller Select bit controls which DMA controller receives the DMA request and returns the DMA acknowledge signal for the corresponding ASC DMA channel.
0 – CPU DMA controller is selected.
1 – DSP DMA controller is selected.

9.3.14 ASC DSP DMA Channel Select Register n (ASCDDMASELn)

The ASCDDMASELn registers are 16-bit, read/write registers that provide 5-bit fields for specifying the ASC DMA channels (listed in Table 12) that activate the corresponding DSP DMA channels. Each ASCDDMASELn register provides fields for two DMA channels. ASCDDMASEL0 controls channels 0 and 1, ASCDDMASEL1 controls channels 2 and 3, ASCDDMASEL2 controls channels 4 and 5, and ASCDDMASEL3 controls channels 6 and 7. After reset, these registers are clear.

15 – 13	12 – 8	7 – 5	4 – 0
Reserved	DDMA(2n+1)	Reserved	DDMA(2n)

DDMA The ASCDDMASELn registers are 16-bit, read/write registers that provide 5-bit fields for specifying the ASC DMA channels (listed in Table 9-4) that activate the corresponding DSP DMA channels. Each ASCDDMASELn register provides fields for two DMA channels. ASCDDMASEL0 controls channels 0 and 1, ASCDDMASEL1 controls channels 2 and 3, ASCDDMASEL2 controls channels 4 and 5, and ASCDDMASEL3 controls channels 6 and 7. After reset, these registers are clear.

9.3.15 ASC Interrupt Select Register (ASCINTSEL)

The ASCINTSEL register is a 16-bit, read/write register that selects between the CPU and DSP interrupt controllers for the ASC interrupt channels (listed in Table 9-3). Clear bits select the CPU interrupt controller, and set bits select the DSP interrupt controller. After reset, this register is clear.

15 – 11	10 – 0
Reserved	ICH

ICH The Interrupt Controller Select bit controls which interrupt controller receives the interrupt request from the corresponding ASC interrupt channel.
0 – CPU interrupt controller is selected.
1 – DSP interrupt controller is selected.

10 External Bus Interface Unit

The External Bus Interface Unit (EBIU) provides a memory bus interface that supports three types of external memory:

- Asynchronous RAM
- Page-Mode Flash Memory
- ROM

Because the CP3SP33 does not have on-chip non-volatile memory, the EBIU is typically used to add at least one non-volatile memory device for holding the boot code. The EBIU provides programmable chip select outputs $\overline{\text{XCSn}}$ for two or three memory devices, depending on package type. The memory type, base address, width (8, 16, or 32 bits), size, and timing are independently programmable for each of the memory devices.

The physical address space available on the external bus does not occupy a fixed location in the address space of the CPU or Teak on-chip 32-bit buses. The location(s) are programmable in the EBIU registers. The EBIU asserts a chip select to an external memory device when an on-chip bus uses a region of its address space mapped to an external memory device. There are two regions in the on-chip bus address space which are available for external devices:

- *0002 0000h to 00FE FFFFh*—a 16M-byte space, minus a 128K space at the bottom and a 64K space at the top
- *0100 0000h to 0FFF FFFFh*—a 240M-byte space (256M, minus a 16M space at the bottom)

The EBIU does not relocate the address passed from the on-chip AHB buses to the external bus. The mapping only affects activation of the chip select signals. Any of the chip selects may be configured for either of two regions.

10.1 External Bus Signals

There are 6 types of bus signals. The number of signals varies with the package type, as shown in Table 10-1.

Table 10-1. External Bus Signals

NAME	PACKAGE PIN COUNT		DESCRIPTION
	FBGA-224	FBGA-144	
XAn	23	21	Address Bus
XDn	32	16	Data Bus
XWE	1	1	Write Enable
XOE	1	1	Output Enable
XBEn	4	2	Byte Enables
XCSn	3	2	Chip Selects

The byte order is little-endian (LSB at lowest address). The EBIU shifts the address to correspond to the width of the external memory device, so the XA address may be either a byte, word, or doubleword address. Mixing of memory types is allowed.

10.2 Default Memory Configuration

The default configuration after reset for the external memory devices is shown in Table 10-2. Booting from an 8-bit external memory is not supported.

The cycle times are quoted in HCLK Clock periods. By default, the HCLK Clock frequency following reset is the input clock frequency (X1CLKI/X1CLKO or CLKIN) divided by 16.

Table 10-2. Default Memory Configuration

PARAMETER	CHIP SELECT		
	XCS2	XCS1	XCS0
Memory Type	RAM	RAM	RAM
Base Address	0100 0000h	0080 0000h	0040 0000h
Memory Size	16M Bytes	8M Bytes	8M Bytes
Memory Width	16 Bits	32 Bits	16 Bits
Read Cycle Time	22 Cycles	22 Cycles	22 Cycles
RAM Write Address Setup Time	1 Cycle	1 Cycle	1 Cycle
RAM Write Address and Data Hold Time	1 Cycle	1 Cycle	1 Cycle
RAM Write Pulse Width	4 Cycles	4 Cycles	4 Cycles
Bus Turnaround Time	2 Cycles	2 Cycles	2 Cycles
Page-Mode	Disabled	Disabled	Disabled
Page-Mode Read Cycle Time	4 Cycles	3 Cycles	1 Cycle
Page Size	4 Words	4 Words	4 Words
Memory Read Pipe Stages	0	0	0

10.3 External Bus Cycle Timing

Figure 10-1 shows the timing of a read cycle on the external memory bus.

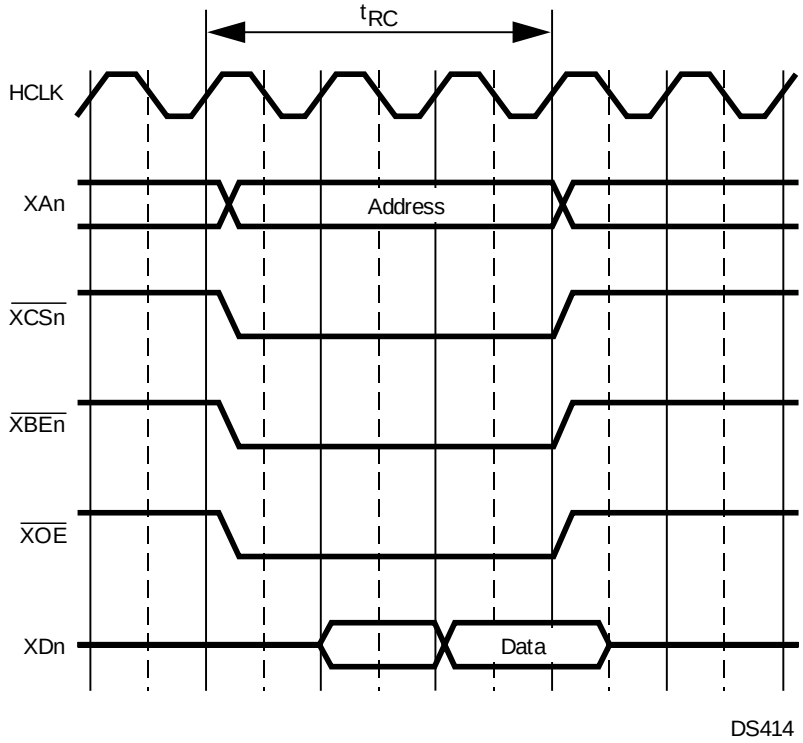


Figure 10-1. Read Cycle Timing

Figure 10-2 shows the timing of a write cycle on the external memory bus.

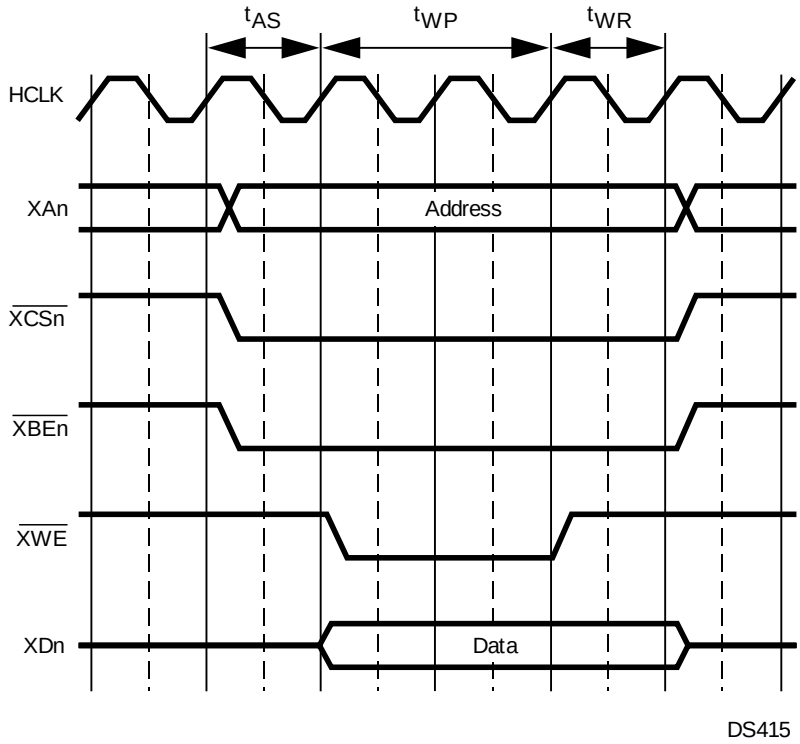


Figure 10-2. Write Cycle Timing

An 8-level write buffer allows the EBIU to accept a write cycle and release the bus master (CPU, DSP, or one of the DMA controllers) to continue execution without waiting for the write data to propagate to external memory.

A read cycle is not allowed to propagate to external memory while there are any writes pending in the write buffer. The read cycle is stalled until the write buffer has been flushed to memory. This mechanism enforces memory coherency when the target of a read is modified by a pending write. In some cases, software performance can be optimized by grouping writes together (for example, by loop unrolling) to avoid alternating between reads and writes.

Figure 10-3 shows the timing of page-mode read cycles.

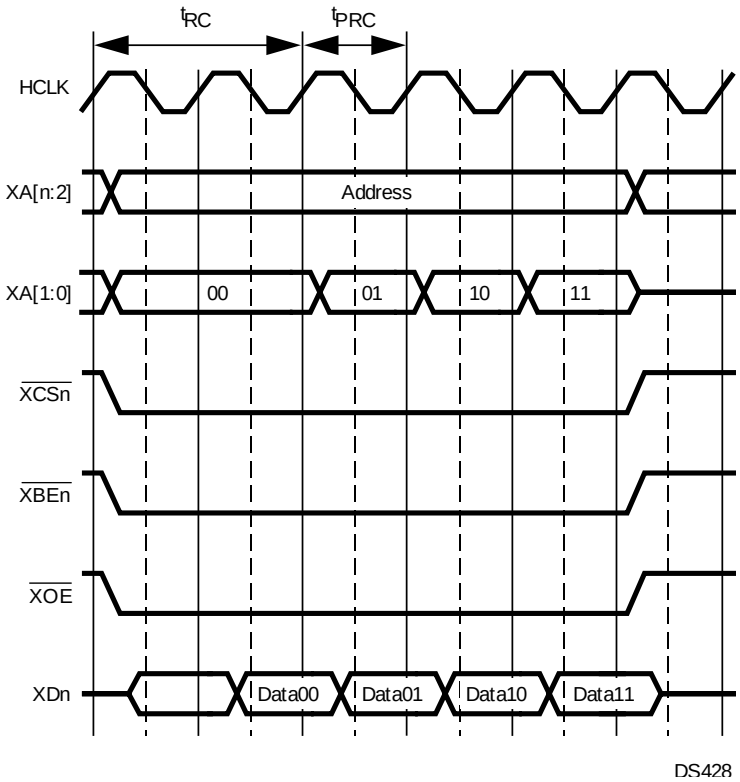


Figure 10-3. Page-Mode Read Cycle Timing

Figure 10-4 shows the read-to-write and write-to-read bus turn-around timing.

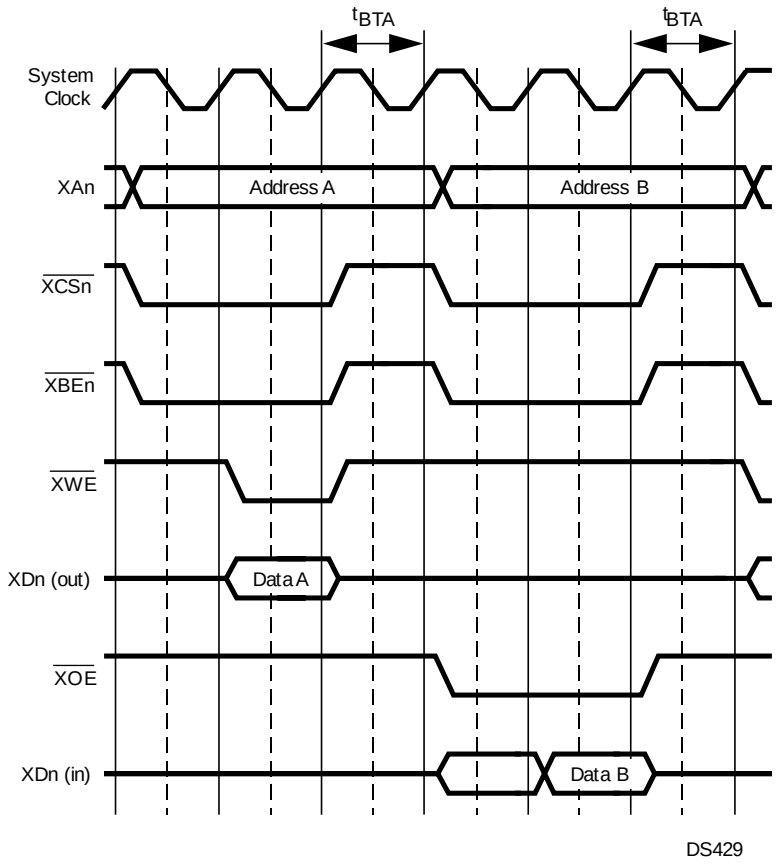


Figure 10-4. Bus Turnaround Timing

10.4 EBIU Registers

For all registers, the contents should only be changed once, in the initialization routine after reset.

The EBIU has two global control registers, two registers specific to each chip select, and 3 registers for defining 3 sets of timing parameters. Any of the chip selects may use any of the timing parameters.

Table 10-3. Bus Control Registers

NAME	ADDRESS	DESCRIPTION
SMCTLR	FF 00A4h	Static Memory Control Register
SCSLR0	FF 0014h	Chip Select Register 0
SMSKR0	FF 0054h	Mask Register 0
SCSLR1	FF 0018h	Chip Select Register 1
SMSKR1	FF 0058h	Mask Register 1
SCSLR2	FF 001Ch	Chip Select Register 2
SMSKR2	FF 005Ch	Mask Register 2
SMTMGR_SET0	FF 0094h	Static Memory Timing Register 0
SMTMGR_SET1	FF 0098h	Static Memory Timing Register 1
SMTMGR_SET2	FF 009Ch	Static Memory Timing Register 2
FLASH_TRPDR	FF 00A0h	Flash t_{RPD} Timing Register

10.4.1 Static Memory Control Register (SMCTLR)

The SMCTLR register is a 32-bit, read/write register that selects the bus width of the memory devices associated with the chip selects. At reset, the register is initialized to 0000 0401h, which selects 16-bit width for $\overline{\text{XCS2}}$ and $\overline{\text{XCS0}}$, and 32-bit width for $\overline{\text{XCS1}}$. The register format is shown below.

31 – 16	15 – 13	12 – 10	9 – 7	6 – 0
Reserved	SM_DW_S2	SM_DW_S1	SM_DW_S0	Reserved

SM_DW_S0 The Static Memory Data Bus Width field selects the width of the $\overline{\text{XCS0}}$ device.
000 – 16 bits.
001 – 32 bits.
100 – 8 bits.

SM_DW_S1 The Static Memory Data Bus Width field selects the width of the $\overline{\text{XCS1}}$ device.
000 – 16 bits.
001 – 32 bits.
100 – 8 bits.

SM_DW_S2 The Static Memory Data Bus Width field selects the width of the $\overline{\text{XCS2}}$ device.
000 – 16 bits.
001 – 32 bits.
100 – 8 bits.

10.4.2 Chip Select Register n (SCSLRn)

The SCSLRn registers are 32-bit, read/write registers that specify base addresses for the chip selects. Only bits 31:16 can be used, because the minimum memory device size is 64K bytes. Additional bits are masked off if the size exceeds 64K. At reset, the SCSLR0 register is initialized to 0040 0000h, SCSLR1 is initialized to 0080 0000h, and SCSLR2 is initialized to 0100 0000h. The register format is shown below.

31 – 16	15 – 0
EXT_BASE_ADDR	Reserved

EXT_BASE_ADDR The External Base Address field specifies the base address of the external memory device enabled by the chip select.

10.4.3 Mask Register n (SMSKRn)

The SMSKRn registers are 32-bit, read/write registers that specify timing parameters, memory type, and memory size for the chip selects. At reset, the SMSKR0 register is initialized to 0000 0028h, SMSKR1 is initialized to 0000 0128h, and SMSKR2 is initialized to 0000 0229h. The register format is shown below.

31 11	10 8	7 5	4 0
Reserved	REG_SEL	MEM_TYPE	MEM_SIZE

MEM_SIZE The Memory SIze field specifies the size of the external memory device. Values of 12h to 1Fh are reserved.
00000 – No memory connected.
00001 – 64K bytes.
00010 – 128K bytes.
00011 – 256K bytes.
00100 – 512K bytes.
00101 – 1M bytes.
00110 – 2M bytes.
00111 – 4M bytes.
01000 – 8M bytes.
01001 – 16M bytes.
01010 – 32M bytes.
01011 – 64M bytes.
01100 – 128M bytes.
01101 – 256M bytes.
01110 – 512M bytes.
01111 – 1G bytes.
10000 – 2G bytes.
10001 – 4G bytes

MEM_TYPE The Memory Type field specifies the type of external memory device.
000 – Reserved.
001 – RAM.
010 – Flash memory.
011 – ROM.

REG_SEL The Register Select field specifies timing parameters in one of the three SMTMGR_SETn registers to be used with the external memory device.
000 – SMTMGR_SET0 register.
001 – SMTMGR_SET1 register.
010 – SMTMGR_SET2 register.

10.4.4 Static Memory Timing Register n (SMTMGR_SETn)

The SMTMGR_SETn registers are 32-bit, read/write registers that specify timing parameters that can be selected by the REG_SEL field in the SMSKRn registers. Any SMSKRn register may select any SMTMGR_SETn register. At reset, the SMTMGR_SET0 register is initialized to 0002 0D45h, the SMTMGR_SET1 register is initialized to 0012 0D45h, and the SMTMGR_SET2 register is initialized to 001A 0D45h. The register format is shown below.

31 30	29 28	27 26	25 24	23	22 19	18 16	15 10	9 8	7 6	5 0
Res.	SMRP	Res.	PS	PM	T_PRC	T_BTA	T_WR	T_WR	T)AS	T_RC

T_RC The t_{RC} field + 1 specifies the number of HCLK Clock cycles in a read cycle

T_AS The t_{AS} field specifies the number of HCLK Clock cycles in the write address setup time.
00 – Reserved.
01 – 1 cycle.
10 – 2 cycles.
11 – 3 cycles.

T_WR The t_{WR} field specifies the number of HCLK Clock cycles in the write address data hold time.

T_WP The t_{WP} field + 1 specifies the number of HCLK Clock cycles in a write pulse.

T_BTA The t_{BTA} field + 1 specifies the number of HCLK Clock cycles in the bus turnaround time. The same value is used for read-to-write and write-to-read turnaround.

T_PRC The t_{PRC} field + 1 specifies the number of HCLK Clock cycles in the page-mode read cycle time.

PM The Page Mode bit specifies whether page mode is used to access the device
0 – Page mode disabled.
1 – Page mode enabled.

PS The Page Size field specifies the page size. If the page size is larger than 64 bytes, the PS field should be set to 11b.
00 – 8 bytes.
01 – 16 bytes.
10 – 32 bytes.
11 – 64 bytes.

SMRP The Static Memory Read Pipe field specifies the number of registers inserted into the read data path for correctly latching the data.

10.4.5 Flash t_{RPD} Timing Register (FLASH_TRPDR)

The FLASH_TRPDR register is a 32-bit, read/write register that specifies the number of clock cycles between flash memory reset/power-down and the first flash read/write cycle. At reset, the FLASH_TRPDR register is initialized to 0000 00C8h.

31 12	11 0
Reserved	T_RPD

T_RPD The t_{RPD} field + 1 specifies the number of clock cycles between flash memory reset/ power-down and the first flash memory read/ write cycles. At reset, the default is 201 cycles (value = C8h = 200d).

10.5 Usage Notes

The EBIU registers must be programmed in the following sequence:

1. SCSLRn
2. SMSKRn
3. SMTMGR_SETn
4. FLASH_TRPDR
5. SMCTRL

11 System Configuration

11.1 Operating Environment

The operating environment controls the reset vector (boot address). In ERE16 mode, the vector is 0040 0000h. In ERE32 mode, the vector is 0080 0000h.

The operating mode of the device is controlled by the states sampled from the ENV[1:0] pins at reset, as shown in Table 11-1. Internal pullups on the ENV[1:0] pins select ERE16 mode if these pins are allowed to float.

Table 11-1. Operating Environment Selection

ENV[1:0]	OPERATING ENVIRONMENT
11	ERE16 mode (boot from 0040 0000h)
10	ERE32 mode (boot from 0080 0000h)

11.2 Freeze Mode

For debugging purposes, a Freeze mode is available which has the following effects:

- Disables maskable interrupts.
- Suspends DMA transactions.
- Inhibits certain state changes in the following modules (see individual module descriptions for details): Advanced Audio Interface, A/D Converter, Codec, CVSD/ PCM Converters, Multi-Function Timers, Timing and Watchdog Module, and Versatile Timer Units.
- Inhibits automatic clear-on-read function applied to the register bits listed in Table 11-2.

Table 11-2. Register Bits Affected By Freeze Mode

MODULE	REGISTERS	BITS
ACCESS.bus	ACBnST	SDAST
A/D Converter	ADCRESLT	ADC_DONE, ADC_OFLW, SIGN, ADC_RESULT
Advanced Audio Interface (AAI)	ARSCR	RXO, RXE, RXF, RXAF
	ATSCR	TXU, TXF, TXE, TXAE
Codec	TCDCADCn	ADCDATA
CVSD/PCM Converter	CVSTATn	PCMINT, CVE, CVF
Microwire/SPI	MWnDAT	RBF
Timing and Watchdog Module	T0CSR	TC
USART0, UART 1, 2, 3	URBUFn	URBF
	USTATn	UPE, UFE, UDOE, UBKD

Freeze mode can be entered by setting the FREEZE bit in the MCFG register or using a mechanism enabled through the Serial Debug Interface (SDI). Freeze mode does not directly affect the DSP subsystem.

NOTE

Debugging tools may assert Freeze mode to gather information, which may cause periodic fluctuations in response (bus availability, interrupt latency, etc.). Anomalous behavior often may be traced to the activity of these tools.

11.3 System Configuration Registers

The system configuration registers control and provide status for certain aspects of device setup and operation, such as indicating the states sampled from the ENV[1:0] inputs. The system configuration registers are listed in Table 11-3.

Table 11-3. System Configuration Registers

NAME	ADDRESS	DESCRIPTION
MCFG	FF F400h	Module Configuration Register
MSTAT	FF F404h	Module Status Register
SWRESET	FF F408h	Software Reset Register
SYSCFG	FF F40Ch	System Configuration Register

11.4 Module Configuration Register (MCFG)

The MCFG register is a byte-wide, read/write register that selects the clock output features of the device and enables Freeze mode. The MCFG register format is shown below. At reset, the MCFG register is initialized to 00h.

7	6	5	4	3	2	1	0
Reserved	FREEZE	Reserved		ENV1SEL	ENV0SEL	ENV1OE	ENV0OE

ENV0OE The ENV0 Output Enable bit enables driving an internal clock signal selected by the ENV0SEL bit on the ENV0 pin. If no signal is driven on the ENV0 pin, it will be undriven (high impedance) after reset.
0 – ENV0 is undriven after reset.
1 – ENV0 is driven by an internal clock signal.

ENV1OE The ENV1 Output Enable bit enables driving an internal clock signal selected by the ENV1SEL bit on the ENV1 pin. If no signal is driven on the ENV1 pin, it will be undriven (high impedance) after reset.
0 – ENV1 is undriven after reset.
1 – ENV1 is driven by an internal clock signal.

ENV0SEL The ENV0 Select bit selects an internal clock signal to be available for driving on the ENV0 pin. The ENV0OE bit must be set to enable driving the selected signal on the ENV0 pin.
0 – Slow Clock is available to drive on ENV0.
1 – PLL1 Clock is available to drive on ENV0.

ENV1SEL The ENV1 Select bit selects an internal clock signal to be available for driving on the ENV1 pin. The ENV1OE bit must be set to enable driving the selected signal on the ENV1 pin.
0 – Main Clock is available to drive on ENV1.
1 – PLL2 Clock is available to drive on ENV1.

FREEZE The Freeze bit controls whether the device is placed in Freeze mode, which is a debugging mode that inhibits certain automatic state changes in register bits, such as timers and clear-on-read registers. See Section 11.2 for more information about Freeze mode. The Serial Debug Interface (SDI) also has the capability of putting the device in Freeze mode through a separate mechanism.
0 – Freeze mode disabled.
1 – Freeze mode enabled.

11.5 Module Status Register (MSTAT)

The MSTAT register is a byte-wide, read-only register that indicates the general status of the device. The MSTAT register format is shown below. At reset, the MSTAT register is initialized to 00h, except for bits 0 and 1 which are sampled from the ENV pins.

7	6	5 – 2	1 – 0
ISPRST	WDRST	Reserved	OENV

OENV The Operating Environment bits hold the states sampled from the ENV[1:0] input pins at reset. These states are controlled by external hardware at reset and are held constant in the register until the next reset.

WDRST The Watchdog Reset bit indicates that a Watchdog timer reset has occurred. Write a 1 to this bit to clear it. Power-on reset also clears this bit.
0 – No Watchdog timer reset has occurred since this bit was last cleared.
1 – A Watchdog timer reset has occurred since this bit was last cleared.

ISPRST The Software ISP Reset bit indicates that a SWRESET(ISP) reset has occurred since the bit was last cleared. This bit is cleared by a SWRESET(CLR) sequence or a power-on reset. See the description of the SWRESET register for more information.
0 – No SWRESET(ISP) reset has occurred since this bit was last cleared.
1 – A SWRESET(ISP) reset has occurred since this bit was last cleared.

11.6 Software Reset Register (SWRESET)

The SWRESET register is an 8-bit, write-only register which provides a mechanism for software to initiate a reset. There are two software reset sequences, called SWRESET(ISP) and SWRESET(CLR), which are provided for compatibility with other CP3000 devices that have an in-system programming (ISP) mode, however the CP3SP33 does not have an ISP mode.

To initiate a SWRESET(ISP) reset, write the value E1h to the SWRESET register, followed within 127 PCLK Clock cycles by writing the value 3Eh. The reset then follows immediately. After the SWRESET(ISP) reset occurs, the ISPRST bit in the MSTAT register is set.

To initiate a SWRESET(CLR) reset, write the value E1h to the SWRESET register, followed within 127 PCLK Clock cycles by writing the value 0Eh. The reset then follows immediately. After the SWRESET(CLR) reset occurs, the ISPRST bit in the MSTAT register is clear.

11.7 System Configuration Register (SYSCFG)

The SYSCFG register is a byte-wide, read/write register that indicates the general status of the device. The SYSCFG register format is shown below. At reset, the SYSCFG register is initialized to 00h.

7	6 – 4	3	2	1	0
XDPUDIS	Reserved	USBIDDIGPUEN	USBHCLKDIS	BTHCKLDIS	RFCKEN

RFCKEN	The RF Clock Enable bit controls whether Main Clock is driven on the CLKIN/RFCK pin. 0 – Main Clock is not driven on CLKIN/RFCK. 1 – Main Clock is driven on CLKIN/RFCK.
BTHCLKDIS	The Bluetooth Clock Disable bit controls whether HCLK Clock to the Bluetooth module is disabled. 0 – HCLK Clock is available to the Bluetooth module. 1 – HCLK Clock is not available to the Bluetooth module.
USBHCLKDIS	The USB Clock Disable bit controls whether the HCLK Clock to the USB module is disabled. 0 – HCLK Clock is available to the USB module. 1 – HCLK Clock is not available to the USB module.
USBIDDIGPUEN	The USB IDDIG Pullup Enable bit controls whether the USB module can enable the internal pullup on the PE15 port pin. This mode only applies when the IDDIG alternate function for PE15 is enabled. 0 – Internal pullup on PE15 is disabled when IDDIG alternate function is selected. 1 – USB module controls internal pullup on PE15 when IDDIG alternate function is selected.
XDPUDIS	The XD Pullup Disable bit controls whether weak pullup resistors are enabled on the XD external data bus. 0 – Weak pullups on XD bus. 1 – No pullups on XD bus.

12 CPU DMA Controller

The CPU DMA controller (DMAC) can be used to accelerate peripheral-to-memory, memory-to-peripheral, and memory-to-memory block transfers. Because it uses cycle stealing to interleave bus cycles with the CPU, DMA-based data movement uses the available bandwidth on the CPU core bus more efficiently than software-based data movement

The DMAC provides 16 DMA channels, which may be assigned to any of 34 peripheral registers. For registers that are loaded by the peripheral (such as a UART receive register), the DMAC gets a DMA request when the register is loaded. It then reads the register and writes the data to memory. For registers that are unloaded by the peripheral (such as a UART transmit register), the DMAC gets a DMA request when the register is empty. It then reads data from memory and writes the data to the register. Only one register at a time may be enabled to use a DMA channel. Any channel which is not enabled for peripheral DMA may be used for software DMA (memory-to-memory block transfers).

The DMAC has a register-based programming interface (as opposed to I/O control blocks). After loading the registers with source and destination addresses, as well as block size and type of operation, a DMAC channel is ready to respond to DMA transfer requests. A request can only come from onchip peripherals or software, not external peripherals. On receiving a DMA transfer request, if the channel is enabled, the DMAC performs the following operations:

1. Arbitrates to become master of the CPU core bus.
2. Determines priority among the DMAC requests. Priority is linear, with channel 0 having the highest priority.
3. Executes data transfer bus cycle(s) specified by the programming of the control registers for the channel being serviced. This may be a single cycle or a four cycle burst.
4. If the DMA transfer cycle is complete, the DMAC does the following:
   - Updates the termination bits.
   - Asserts an interrupt (if enabled).
5. Returns control of the CPU core bus to the CPU, even if a DMA request continues to be asserted. Priority among DMA channels is re-determined after every cycle.

All DMA transfers are indirect mode, in which the data is read from the source into a DMAC buffer register, then written from the buffer register to the destination.

Each DMAC channel has ten 32-bit control and status reg-isters. DMAC registers are named with the suffix n, in which n is 0 to 15, representing the channel number.

If all of the channels are disabled (DMACNTn.CHEN = 0), the clock to the DMA module is disabled to reduce power consumption.

12.1 DMA-Capable Peripherals

Table 12-1 shows the DMA-capable peripherals, which may be assigned to any of the 16 DMA channels by programming the SRCRQ field of the DMACNTn registers. The SRCRQ field only selects the source of the DMA request signal and the receiver for the DMA acknowledge signal. It is still necessary to set up the address of the peripheral, the transfer direction, and other DMA channel control settings.

Table 12-1. DMA-Capable Peripherals

SRCRQ	PERIPHERAL	TRANSACTION	REGISTER	ASC CHANNEL
0	UART3	R	RXBUF3	0
1	UART3	W	TXBUF3	1
2	CVSD/PCM0	R	PCMOUT0	2
3	CVSD/PCM0	W	PCMIN0	3
4	CVSD/PCM1	R	PCMOUT1	4
5	CVSD/PCM1	W	PCMIN1	5
6	AAI Slot 0	R	ARDR0	6
7	AAI Slot 0	W	ATDR0	7
8	AAI Slot 1	R	ARDR1	8
9	AAI Slot 1	W	ATDR1	9
10	Codec ADC1	R	TCDCADC1	10
11	Codec ADC2	R	TCDCADC2	11
12	Codec DAC Left	W	TCDCLEFT	12
13	Codec DAC Right	W	TCDCRIGHT	13
14	I^2S Left	R	I2SRXDATL	14
15	I^2S Right	R	I2SRXDATR	15
16	I^2S Left	W	I2STXDATL	16
17	I^2S Right	W	I2STXDATR	17
18	MWSPI1	R	MWDAT	18
19	MWSPI1	W	MWDAT	19
20	ACCESS.bus 1	R/W	ACBSDA	20
21	USART0	R	RXBUF0	
22	USART0	W	TXBUF0	
23	UART1	R	RXBUF1	
24	UART1	W	TXBUF1	
25	UART2	R	RXBUF2	
26	UART2	W	TXBUF2	
27	CVSD/PCM0	R	CVSDOUT0	
28	CVSD/PCM0	W	CVSDIN0	
29	CVSD/PCM1	R	CVSDOUT1	
30	CVSD/PCM1	W	CVSDIN1	
31	AAI Slot 2	R	ARDR2	
32	AAI Slot 2	W	ATDR2	

Table 12-1. DMA-Capable Peripherals (continued)

SRCRQ	PERIPHERAL	TRANSACTION	REGISTER	ASC CHANNEL
33	MWSPI0	R	MWDAT	
34	MWSPI0	W	MWDAT	
35	ACCESS.bus 0	R/W	ACBSDA	

The DMA request and DMA acknowledge signals for peripherals shared with the DSP DMA controller are sent through the Audio Subsystem Controller (ASC), which multiplexes these signals between the CPU DMA controller and the DSP DMA controller. In Table 12-1, the ASC channel number refers to the multiplexer channel used to select the DMA controller that handles the DMA request and acknowledge signals for that peripheral.

12.2 Transfer Types

The DMAC supports three transfer types:

- Single Transfers—A single read cycle followed by a single write cycle.
- Burst Transfers—A four-cycle burst read from memory followed by a four-cycle burst write to memory. Software DMA requests (memory-to-memory block transfers) are always burst transfers.
- Data Collection Transfers—A four-cycle burst read from memory followed by four single-cycle writes to a peripheral, or four single-cycle reads from a peripheral followed by a four-cycle burst write to memory.

Priority among DMA requests is linear for transfers of the same type (channel 0 has highest priority), however burst transfers and data collection transfers have priority over single transfers.

During a single transfer, the transfer cycle size (number of bytes per cycle) is controlled by the WMODE and TCS bits of the DMACNTn register, as shown in Table 12-2. During a burst, the transfer cycle size is always 4 bytes.

Table 12-2. Transfer Cycle Size

TRANSFER CYCLE SIZE	WMODE	TCS
1 Byte	0	0
2 Bytes	0	1
4 Bytes	1	X

The transfer type is selected by the SWRQ and BBE bits in the DMACNTn register, as shown in Table 12-3. Only channels 0 and 1 support data collection mode.

Table 12-3. DMA Transfer Types

TRANSFER TYPE	SWRQ	BBE
Single Cycle	0	0
Data Collection	0	1
Burst	1	X

12.2.1 DMA Buffer Flush

When data collection mode has been used to read data from a peripheral, the number of bytes received from the peripheral might not correspond to an integral number of bursts. To flush data remaining in the buffer, perform the following steps:

1. Disable the peripheral DMA request. Do not clear the DMA channel enable bit (the CHEN bit in the DMACNTn register), because clearing this bit will clear the buffer.
2. Read the BNE bit and the BLV field in the DMASTATn register. The BNE bit indicates whether there is valid data in the buffer. The BLV field indicates the number of valid bytes in the buffer.
3. Clear the BBE bit in the DMACNTn register. This enables a burst write to memory, even though the buffer is not full. Invalid entries in the buffer are written to memory with undefined data.

A buffer flush will only occur when the DIR bit in the DMACNTn register is clear, and the buffer is not empty (the BNE bit in the DMASTATn register is set). If the DIR bit is set, the data in the buffer is discarded.

12.3 Transfer Modes

A DMAC channel may be used in one of three different transfer modes:

- Single Buffer—The transfer performed for the channel is specified by the states sampled from the control registers when the last DMA request was enabled.
- Double Buffer—The transfer performed for the channel is specified by the states sampled from the control registers when the last DMA request was enabled. A new set of states is loaded into the control registers to specify the next transfer to be performed for the channel.
- Auto-initialize—The transfer performed for the channel is specified in its control registers, and the operation is repeated as long as the channel is enabled.

The OT bit in the DMACNTn register is clear for single buffer and double buffer mode. The OT bit is set to enable auto-initialize mode.

Double buffer mode is obtained by reloading the DMAC channel control registers after a DMA request has been enabled. A DMA request is enabled by loading the DMACNTn register with a set CHEN bit. After enabling a DMA request, the DMA control registers may be loaded with new states. The BLTRn register must be written last, because loading this register sets the VLD bit in the DMASTATn register, which indicates that new values have been loaded into the control registers.

When the PF bit in the DMACNTn register is clear, the DMAC is the flow controller for the transfer. When the PF bit is set, the peripheral is the flow controller. In double buffer mode with the peripheral as the flow controller, the BLTRn register must be written so that the VLD bit becomes set, even though the value in the BLTRn register is not used.

12.3.1 Single Buffer Mode

This mode provides the simplest way to accomplish a single data transfer.

12.3.1.1 Initialization

1. Select the peripheral device by writing the SRCRQ field in the DMACNTn register.
2. Write the block transfer addresses and byte count into the corresponding ADCAn, ADCBn, and BLTCn counters.
3. Clear the DMACNTn.OT bit to select non-auto-initialize mode. Clear the DMASTAT.VLD bit by writing a 1 to it.
4. Set the DMACNTn.CHEN bit to activate the channel and enable it to respond to DMA transfer requests. If the DMACNTn register is loaded for any other reason, the DMACNTn.CHEN bit must be clear to avoid prematurely starting a new DMA transfer.

12.3.1.2 Termination

When the DMAC is the flow controller, the transfer terminates when the transfer count in the BLTCn register reaches zero. When the peripheral is the flow controller, the peripheral signals the end of transfer. On termination:

1. The DMASTAT.TC and DMASTAT.OVR bits are set, and the DMASTAT.CHAC bit is cleared.
2. An interrupt is asserted if enabled by the DMACNTn.ETC or DMACNTn.EOVR bits.

12.3.2 Double Buffer Mode

This mode allows software to set up the next DMA transfer while the current transfer proceeds.

12.3.2.1 Initialization

1. Select the peripheral device by writing the SRCRQ field in the DMACNTn register.
2. Write the block transfer addresses and byte count into the ADCAn, ADCBn, and BLTCn counters.
3. Clear the DMACNTn.OT bit to select non-auto-initialize mode. Clear the DMASTAT.VLD bit by writing a 1 to it.
4. Set the DMACNTn.CHEN bit. This activates the cha nel and enables it to respond to DMA transfer requests. If the DMACNTn register is loaded for any other reason, the DMACNTn.CHEN bit must be clear to avoid prematurely starting a new DMA transfer.
5. While the current block transfer proceeds, write the addresses and byte count for the next block into the ADRAn, ADRBn, and BLTRn registers. The BLTRn register must be written last, because writing this register sets the DMASTAT.VLD bit which indicates to the DMAC that the parameters for the next transfer have been updated.

12.3.2.2 Continuation/Termination

When the BLTCn counter reaches 0 (DMACNTn.PF = 0) or the last request has been processed (DMACNTn.PF = 1):

1. If the DMACNTn.PF bit is clear, the DMASTATn.TC bit is set.
2. If enabled by the DMACNTn.ETC bit, an interrupt is asserted.
3. The DMAC channel checks the DMASTAT.VLD bit.

If the DMASTAT.VLD bit is set:

1. The channel copies the ADRAn, ADRBn, and BLTRn values into the ADCAn, ADCBn, and BLTCn registers.
2. The DMASTAT.VLD bit is cleared.
3. The next block transfer is started.

If the DMASTAT.VLD bit is clear:

1. The transfer operation terminates.
2. The channel sets the DMASTAT.OVR bit.
3. The DMASTAT.CHAC bit is cleared.
4. If enabled by the DMACNTn.EOVR bit, an interrupt is asserted.

12.3.3 Auto-Initialize Mode

This mode causes the DMA channel to repeat the same operation continuously without software intervention. The operation is repeated until the channel is disabled.

12.3.3.1 Initialization

1. Select the peripheral device by writing the SRCRQ field in the DMACNTn register.
2. Write the block addresses and byte count into the ADCAn, ADCBn, and BLTCn counters, as well as the ADRAn, ADRBn, and BLTRn registers.
3. Set the DMACNTn.OT bit to select auto-initialize mode.
4. Set the DMACNTn.CHEN bit to activate the channel and enable it to respond to DMA transfer requests. If the DMACNTn register is loaded for any other reason, the DMACNTn.CHEN bit must be clear to avoid prematurely starting a new DMA transfer.

12.3.3.2 Continuation

When the BLTCn counter reaches 0 (DMACNTn.PF = 0) or the last request has been processed (DMACNTn.PF = 1):

1. The contents of the ADRAn, ADRBn, and BLTRn registers are copied to the ADCAn, ADCBn, and BLTCn counters.
2. The DMAC channel checks the value of the DMASTAT.TC bit.

If the DMASTAT.TC bit is set and the DMACNTn.PF bit is clear:

1. The DMASTAT.OVR bit is set.
2. If enabled by the DMACNTn.EOVR bit, an interrupt is asserted.
3. The DMAC operation is repeated.

If the DMASTAT.TC and DMACNTn.PF bits are clear:

1. The DMASTAT.TC bit is set.
2. If enabled by the DMACNTn.ETC bit, an interrupt is asserted.
3. The DMAC operation is repeated.

12.3.3.3 Termination

The DMA transfer is terminated when the DMACNTn.CHEN bit is cleared.

12.4 Software DMA Request

In addition to the hardware requests from peripherals, a DMA transfer request can also be initiated by software. A software DMA transfer request is used for memory-to-memory block transfers.

When the DMACNTn.SWRQ bit is set, the corresponding DMA channel receives a software DMA request. When the DMACNTn.SWRQ bit is clear, the software DMA request for the channel is inactive.

A software DMA request may only be asserted when the associated peripheral DMA request is disabled and the channel is inactive. Software can poll the DMASTAT.CHAC bit to determine whether the channel is currently active.

When the DMACNTn.DIR bit is 0, the first bus cycle reads data from the source using the ADCAn counter, while the second bus cycle writes the data into the destination using the ADCBn counter. When the DMACNTn.DIR bit is set, the first bus cycle reads data from the source using the ADCBn counter, while the second bus cycle writes the data into the destination addressed by the ADCAn counter.

12.5 DMA Request Timeout

Each DMA channel may assert a timeout interrupt if too much time occurs between DMA requests. A 10-bit prescaler divides the HCLK Clock by 1024 to provide a timebase for a timeout counter, as shown in Figure 12-1.

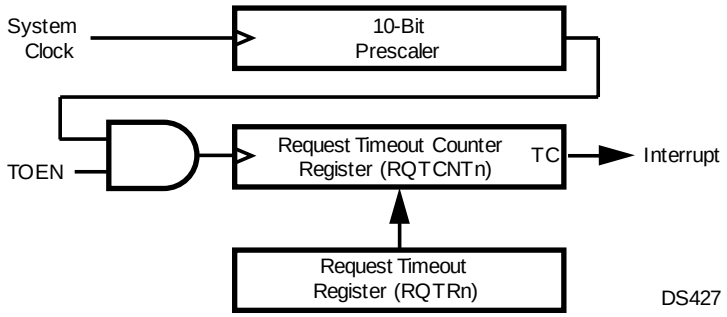


Figure 12-1. DMA Request Timeout Counter

Software loads a timeout interval in the RQTRn register. When a DMA request is received, the RQTCNTn register is loaded with the contents of the RQTRn register. If the TOEN bit is set, the RQTCNTn register decrements on every clock received from the prescaler. When the RQTCNTn register reaches its terminal count, an interrupt is asserted.

Once the timeout interrupt is enabled and a DMA request is received, the timeout interrupt will occur unless: the next DMA request is received before the end of the timeout interval, or the TOEN bit is cleared.

12.6 Error Response

When an error occurs, the DMASTATn.ERR bit is set, and the DMASTATn.CHAC bit is cleared. The current transfer cycle is completed or terminated, and the channel cannot be used again until the ERR bit is cleared (by writing 1 to it). An error may occur because the address was invalid or an incremental access goes out of bounds.

12.7 Freeze Mode

When the Freeze mode is entered, all DMA operations are stopped. Pending operations are stopped after completion of the current transfer. They will start again when the Freeze mode is exited. This allows breakpoints to be used in debug systems.

12.8 Register Programming

The DMAC only handles address-aligned transfers. The addresses loaded into ADCAn, ADRAn, ADCBn, and ADRBn must be multiples of the transfer cycle size. Do not write the counter registers ADCAn, ADCBn, or BLTCn while the channel is active (DMASTAT.CHAC = 1). When a channel is activated for DMAC flow control, the BLTCn and BLTRn registers must hold values greater than 0.

12.9 DMA Controller Register Set

There are 16 identical sets of 10 DMA controller registers, as listed in Table 12-4.

Table 12-4. DMA Controller Registers

Name	Address	Description		Name	Address	Description
ADCA0	FF 0400h	Device A Address Counter Register		ADRB4	FF 050Ch	Device B Address Register
ADRA0	FF 0404h	Device A Address Register		BLTC4	FF 0510h	Block Length Counter Register
ADCB0	FF 0408h	Device B Address Counter Register		BLTR4	FF 0514h	Block Length Register
ADRB0	FF 040Ch	Device B Address Register		RQTR4	FF 0518h	Request Timeout Register

Table 12-4. DMA Controller Registers (continued)

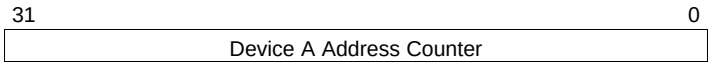
Name	Address	Description	Name	Address	Description
BLTC0	FF 0410h	Block Length Counter Register	RQTCNT4	FF 051Ch	Request Timeout Counter Register
BLTR0	FF 0414h	Block Length Register	DMACNT4	FF 0520h	DMA Control Register
RQTR0	FF 0418h	Request Timeout Register	DMASTAT4	FF 0524h	DMA Status Register
RQTCNT0	FF 041Ch	Request Timeout Counter Register	ADCA5	FF 0540h	Device A Address Counter Register
DMACNT0	FF 0420h	DMA Control Register	ADRA5	FF 0544h	Device A Address Register
DMASTAT0	FF 0424h	DMA Status Register	ADCB5	FF 0548h	Device B Address Counter Register
ADCA1	FF 0440h	Device A Address Counter Register	ADRB5	FF 054Ch	Device B Address Register
ADRA1	FF 0444h	Device A Address Register	BLTC5	FF 0550h	Block Length Counter Register
ADCB1	FF 0448h	Device B Address Counter Register	BLTR5	FF 0554h	Block Length Register
ADRB1	FF 044Ch	Device B Address Register	RQTR5	FF 0558h	Request Timeout Register
BLTC1	FF 0450h	Block Length Counter Register	RQTCNT5	FF 055Ch	Request Timeout Counter Register
BLTR1	FF 0454h	Block Length Register	DMACNT5	FF 0560h	DMA Control Register
RQTR1	FF 0458h	Request Timeout Register	DMASTAT5	FF 0564h	DMA Status Register
RQTCNT1	FF 045Ch	Request Timeout Counter Register	ADCA6	FF 0580h	Device A Address Counter Register
DMACNT1	FF 0460h	DMA Control Register	ADRA6	FF 0584h	Device A Address Register
DMASTAT1	FF 0464h	DMA Status Register	ADCB6	FF 0588h	Device B Address Counter Register
ADCA2	FF 0480h	Device A Address Counter Register	ADRB6	FF 058Ch	Device B Address Register
ADRA2	FF 0484h	Device A Address Register	BLTC6	FF 0590h	Block Length Counter Register
ADCB2	FF 0488h	Device B Address Counter Register	BLTR6	FF 0594h	Block Length Register
ADRB2	FF 048Ch	Device B Address Register	RQTR6	FF 0598h	Request Timeout Register
BLTC2	FF 0490h	Block Length Counter Register	RQTCNT6	FF 059Ch	Request Timeout Counter Register
BLTR2	FF 0494h	Block Length Register	DMACNT6	FF 05A0h	DMA Control Register
RQTR2	FF 0498h	Request Timeout Register	DMASTAT6	FF 05A4h	DMA Status Register
RQTCNT2	FF 049Ch	Request Timeout Counter Register	ADCA7	FF 05C0h	Device A Address Counter Register
DMACNT2	FF 04A0h	DMA Control Register	ADRA7	FF 05C4h	Device A Address Register
DMASTAT2	FF 04A4h	DMA Status Register	ADCB7	FF 05C8h	Device B Address Counter Register
ADCA3	FF 04C0h	Device A Address Counter Register	ADRB7	FF 05CCh	Device B Address Register
ADRA3	FF 04C4h	Device A Address Register	BLTC7	FF 05D0h	Block Length Counter Register
ADCB3	FF 04C8h	Device B Address Counter Register	BLTR7	FF 05D4h	Block Length Register
ADRB3	FF 04CCh	Device B Address Register	RQTR7	FF 05D8h	Request Timeout Register
BLTC3	FF 04D0h	Block Length Counter Register	RQTCNT7	FF 05DCh	Request Timeout Counter Register
BLTR3	FF 04D4h	Block Length Register	DMACNT7	FF 05E0h	DMA Control Register
RQTR3	FF 04D8h	Request Timeout Register	DMASTAT7	FF 05E4h	DMA Status Register
RQTCNT3	FF 04DCh	Request Timeout Counter Register	ADCA8	FF 0600h	Device A Address Counter Register
DMACNT3	FF 04E0h	DMA Control Register	ADRA8	FF 0604h	Device A Address Register
DMASTAT3	FF 04E4h	DMA Status Register	ADCB8	FF 0608h	Device B Address Counter Register
ADCA4	FF 0500h	Device A Address Counter Register	ADRB8	FF 060Ch	Device B Address Register
ADRA4	FF 0504h	Device A Address Register	BLTC8	FF 0610h	Block Length Counter Register
ADCB4	FF 0508h	Device B Address Counter Register	BLTR8	FF 0614h	Block Length Register
RQTR8	FF 0618h	Request Timeout Register	ADCB12	FF 0708h	Device B Address Counter Register
RQTCNT8	FF 061Ch	Request Timeout Counter Register	ADRB12	FF 070Ch	Device B Address Register
DMACNT8	FF 0620h	DMA Control Register	BLTC12	FF 0710h	Block Length Counter Register
DMASTAT8	FF 0624h	DMA Status Register	BLTR12	FF 0714h	Block Length Register
ADCA9	FF 0640h	Device A Address Counter Register	RQTR12	FF 0718h	Request Timeout Register
ADRA9	FF 0644h	Device A Address Register	RQTCNT12	FF 071Ch	Request Timeout Counter Register
ADCB9	FF 0648h	Device B Address Counter Register	DMACNT12	FF 0720h	DMA Control Register
ADRB9	FF 064Ch	Device B Address Register	DMASTAT12	FF 0724h	DMA Status Register
BLTC9	FF 0650h	Block Length Counter Register	ADCA13	FF 0740h	Device A Address Counter Register
BLTR9	FF 0654h	Block Length Register	ADRA13	FF 0744h	Device A Address Register
RQTR9	FF 0658h	Request Timeout Register	ADCB13	FF 0748h	Device B Address Counter Register
RQTCNT9	FF 065Ch	Request Timeout Counter Register	ADRB13	FF 074Ch	Device B Address Register

Table 12-4. DMA Controller Registers (continued)

Name	Address	Description		Name	Address	Description
DMACNT9	FF 0660h	DMA Control Register		BLTC13	FF 0750h	Block Length Counter Register
DMASTAT9	FF 0664h	DMA Status Register		BLTR13	FF 0754h	Block Length Register
ADCA10	FF 0680h	Device A Address Counter Register		RQTR13	FF 0758h	Request Timeout Register
ADRA10	FF 0684h	Device A Address Register		RQTCNT13	FF 075Ch	Request Timeout Counter Register
ADCB10	FF 0688h	Device B Address Counter Register		DMACNT13	FF 0760h	DMA Control Register
ADRB10	FF 068Ch	Device B Address Register		DMASTAT13	FF 0764h	DMA Status Register
BLTC10	FF 0690h	Block Length Counter Register		ADCA14	FF 0780h	Device A Address Counter Register
BLTR10	FF 0694h	Block Length Register		ADRA14	FF 0784h	Device A Address Register
RQTR10	FF 0698h	Request Timeout Register		ADCB14	FF 0788h	Device B Address Counter Register
RQTCNT10	FF 069Ch	Request Timeout Counter Register		ADRB14	FF 078Ch	Device B Address Register
DMACNT10	FF 06A0h	DMA Control Register		BLTC14	FF 0790h	Block Length Counter Register
DMASTAT10	FF 06A4h	DMA Status Register		BLTR14	FF 0794h	Block Length Register
ADCA11	FF 06C0h	Device A Address Counter Register		RQTR14	FF 0798h	Request Timeout Register
ADRA11	FF 06C4h	Device A Address Register		RQTCNT14	FF 079Ch	Request Timeout Counter Register
ADCB11	FF 06C8h	Device B Address Counter Register		DMACNT14	FF 07A0h	DMA Control Register
ADRB11	FF 06CCh	Device B Address Register		DMASTAT14	FF 07A4h	DMA Status Register
BLTC11	FF 06D0h	Block Length Counter Register		ADRA15	FF 07C4h	Device A Address Register
BLTR11	FF 06D4h	Block Length Register		ADCB15	FF 07C8h	Device B Address Counter Register
RQTR11	FF 06D8h	Request Timeout Register		ADRB15	FF 07CCh	Device B Address Register
RQTCNT11	FF 06DCh	Request Timeout Counter Register		BLTC15	FF 07D0h	Block Length Counter Register
DMACNT11	FF 06E0h	DMA Control Register		BLTR15	FF 07D4h	Block Length Register
DMASTAT11	FF 06E4h	DMA Status Register		RQTR15	FF 07D8h	Request Timeout Register
ADCA12	FF 0700h	Device A Address Counter Register		RQTCNT15	FF 07DCh	Request Timeout Counter Register
ADRA12	FF 0704h	Device A Address Register		DMACNT15	FF 07E0h	DMA Control Register
ADCA15	FF 07C0h	Device A Address Counter Register		DMASTAT15	FF 07E4h	DMA Status Register

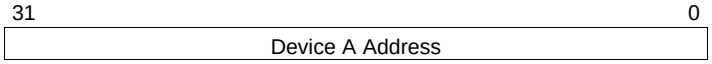
12.9.1 Device A Address Counter Register n (ADCAn)

The Device A Address Counter register is a 32-bit, read/ write register. It holds the current address of either the source data item or the destination location, depending on the state of the DIR bit in the DMACNTn register. The ADA bit of DMACNTn register controls whether to adjust the pointer in the ADCAn register by the step size specified in the INCA field of DMACNTn register.



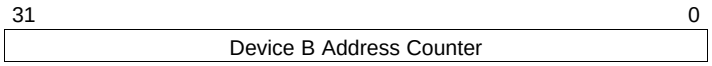
12.9.2 Device A Address Register n (ADRAn)

The Device A Address register is a 32-bit, read/write register. It holds the starting address of either the next source data block, or the next destination data area, according to the DIR bit in the DMACNTn register.



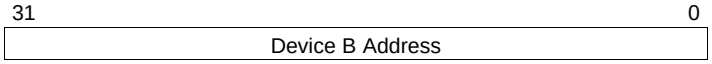
12.9.3 Device B Address Counter Register n (ADCBn)

The Device B Address Counter register is a 32-bit, read/ write register. It holds the address of either the source data item, or the destination location, according to the DIR bit in the DMACNTn register. The ADCBn register is updated after each transfer cycle by INCB field of the DMACNTn register according to ADB bit of the DMACNTn register.



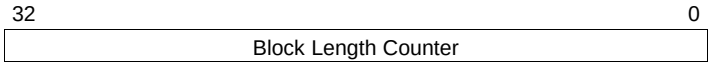
12.9.4 Device B Address Register n (ADRBn)

The Device B Address register is a 32-bit, read/write register. It holds the starting address of either the next source data block or the next destination data area, according to the DIR bit in the DMACNTn register.



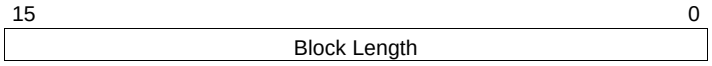
12.9.5 Block Length Counter Register n (BLTCn)

The Block Length Counter register is a 32-bit, read/write register. It holds the current number of DMA transfers to be executed in the current block. 0000 0000h is interpreted as 232-1 transfer cycles. BLTCn is decremented by one after each transfer cycle. A DMA transfer may consist of 1, 2, or 4 bytes, as selected by the TCS and WMODE bits in the DMACNTn register.



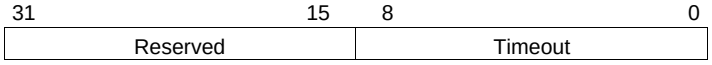
12.9.6 Block Length Register n (BLTRn)

The Block Length register is a 32-bit, read/write register. It holds the number of DMA transfers to be performed for the next block. 0000 0000h is interpreted as 232-1 transfer cycles. Writing this register automatically sets the DMASTAT.VLD bit.



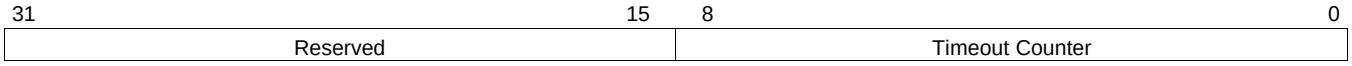
12.9.7 Request Timeout Register n (RQTRn)

The Request Timeout register is a 32-bit, read/write register. It holds the timeout time. After reset, the RQTRn registers are clear.



12.9.8 Request Timeout Counter Register n (RQTCNTn)

The Request Timeout Count register is a 32-bit, read-only register. It holds the current timeout count. A RQTCNTn register is loaded from its corresponding RQTRn register when its DMA request is asserted. The RQTCNTn register is decremented if the DMACNT.TOEN bit is set. After reset, the RQTCNTn registers are clear.



12.9.9 DMA Control Register n (DMACNTn)

The DMA Control register is a 32-bit, read/write register that controls the operation of DMA channel n. This register is initialized to 000C 0000h at reset. Reserved bits must be written with 0.

7	6	5	4	3	2	1	0
Reserved	OT	DIR	TCS	DTO	EOVR	ETC	CHEN

15	14	13	12	11	10	9	8
WMODE	INCB		ADB	INCA		ADA	SWRQ

7	6	5	4	3	2	1	0
Reserved	OT	DIR	TCS	DTO	EOVR	ETC	CHEN

31	12	11	0
Reserved		SRCRQ	

CHEN The Channel Enable bit must be set to enable any DMA operation on this channel. Writing a 1 to this bit starts a new DMA transfer even if it is currently a 1. If all DMACNTn.CHEN bits are clear, the DMA clock is disabled to reduce power.
0 – Channel disabled.
1 – Channel enabled.

ETC If the Enable Interrupt on Terminal Count bit is set, it enables an interrupt when the DMASTAT.TC bit is set.
0 – Interrupt disabled.
1 – Interrupt enabled

EOVR If the Enable Interrupt on OVR bit is set, it enables an interrupt when the DMASTAT.OVR bit is set.
0 – Interrupt disabled.
1 – Interrupt enabled.

ETO If the Enable Interrupt on Timeout bit is set, it enables an interrupt when the DMASTAT.TO bit is set.
0 – Interrupt disabled.
1 – Interrupt enabled.

TCS The Transfer Cycle Size bit specifies the number of bytes transferred in each DMA transfer cycle. This bit will be overridden if the WMODE bit is set.
0 – Byte transfers (8 bits per cycle).
1 – Word transfers (16 bits per cycle).

DIR The Transfer Direction bit specifies the direction of the transfer relative to Device A.
0 – Device A (addressed by the ADCAn register) is the source.
1 – Device A is the destination

OT The Operation Type bit specifies the operation mode of the DMA controller.
0 – Single-buffer mode or double-buffer mode enabled.
1 – Auto-Initialize mode enabled.

SWRQ The Software DMA Request bit is written with a 1 to initiate a software DMA request. Writing a 0 to this bit deactivates the software DMA request. The SWRQ bit must only be written when the peripheral DMA requests for the channel are disabled and the channel is inactive (DMASTAT.CHAC = 0).
0 – Software DMA request is inactive.
1 – Software DMA request is active.

ADA If the Device A Address Control bit is set, it enables updating the Device A address.
0 – ADCAn address unchanged.
1 – ADCAn address incremented or decremented, as controlled by the INCA field and WMODE and TCS bits.

INCA The Increment/Decrement ADCAn field together with the WMODE and TCS bits control the step size for the Device A address increment/decrement.

INCA	WMODE	TCS	Step Size
0	0	0	1
1	0	0	2
10	0	0	-1
11	0	0	-2
0	0	1	2
1	0	1	4
10	0	1	-2
11	0	1	-4
0X	1	X	4
1X	1	X	-4

If the Device B Address Control bit is set, it enables updating the Device B Address.
0 – ADCBn address unchanged.
1 – ADCBn address incremented or decremented, as controlled by the INCB field and WMODE and TCS bits.

INCB The Increment/Decrement ADCBn field specifies the step size for the Device B address increment/decrement.

INCB	WMODE	TCS	Step Size
0	0	0	1
1	0	0	2
10	0	0	-1
11	0	0	-2
0	0	1	2
1	0	1	4
10	0	1	-2
11	0	1	-4
0X	1	X	4
1X	1	X	-4

WMODE The 32-bit Word Mode bit controls whether the transfer cycle size is 32 bits. If set, it overrides the TCS bit.
0 – Transfer cycle size selected by the TCS bit.
1 – Transfer cycle size is 32 bits.

BBE The Burst Buffer Enable bit controls whether Single transfer type or data collection/burst transfer type is used. Only channels 0 and 1 have burst buffers. For the other channels, the BBE bit is reserved.
0 – Single transfer type.
1 – Data collection or burst transfer type.

PF The Peripheral Flow bit controls whether the DMAC or the peripheral is the flow controller.
0 – The DMAC is the flow controller. The DMA transfer is terminated when the BLTCn register counts down to zero.
1 – The peripheral is the flow controller.

HPROT The HPROT field must be 0011b (default value).

TOEN The Enable Timeout bit controls whether DMA request timeout monitoring is enabled.
0 – Timeout disabled.
1 – Timeout enabled.

SRCRQ The Source Request field specifies a peripheral register used as the source or destination for a DMA transfer.

12.9.10 DMA Status Register n (DMASTATn)

The DMA status register is a 32-bit register that holds the status information for the DMA channel. This register is cleared at reset. The reserved bits always return zero when read. The ERR, VLD, OVR, and TC bits are sticky (once set by the occurrence of the specific condition, they remain set until explicitly cleared by software). These bits can be cleared by writing 1 to their bit positions in the DMASTAT register to be cleared. Writing 0 to these bits has no effect.

31 16	15 14	13	12 8	7 6	5	4	3	2	1	0
Reserved	Reserved		BLV	Reserved	BNE	ERR	VLD	CHAC	OVR	TC

TC — The Terminal Count bit indicates whether the transfer was completed by a terminal count condition (BLTCn Register reached 0 and the PF bit was clear).
0 – Terminal count condition did not occur.
1 – Terminal count condition occurred.

OVR — The behavior of the Channel Overrun bit depends on the operation mode (single buffer, double buffer, or auto-initialize) of the DMA channel.

In double-buffered mode (DMACNTn.OT = 0): The OVR bit is set when the present transfer is completed (BLTCn = 0), but the parameters for the next transfer (address and block length) are not valid (DMASTATn.VLD = 0).

In auto-initialize mode (DMACNTn.OT = 1): The OVR bit is set when the present transfer is completed (BLTCn = 0), and the DMASTATn.TC bit is still set.

In single-buffer mode: Operates in the same way as double-buffer mode. In single-buffered mode, the DMASTATn.VLD bit should always be clear, so it will also be set when the DMASTATn.TC bit is set. Therefore, the OVR bit can be ignored in this mode.

CHAC — The Channel Active bit continuously indicates the active or inactive status of the channel, and therefore, it is read-only. Data written to the CHAC bit is ignored.
0 – Channel inactive.
1 – Indicates that the channel is active (CHEN bit in the CNTLn register is 1 and BLTCn > 0)

VLD — The Transfer Parameters Valid bit indicates whether the transfer parameters for the next block to be transferred are valid. Writing the BLTRn register automatically sets this bit. The bit is cleared in the following cases:

- The present transfer is completed and the ADRAn, ADRBn, and BLTRn registers have been loaded to the ADCAn, ADCBn, and BLTCn registers.
- Writing 1 to the VLD bit.

ERR — The Error bit indicates whether an error response has been detected during the last transfer. This may occur because the address was invalid or an incremental access goes out of bounds.
0 – Error condition did not occur.
1 – Error condition occurred.

BNE — The Buffer Not Empty bit indicates whether there is valid data in the burst buffer.
0 – Burst buffer is empty.
1 – Burst buffer holds valid data.

BLV — The Burst Buffer Level field indicates the number of valid bytes in the burst buffer.

TO — The Timeout bit indicates that a timeout event occurred (RQTCNTn reached its terminal count). This bit is cleared every time the DMA request for this channel is asserted.
0 – No timeout occurred since this bit was last cleared.
1 – A timeout occurred.

13 Interrupts

The Interrupt Control Unit (ICU) receives interrupt requests from internal and external sources and generates interrupts to the CPU. The highest-priority interrupt is the NonMaskable Interrupt (NMI), which is triggered by a falling edge received on the NMI input pin.

The maskable interrupts (IRQn) have a fixed, linear priority from IRQ0 through IRQ70, in which IRQ0 has the lowest priority and IRQ70 has the highest priority. IRQ0 is not implemented, so IRQ1 is the lowest priority maskable interrupt that may occur in normal operation.

13.1 Non-Maskable Interrupts

The Interrupt Control Unit (ICU) receives the external NMI input and generates the NMI signal driven to the CPU. The NMI input is an asynchronous input with Schmitt trigger characteristics and an internal synchronization circuit, therefore no external synchronizing circuit is needed. The NMI pin triggers an exception on its falling edge.

13.1.1 Non-Maskable Interrupt Processing

At reset, NMI interrupts are disabled and must remain disabled until software initializes the interrupt table, interrupt base register (INTBASE), and the interrupt mode. The external NMI interrupt is enabled by setting the EXNMI.ENLCK bit and will remain enabled until a reset occurs. Alternatively, the external NMI interrupt can be enabled by setting the EXNMI.EN bit and will remain enabled until an NMI interrupt or a reset occurs.

13.2 Maskable Interrupts

The IRQn interrupt channels are level-sensitive. Any edge sensitivity must be implemented at the interrupt source. The IRQ interrupts are enabled and disabled by the E and I bits in the PSR register. Both bits must be set to enable maskable interrupts. The EI and DI instructions are used to set (enable) and clear (disable) the E bit.

Each interrupt source can be individually enabled or disabled under software control through the IENR register and also through interrupt enable bits in the peripherals that request the interrupts.

13.2.1 Maskable Interrupt Processing

The IVECT register holds the interrupt vector number of the enabled and pending interrupt with the highest priority, mapped to the range 10h to 56h. IRQ0 is mapped to 10h, while IRQ70 is mapped to 56h. The CPU performs an interrupt acknowledge bus cycle on receiving a maskable interrupt request from the ICU. During the interrupt acknowledge cycle, a byte is read from address FF FE00h (the IVECT register). The byte is used as an index into the Dispatch Table to determine the address of the interrupt handler.

Because IRQ0 is not connected to any interrupt source, the interrupt vector number 10h should not be generated. However, an entry should be provided for this vector in the dispatch table that points to a default interrupt handler. One possible condition in which this vector number may occur is deassertion of an interrupt at its source before the interrupt controller generates an interrupt acknowledge cycle.

13.2.2 Maskable Interrupt Sources

Table 13-1 shows the interrupt sources assigned to the maskable interrupts.

Table 13-1. Maskable Interrupts Assignment

IRQn	DESCRIPTION	ASC CHANNEL
IRQ70	RTI (Timer 0)	
IRQ69	Real-Time Clock	
IRQ68	Bluetooth LLC 0	

Table 13-1. Maskable Interrupts Assignment (continued)

IRQn	DESCRIPTION	ASC CHANNEL
IRQ67	Bluetooth LLC 1	
IRQ66	Bluetooth LLC 2	
IRQ65	Bluetooth LLC 3	
IRQ64	Bluetooth LLC 4	
IRQ63	Bluetooth LLC 5	
IRQ62	Reserved	
IRQ61	USB Interrupt	
IRQ60	Teak DSP	
IRQ59	DMA Channel 0	
IRQ58	DMA Channel 1	
IRQ57	DMA Channel 2	
IRQ56	DMA Channel 3	
IRQ55	DMA Channel 4	
IRQ54	DMA Channel 5	
IRQ53	DMA Channel 6	
IRQ52	DMA Channel 7	
IRQ51	DMA Channel 8	
IRQ50	DMA Channel 9	
IRQ49	DMA Channel 10	
IRQ48	DMA Channel 11	
IRQ47	DMA Channel 12	
IRQ46	DMA Channel 13	
IRQ45	DMA Channel 14	
IRQ44	DMA Channel 15	
IRQ43	Reserved	
IRQ42	Reserved	
IRQ41	CAN0	
IRQ40	USART0 Rx	
IRQ39	USART0 Tx	
IRQ38	USART0 CTS	
IRQ37	TA0 (MFT0 Port A)	
IRQ36	TB0 (MFT0 Port B)	
IRQ35	TA1 (MFT1 Port A)	10
IRQ34	TB1 (MFT1 Port B)	9
IRQ33	VTU0A (VTU Interrupt Request 1)	
IRQ32	VTU0B (VTU Interrupt Request 2)	
IRQ31	VTU0C (VTU Interrupt Request 3)	
IRQ30	VTU0D (VTU Interrupt Request 4)	
IRQ29	Microwire/SPI 0 Rx/Tx	
IRQ28	Codec	8
IRQ27	Advanced Audio Interface	7
IRQ26	I^2S Interface	6
IRQ25	CVSD/PCM Converter 0	5
IRQ24	CVSD/PCM Converter 1	4
IRQ23	ACCESS.bus 0	
IRQ22	VTU1A (VTU Interrupt Request 1)	
IRQ21	VTU1B (VTU Interrupt Request 2)	

Table 13-1. Maskable Interrupts Assignment (continued)

IRQn	DESCRIPTION	ASC CHANNEL
IRQ20	VTU1C (VTU Interrupt Request 3)	
IRQ19	VTU1D (VTU Interrupt Request 4)	
IRQ18	CAN1	
IRQ17	UART1 Rx	
IRQ16	UART1 Tx	
IRQ15	UART2 Rx	
IRQ14	UART2 Tx	
IRQ13	UART3 Rx	3
IRQ12	UART3 Tx	2
IRQ11	ACCESS.bus 1	1
IRQ10	Microwire/SPI 1 Rx/Tx	0
IRQ9	ADC (Done)	
IRQ8	MIWU Interrupt 0	
IRQ7	MIWU Interrupt 1	
IRQ6	MIWU Interrupt 2	
IRQ5	MIWU Interrupt 3	
IRQ4	MIWU Interrupt 4	
IRQ3	MIWU Interrupt 5	
IRQ2	MIWU Interrupt 6	
IRQ1	MIWU Interrupt 7	
IRQ0	Reserved	

All reserved interrupt vectors must point to default or error interrupt handlers.

13.3 Interrupt Priority Groups

When more than one interrupt is asserted, two levels of priority are used to determine which interrupt is taken:

- *Group Priority*—each interrupt belongs to one of four priority groups numbered from 0 to 3, in which lower group numbers have higher priority.
- *Channel Priority*—each channel has a unique channel number, in which higher channel numbers have higher priority.

Group priority always takes precedence over channel priority. Channel priority is only used to resolve priority among interrupts in the same group.

Any interrupt may belong to any group, as programmed in the INTGPAR and INTGPBR registers. The corresponding bits in each register assign the interrupt channel to one of the groups, as shown in Table 13-2.

Table 13-2. Interrupt Priority Group Assignment

Priority Group for IRQn	INTGPBRn	INTGPARn
Group 0 (highest priority)	0	0
Group 1	0	1
Group 2	1	0
Group3 (lowest priority)	1	1

For example, if INTGPAR6:1 is loaded with 011101b and INTGPBR6:1 is loaded with 010001b, the priority groups for these interrupts are assigned as shown in Figure 13-1.

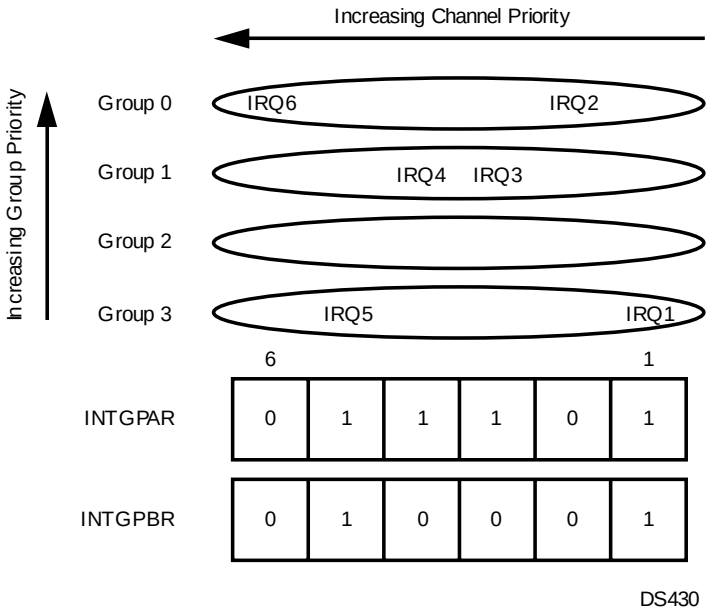


Figure 13-1. Interruupt Priority Groups

If the six IRQ1 through IRQ6 interrupts are asserted, then IRQ6 is taken because this interrupt belongs to the highest priority group (lowest group number), and it has the highest channel number within that group.

If only IRQ1 and IRQ5 are asserted, then IRQ5 is taken because this is the highest priority interrupt in group 3 and there are no asserted interrupts in other groups.

13.4 Nested Interrupts

Nested NMI interrupts are permanently enabled when the ENLCK bit in the EXNMI register is used to enable NMI interrupts. When nesting is not desired, the EN bit may be set to enable one occurrence of the NMI interrupt, after which NMI interrupts are disabled until the EN bit is set again.

Nested maskable interrupts are disabled by default, because the I bit in the PSR is automatically cleared when the interrupt is acknowledged. An interrupt handler can allow nested maskable interrupts by setting the I bit using the LPR instruction.

Nesting of specific maskable interrupts can be selectively enabled or disabled using the IENR register, before setting the I bit. Any number of levels of nested interrupts are allowed, limited only by the available memory for the interrupt stack.

13.5 Software Interrupts

Setting a bit in the SOFTR register requests the corresponding maskable interrupt. The request stays active until the bit is cleared by software or a device reset. Software interrupt requests are maskable in the IENR registers, and active interrupts (from either hardware or software sources) are indicated in the ISTR register.

13.6 Interrupt Controller Registers

Table 13-3 lists the interrupt controller registers.

Table 13-3. Interrupt Controller Registers

NAME	ADDRESS	DESCRIPTION
IVECT	FF FE00h	Interrupt Vector Register
NMISTAT	FF FE04h	Non-Maskable Interrupt Status Register
EXNMI	FF FE08h	External NMI Trap Control and Status Register
ISTR0	FF FE10h	Interrupt Status Register 0
ISTR1	FF FE14h	Interrupt Status Register 1
ISTR2	FF FE18h	Interrupt Status Register 2
IENR0	FF FE20h	Interrupt Enable and Mask Register 0
IENR1	FF FE24h	Interrupt Enable and Mask Register 1
IENR2	FF FE28h	Interrupt Enable and Mask Register 2
SOFTR0	FF FE40h	Software Interrupt Register 0
SOFTR1	FF FE44h	Software Interrupt Register 1
SOFTR2	FF FE48h	Software Interrupt Register 2
INTGPAR0	FF FE50h	Interrupt Priority Group A Register 0
INTGPAR1	FF FE58h	Interrupt Priority Group A Register 1
INTGPAR2	FF FE60h	Interrupt Priority Group A Register 2
INTGPBR0	FF FE54h	Interrupt Priority Group B Register 0
INTGPBR1	FF FE5Ch	Interrupt Priority Group B Register 1
INTGPBR2	FF FE64h	Interrupt Priority Group B Register 2
IDBG	FF FEFCh	Interrupt Debug Register

13.6.1 Interrupt Vector Register (IVCT)

The IVCT register is a 32-bit, read-only register which reports the encoded value of the highest priority maskable interrupt that is both asserted and enabled. The register is read by the CPU during an interrupt acknowledge bus cycle, and INTVECT is valid during that time. It may contain invalid data while INTVECT is updated. The register is initialized to 0000 0010h at reset.

31 … 8	7 … 0
Reserved	INTVECT

INTVECT The Interrupt Vector field indicates the highest priority interrupt which is both asserted and enabled. The valid range is from 10h to 56h.

13.6.2 Non-Maskable Interrupt Status Register (NMISTAT)

The NMISTAT register is a 32-bit, read-only register. It holds the status of the current pending Non-Maskable Interrupt (NMI) requests. On the CP3SP33, the external NMI input is the only source of NMI interrupts. The NMISTAT register is cleared on reset and each time its contents are read.

31 … 1	0
Reserved	EXT

EXT The External NMI request bit indicates whether an external non-maskable interrupt request has occurred. Refer to the description of the EXNMI register below for additional details.
0 – No external NMI request.
1 – External NMI request has occurred.

13.6.3 External NMI Trap Control and Status Register (EXNMI)

The EXNMI register is a 32-bit, read/write register. It indicates the current value of the NMI pin and controls the NMI interrupt trap generation based on a falling edge of the NMI pin. ENCLK, PIN, and EN are cleared on reset. When writing to this register, all reserved bits must be written with 0 for the device to function properly

31	3	2	1	0
Reserved		ENLCK	PIN	EN

EN The EXNMI trap enable bit is one of two bits that can be used to enable NMI interrupts. The bit is cleared by hardware at reset and whenever the NMI interrupt occurs (EXNMI.EXT set). It is intended for applications where the NMI input toggles frequently but nested NMI traps are not desired. For these applications, the EN bit needs to be re-enabled before exiting the trap handler. When used this way, the ENLCK bit should never be set. The EN bit can be set and cleared by software (software can set this bit only if EXNMI.EXT is cleared), and should only be set after the interrupt base register and the interrupt stack pointer have been set up.
0 – NMI interrupts not enabled by this bit (but may be enabled by the ENLCK bit).
1 – NMI interrupts enabled.

PIN The PIN bit indicates the state (non-inverted) on the NMI input pin. This bit is read-only, data written into it is ignored.
0 – NMI pin not asserted.
1 – NMI pin asserted.

ENLCK The EXNMI trap enable lock bit is used to permanently enable NMI interrupts. Only a device reset can clear the ENLCK bit. This allows the external NMI feature to be enabled after the interrupt base register and the interrupt stack pointer have been set up. When the ENLCK bit is set, the EN bit is ignored.
0 – NMI interrupts not enabled by this bit (but may be enabled by the EN bit).
1 – NMI interrupts enabled.

13.6.4 Interrupt Enable and Mask Register 0 (IENR0)

The IENR0 register is a 32-bit, read/write register which holds bits that individually enable and disable the maskable interrupt sources IRQ1 to IRQ31. The register is initialized to 0000 0000h at reset.

31	1	0
IEN0		Reserved

IEN0 Each Interrupt Enable bit enables or disables the corresponding interrupt channels IRQ1 through IRQ31. Because IRQ0 is not used, bit 0 is ignored.
0 – Interrupt is disabled.
1 – Interrupt is enabled.

13.6.5 Interrupt Enable and Mask Register 1 (IENR1)

The IENR1 register is a 32-bit, read/write register which holds bits that individually enable and disable the maskable interrupt sources IRQ63 to IRQ32. The register is initialized to 0000 0000h at reset.

31	0
IEN1	

IEN1 Each Interrupt Enable bit enables or disables the corresponding interrupt channels IRQ32 through IRQ63.
0 – Interrupt is disabled.
1 – Interrupt is enabled.

13.6.6 Interrupt Enable and Mask Register 2 (IENR2)

The IENR2 register is a 32-bit, read/write register which holds bits that individually enable and disable the maskable interrupt sources IRQ70 to IRQ64. Only bits 6:0 of this register are used. The register is initialized to 0000 0000h at reset.

31	0
IEN2	

IEN2 Each Interrupt Enable bit enables or disables the corresponding interrupt channels IRQ64 through IRQ70.
0 – Interrupt is disabled.
1 – Interrupt is enabled.

13.6.7 Interrupt Status Register 0 (ISTR0)

The ISTR0 register is a 32-bit, read-only register. It indicates which maskable interrupt inputs IRQ31 to IRQ1 are active. These bits are not affected by the state of the corresponding IENR0 bits. Because the IRQ0 interrupt is not used, bit 0 always reads as 0.

31	1	0
IST0		Reserved

IST0 The Interrupt Status bits indicate if a maskable interrupt source is signAling an interrupt request.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.8 Interrupt Status Register 1 (ISTR1)

The ISTR1 register is a 32-bit, read-only register. It indicates which maskable interrupt inputs IRQ63 to IRQ31 are active. These bits are not affected by the state of the corresponding IENR1 bits.

31	0
IST1	

IST1 The Interrupt Status bits indicate if a maskable interrupt source is signaling an interrupt request.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.9 Interrupt Status Register 2 (ISTR2)

The ISTR2 register is a 32-bit, read-only register. It indicates which maskable interrupt inputs IRQ70 to IRQ64 are active. These bits are not affected by the state of the corresponding IENR2 bits. Only bits 6:0 of this register are used.

31	0
IST2	

IST2 The Interrupt Status bits indicate if a maskable interrupt source is signaling an interrupt request.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.10 Software Interrupt Register 0 (SOFTR0)

The SOFTR0 register is a 32-bit, read/write register. Setting a bit in this register activates the corresponding maskable interrupt IRQ31 to IRQ1. Bit 0 of this register is reserved. The register is initialized to 0000 0000h at reset.

31	1	0
SOFT0		Reserved

SOFT0 The Software Interrupt bits activate the corresponding maskable interrupt when written with 1.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.11 Software Interrupt Register 1 (SOFTR1)

The SOFTR1 register is a 32-bit, read/write register. Setting a bit in this register activates the corresponding maskable interrupt IRQ63 to IRQ31. The register is initialized to 0000 0000h at reset.

31	0
SOFT1	

SOFT1 The Software Interrupt bits activate the corresponding maskable interrupt when written with 1.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.12 Software Interrupt Register 2 (SOFTR2)

The SOFTR2 register is a 32-bit, read/write register. Setting a bit in this register activates the corresponding maskable interrupt IRQ70 to IRQ64. Only bits 6:0 of this register are used. The register is initialized to 0000 0000h at reset.

31	0
SOFT2	

SOFT2 The Software Interrupt bits activate the corresponding maskable interrupt when written with 1.
0 – Interrupt is not active.
1 – Interrupt is active.

13.6.13 Interrupt Priority Group A Register 0 (INTGPAR0)

The INTGPAR0 register is a 32-bit, read/write register. Bits in this register specify the least significant bit for selecting the priority group of the corresponding interrupt IRQ31 to IRQ1. Bit 0 of this register is reserved. The register is initialized to 0000 0000h at reset.

31	1	0
INTGPA0		Reserved

INTGPA0 The Interrupt Priority Group A bits specify the least significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 2.
1 – Interrupt belongs to group 1 or 3.

13.6.14 Interrupt Priority Group A Register 1 (INTGPAR1)

The INTGPAR1 register is a 32-bit, read/write register. Bits in this register specify the least significant bit for selecting the priority group of the corresponding interrupt IRQ63 to IRQ31. The register is initialized to 0000 0000h at reset.

31	0
INTGPA1	

INTGPA1 The Interrupt Priority Group A bits specify the least significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 2.
1 – Interrupt belongs to group 1 or 3.

13.6.15 Interrupt Priority Group A Register 2 (INTGPAR2)

The INTGPAR2 register is a 32-bit, read/write register. Bits in this register specify the least significant bit for selecting the priority group of the corresponding interrupt IRQ70 to IRQ64. Only bits 6:0 of this register are used. The register is initialized to 0000 0000h at reset.

31	0
INTGPA2	

INTGPA2 The Interrupt Priority Group A bits specify the least significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 2.
1 – Interrupt belongs to group 1 or 3.

13.6.16 Interrupt Priority Group B Register 0 (INTGPBR0)

The INTGPBR0 register is a 32-bit, read/write register. Bits in this register specify the most significant bit for selecting the priority group of the corresponding interrupt IRQ31 to IRQ1. Bit 0 of this register is reserved. The register is initialized to 0000 0000h at reset.

31	1	0
INTGPB0		Reserved

INTGPB0 The Interrupt Priority Group B bits specify the most significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 1.
1 – Interrupt belongs to group 2 or 3.

13.6.17 Interrupt Priority Group B Register 1 (INTGPBR1)

The INTGPBR1 register is a 32-bit, read/write register. Bits in this register specify the most significant bit for selecting the priority group of the corresponding interrupt IRQ63 to IRQ31. The register is initialized to 0000 0000h at reset.

15	1
INTGPB1	

INTGPB1 The Interrupt Priority Group B bits specify the most significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 1.
1 – Interrupt belongs to group 2 or 3.

13.6.18 Interrupt Priority Group B Register 2 (INTGPBR2)

The INTGPBR2 register is a 32-bit, read/write register. Bits in this register specify the most significant bit for selecting the priority group of the corresponding interrupt IRQ70 to IRQ64. Only bits 6:0 of this register are used. The register is initialized to 0000 0000h at reset.

31	0
INTGPB2	

INTGPB2 The Interrupt Priority Group B bits specify the most significant bit for selecting one of four interrupt priority groups.
0 – Interrupt belongs to group 0 or 1.
1 – Interrupt belongs to group 2 or 3.

13.6.19 Interrupt Debug Register (IDBG)

The IDBG register is a 32-bit, read-only register. Fields in this register indicate the interrupt number of the highest priority currently asserted interrupt and the interrupt number returned to the CPU during the last interrupt acknowledge cycle. (These may differ if a higher-priority interrupt is asserted after the interrupt acknowledge cycle.) The register is initialized to 0000 0000h at reset.

31 16	15 8	7 0
Reserved	IRQVECT	INTVECT

INTVECT The INTVECT field indicates the interrupt vector number of the highest priority currently asserted interrupt.

IRQVECT The IRQVECT field indicates the interrupt vector number returned to the CPU during the last interrupt acknowledge cycle.

13.7 Usage Notes

The recommended initialization sequence is:

1. Initialize the INTBASE register of the CPU.
2. Prepare the interrupt routines of the relevant interrupts.
3. Set up the interrupt conditions in the peripherals.
4. Set the relevant bits in the interrupt enable/mask registers (IENRn).
5. Set the I bit in the PSR.

Clearing an interrupt request before it is serviced can cause a spurious interrupt, (i.e., the CPU detects an interrupt not reflected by IVECT). Software must clear interrupt requests only after interrupts are disabled.

Clearing any of the interrupt enable bits should be performed while the I bit in the PSR register is clear.

14 Clock Generation

The clock circuitry includes oscillators for generating a 12MHz Main Clock and an optional 32.768-kHz Slow Clock from external crystal networks. Alternatively, either clock may be replaced by an external clock source.

Two identical PLLs are available to provide higher clock frequencies (up to 96 MHz) using Main Clock as a reference frequency. The PLLs also may be used to synchronize with an external clock driven on the I2S interface (I2SCLK).

Most of the functional blocks of the device operate from one of the global clocks generated by this module:

- Main Clock—12-MHz clock generated by an on-chip oscillator or received from an external clock source.
- PLL1 Clock—clock synthesized by PLL1 from Main Clock or the external I2SCLK clock input.
- PLL2 Clock—clock synthesized by PLL2 from Main Clock or the external I2SCLK clock input.
- HCLK Clock—clock used by the CPU and devices on the CPU core AHB bus. HCLK Clock is generated by a prescaler from Main Clock, PLL1 Clock, or PLL2 Clock. Available prescale factors are 1 to 2048 in increments of 1/2 clock period. In Power Save mode, HCLK Clock is driven by Slow Clock.
- PCLK Clock—clock used by devices on the peripheral APB bus. PCLK Clock is generated by a prescaler from HCLK Clock. Available prescale factors are 1, 2, and 4.

- Slow Clock—32.768 kHz clock used in low-power modes. Slow Clock is generated by an on-chip oscillator, received from an external clock source or generated by a prescaler from Main Clock. Available prescale factors are 1 to 8192 in increments of 1/2 clock period.

Figure 14-1 is block diagram of the global clock generation logic.

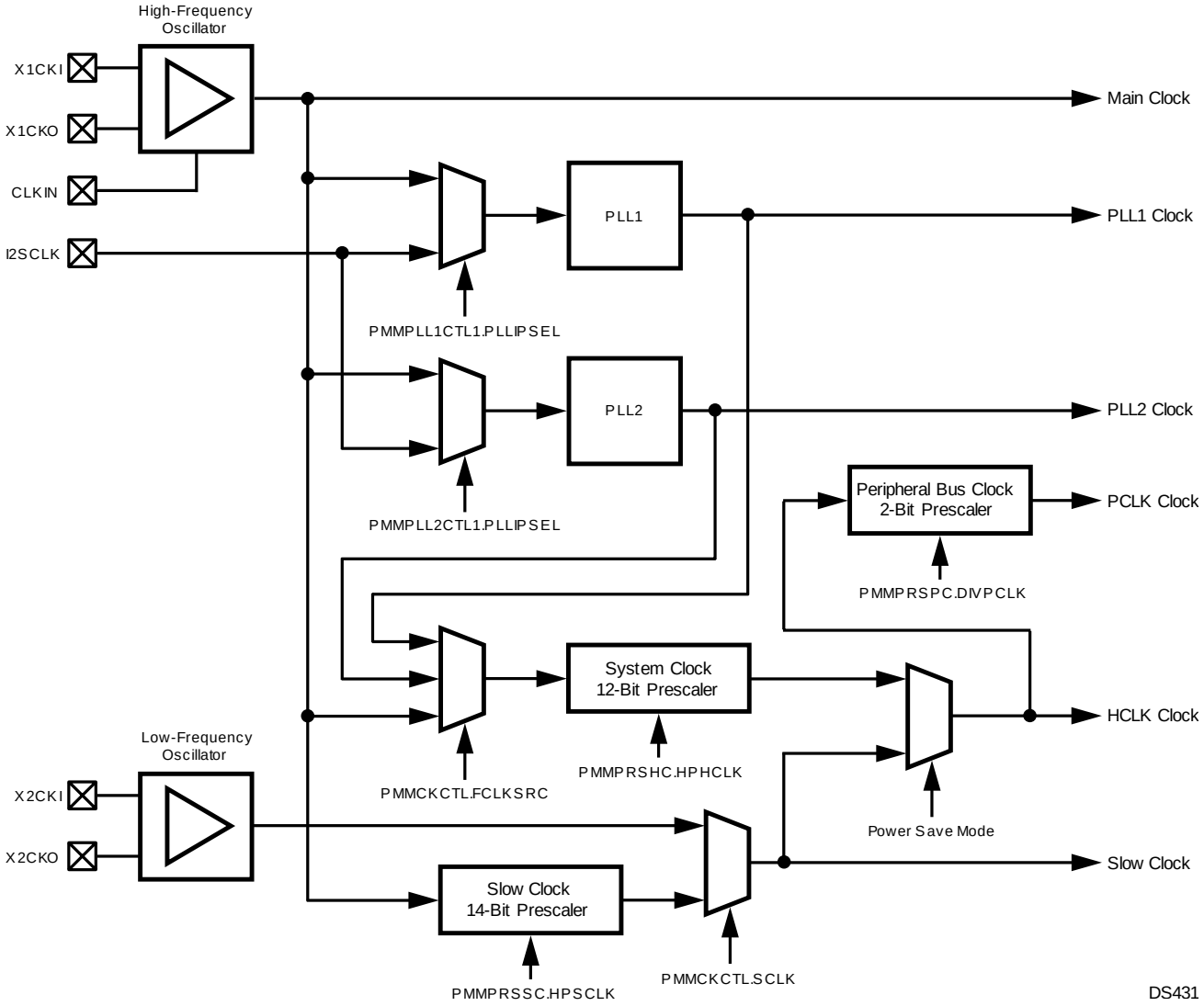


Figure 14-1. Global Clock Generation

14.1 Extrnal Crystal Networks

An external crystal network is connected to the X1CKI and X1CKO pins to generate the Main Clock, unless an external clock signal is driven on the CLKIN pin. A similar external crystal network may be used at pins X2CKI and X2CKO for the Slow Clock. If an external crystal network is not used for the Slow Clock, the Slow Clock is generated through a prescaler from Main Clock.

The crystal network you choose may require external components different from the ones specified in this datasheet. In this case, consult with National's engineers for the component specifications

The crystals and other oscillator components must be placed close to the X1CKI/X1CKO and X2CKI/X2CKO device input pins to keep the trace lengths to an absolute minimum.

Figure 14-2 shows the external crystal network for the X1CKI and X1CKO pins. The crystal must be an AT-cut type. Figure 14-3 shows the external crystal network for the X2CKI and X2CKO pins. The crystal may be an N-cut or XY-bar type. Table 14-1 shows the component specifications for the 12-MHz crystal network, and Table 14-2 shows the component specifications for the 32.768 kHz crystal network.

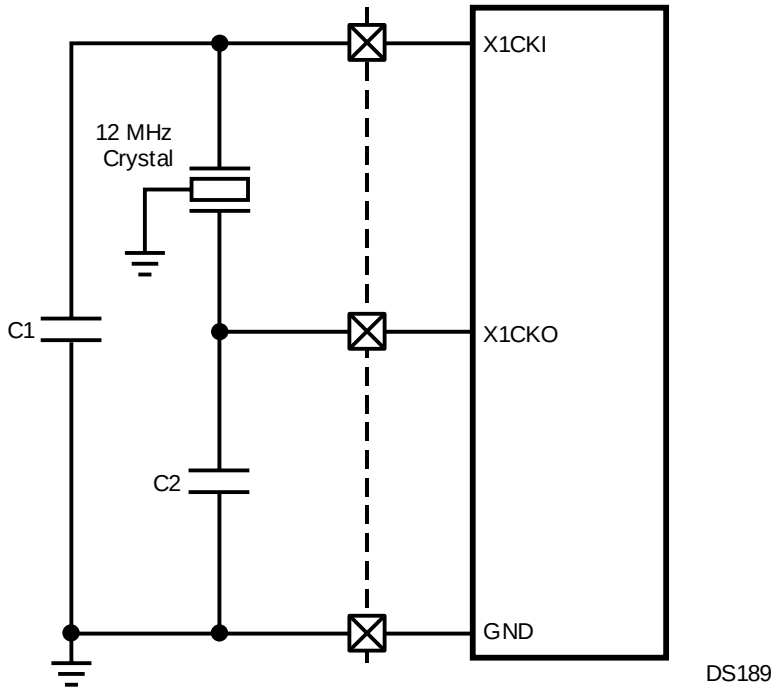


Figure 14-2. 12 MHz Oscillator Crystal Network

Table 14-1. 12 MHz Oscillator Component Values

SYMBOL	COMPONENT	PARAMETER	MIN	TYP	MAX
f	Crystal	Resonant Frequency		12 MHz	
Rm	Crystal	Motional Resistance (ESR)			50 Ω
C0	Crystal	Case Capacitance			7 pF
C1, C2	Capacitors	External Capacitance		22 pF	
CL	Capacitors	Load Capacitance		11 pF	

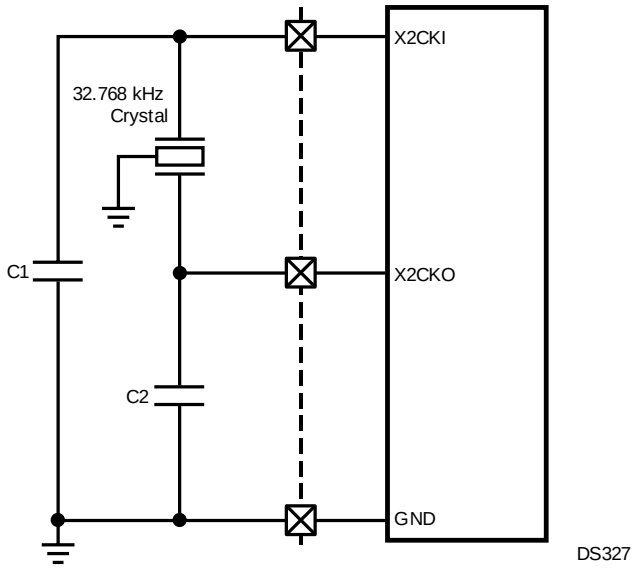


Figure 14-3. 32.768 kHz Oscillator Crystal Network

Table 14-2. 32.768 kHz Oscillator Component Values

SYMBOL	COMPONENT	PARAMETER	MIN	TYP	MAX
f	Crystal	Resonant Frequency		32.768 kHz	
Cm	Crystal	Motional Resistance		1.7 pF	
Rm	Crystal	Motional Resistance (ESR)			65 kΩ
C0	Crystal	Shunt Capacitance			1.3 pF
Q	Crystal	Q Factor		40000	
CL	Capacitors	Load Capacitance		25 pF	

14.2 Main Clock

Main Clock is generated by the 12-MHz high-frequency oscillator or driven by an external signal (typically the LMX5251 or LMX5252 Bluetooth radio chip). It can be stopped by the Power Management Module to reduce power consumption during periods of reduced activity. When the Main Clock is started or restarted, a 14-bit timer generates a start-up delay to determine when the high-frequency oscillator is stable.

An external clock can be driven on the X1CKI input, but it must not exceed 1.8V. Alternatively, a 3.3V external clock can be driven on the CLKIN pin, in which case the X1CKI input must be tied low. The frequency must be a multiple of 12 MHz, up to 96 MHz. (If a frequency other than 12 MHz is chosen, the timing of the external bus must be compatible with the external memory device used to boot up the system.) If the Power Management Module indicates the high-frequency oscillator should be stopped, the on-chip Main Clock signal is stopped even if it was driven by an external clock signal on CLKIN or X1CKI that continues toggling.

14.3 Slow Clock

Slow Clock is necessary for operating the device in reduced power modes and to provide a clock source for modules such as the Timing and Watchdog Module.

The low-frequency oscillator may be used to generate Slow Clock in a manner similar to the Main Clock. It can be stopped by the Power Management Module to reduce power consumption during periods of reduced activity. When the Slow Clock is started or restarted, a 6-bit timer generates a start-up delay to determine when the low-frequency oscillator is stable.

For systems that do not require a reduced power consumption mode, the external crystal network may be omitted for the low-frequency oscillator. In this case, Slow Clock is synthesized by dividing the Main Clock by a prescaler factor. The prescaler consists of a 14-bit prescaler. This allows a choice of clock divisors ranging from 1 to 8192 in increments of 1/2 Main Clock period. The resulting Slow Clock frequency must not exceed 100 kHz.

A software-controlled multiplexer selects either the prescaled Main Clock or the 32.768 kHz oscillator as the Slow Clock. At reset, the prescaled Main Clock is selected, ensuring that the Slow Clock is always present initially. Selection of the 32.768 kHz oscillator as the Slow Clock source disables the clock prescaler, and it may allow the high-frequency oscillator to be turned off in some low-power modes, for minimum power consumption and radiated emissions. The high-frequency oscillator cannot be disabled when the prescaler is selected as the source for Slow Clock.

An external clock may be driven on the X2CKI input. It must not exceed 1.8V.

14.4 PLL Clocks

The CP3SP33 has two identical PLLs for generating clocks. These are two purposes for providing PLL clock generators:

- *Higher Frequency*—the PLLs can generate clock frequencies up to 96 MHz using the 12 MHz Main Clock as a reference. Frequencies above 12 MHz are needed to run the CPU and DSP at their maximum speed. Also, the USB peripheral requires a 60 MHz clock.
- *Synchronization with External Devices*—some devices (such as external codecs) require operation at a precise frequency which is not an integer multiple of a standard CP3SP33 clock frequency. By synchronizing a PLL with the external device, it can be interfaced directly without treating its I/O as asynchronous signals.

After reset, the PLLs are powered down. Software can program and enable the PLLs to start them running. Each PLL has a programmable start-up counter to indicate when the PLL has become stable. By default, the start-up counter will indicate the PLL has become stable after 32K cycles of the PLL Clock.

To enable the PLL after reset or after PLL power down, perform the following sequence:

1. Set the divider ratios in the PMMPLLnMDIV, PMMPLLnNDIV, PMMPLLnNMOD, and PMMPLLnPDIV registers to acceptable values for the desired output frequency. If the number of PLL cycles to allow for stabilization should be different from the default value (32K cycles), set the PLL Startup Counter register (PMMPLLnSTUP).
2. Power up the PLL by setting the PMMPLLnCTL1.PLLEN field to 01b.
3. Wait until the PMMPLLnCTL2.PLLCLKSTB bit has become set, which indicates the PLL is stable.
4. The PMMCKCTL.FCLKSRC field can then be loaded with 00b to select PLL1 Clock as the source for the HCLK Clock. It can be loaded with 01b to select PLL2 Clock as the source. The switch only occurs if the selected PLL is stable. To switch from PLL1 Clock to PLL2 Clock, Main Clock must be selected as the clock source during an intermediate step (direct switching between the two PLL Clocks is not allowed).

   If any of the divisors (PMMPLLnMDIV, PMMPLLnNDIV, PMMPLLnNMOD, and PMMPLLnPDIV) are to be changed, the HCLK Clock source must be switched to Main Clock (if the PLL was driving HCLK Clock), and the PLL must be powered down (for example, by clearing the PMMPLLnCTL1.PLLEN field). Then, the procedure above may be performed to restart the PLL after loading new values in the registers.

14.4.1 PLL Programming

The PLLs have the structure shown in Figure 14-4. The PMMPLLnMDIV, PMMPLLnNDIV, PMMPLLnNMOD, and PMMPLLnPDIV registers specify the M, N, and P divisors which control the operation of the PLL. The N divisor has an integer part specified in the PMMPLLnNDIV register and a fractional part specified in the PMMPLLnNMOD register.

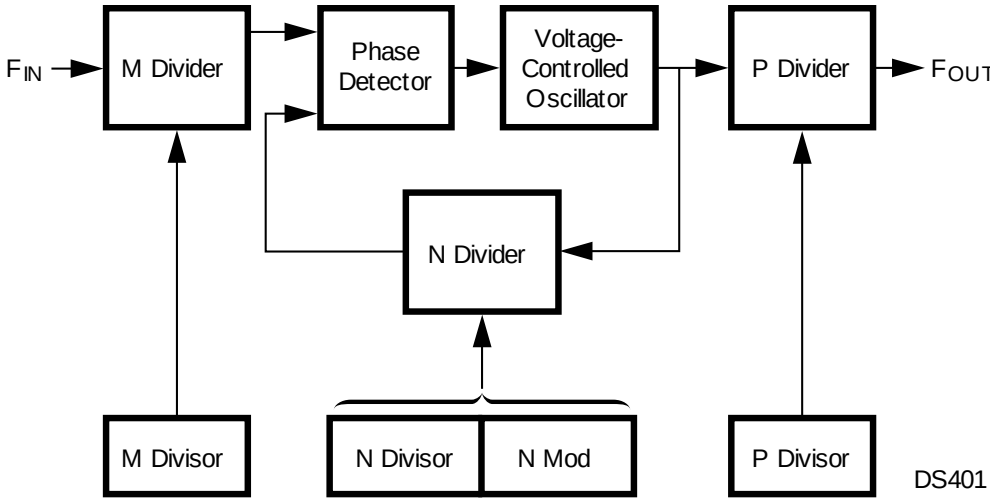


Figure 14-4. PLL Module

The reference frequency F_{REF} is derived from the input frequency F_{IN}, as described by the following equation:

$$c_n = \frac{\Delta t}{2\pi} \int_0^{(2\pi)/(\Delta t)} H_D(w) \cos \frac{2\pi n w}{w_s} dw \quad (1)$$

The voltage-controlled oscillator output frequency F_{VCO} is derived from the input frequency F_{IN}, as described by the following equation:

$$F_{VCO} = F_{IN} \times \left(\frac{NMOD}{32 \times MDIV} \right) \quad (2)$$

F_{REF} must be between 2 and 4 MHz. F_{VCO} must not exceed 153 MHz.

The output frequency FO_{UT} is derived from the input frequency FIN, as described by the following equation:

$$F_{OUT} = F_{IN} \times \left(\frac{NDIV + \frac{NMOD}{32}}{MDIV \times PDIV} \right) \quad (3)$$

It is possible for software to load divisors that result in unstable or marginally stable PLL operation. Therefore, only divisors recommended in Table 14-3 and Table 14-4 should be used. If any other frequencies are desired, contact Texas Instruments.

Table 14-3. Recommended HCLK Clock Divisors

F_{IN} (MHz)	M Div.	N Div.	N Mod.	P Div.	F_{OUT} (MHz)
12	4	32	0	4	24
12	6	64	0	4	32
12	4	48	0	4	36
12	4	32	0	2	48
12	3	30	0	2	60
12	6	64	0	2	64
12	4	48	0	2	72
12	4	30	0	1	90
12	3	24	0	1	96

The PLLs are designed to drive the telematics codec and other devices with audio sample rates (F_s) of 44.1 kHz and 48 kHz (as well as 32 kHz, 24 kHz, 22.05 kHz, etc.) and 125× or 250× oversampling. Therefore, the two common clock requirements are 11.025 MHz (44.1 kHz × 250) and 12.000 MHz (48 kHz × 250). These frequencies can be obtained from any common input clock rate between 10 MHz and 20 MHz as shown in Table 14-4.

Table 14-4. Recommended Audio Clock Divisors

SOURCE	F_S (kHz)	F_{IN} (MHz)	M Div.	N Div.	N Mod.	P Div.
System	44.1	11	5	55	4	11
System	44.1	11.2896	4	46	2	12
System	44.1	12	5	55	3	12
System	44.1	13	5	55	4	13
System	44.1	14.4	4	30	20	10

Table 14-4. Recommended Audio Clock Divisors (continued)

SOURCE	F_S (kHz)	F_{IN} (MHz)	M Div.	N Div.	N Mod.	P Div.
System	44.1	16.2	5	30	20	9
System	44.1	16.8	4	26	8	10
System	44.1	19.2	4	26	8	10
System	44.1	19.44	6	30	20	9
System	44.1	19.8	6	36	24	11
System	48	11	4	48	0	11
System	48	12	4	40	0	10
System	48	12.288	6	46	28	8
System	48	13	13	96	0	8
System	48	14.4	6	40	0	8
System	48	16.2	6	40	0	9
System	48	16.8	7	40	0	8
System	48	19.2	6	30	0	8
System	48	19.44	9	50	0	9
System	48	19.8	6	40	0	11
I^2S	48	1.536	1	78	4	10
I^2S	44.1	1.4112	1	78	4	10
BT/USB	11.025	12	4	36	24	40
BT/USB	32	12	4	32	0	12
GSM	22.05	13	21	102	16	21

14.5 HCLK Clock

HCLK Clock drives the CPU and the modules on the CPU core AHB bus. It is generated by a 12-bit prescaler from Main Clock, PLL1 Clock, or PLL2 Clock. The prescaler allows setting the HCLK Clock period to any length between 1 and 2048 periods of the clock source, in increments of 1/2 period. The default setting for the HCLK Clock period is 16 clock source periods.

Software may configure Slow Clock to drive HCLK Clock during Power Down and Idle modes.

14.6 PCLK Clock

PCLK Clock drives the modules on the peripheral APB bus. It is generated by a 2-bit prescaler from HCLK Clock, for a division factor of 1, 2, or 4. The default is 1. The maximum PCLK Clock frequency is 48 MHz.

14.7 Auxiliary Clocks

Independent clocks are available for certain peripherals that may require clock frequencies different from those provided as global clocks. An auxiliary clock also may be used for peripherals that require independence from the clock-switching mechanism applied to HCLK Clock and PCLK Clock during low-power modes. Eight auxiliary clock generators are provided for these peripherals. Some peripherals have clock multiplexers to allow selection among several available clock sources.

Each auxiliary clock generator has an input multiplexer for selecting an input clock source:

- Main Clock
- PLL1 Clock
- PLL2 Clock

A 12-bit prescaler provides a prescale factors from 1 to 2048 clock periods of the input clock in increments of 1/2 clock period. For example, if the selected clock source is 12 MHz and the desired auxiliary clock frequency is 1 MHz, the divisor should be 2 × 12 or 24. However, the value loaded in the PMMAUXnPRSC prescaler register is biased by 1, so the actual value loaded into the register will be 23 (decimal).

$$\text{PMMAUXnPRSC} = \left(\frac{2 \times F_{MAIN}}{F_{DESIRED}} \right) - 1 \quad (4)$$

Zero is an undefined value, so the shortest period that an auxiliary clock may have is 2 half-clocks, which results in the same clock rate as the prescaler input clock.

By default, the auxiliary clocks are disabled following reset, except the DSP clock (Auxiliary Clock 7).

Each auxiliary clock is available for one or more peripherals, as listed in Table 14-5.

Table 14-5. Auxiliary Clocks Available to Peripherals

AUXILIARY CLOCK	PERIPHERAL	COMMENTS
1	Bluetooth, ADC, AAI	Bluetooth LLC requires an accurate 12 MHz clock, in addition to a variable Auxiliary Clock 1.
2	CVSD/PCM Converter 0	Supports a 2-MHz mode in which data is processed at a fixed rate. Also supports a free-running mode in which data is processed on demand. The free-running mode does not require a specific clock frequency. The CVSD modules may operate from independent clocks.
3	CVSD/PCM Converter 1	
4	Telematics Codec ADC 1, ADC 2, and Stereo DACs	Requires an accurate clock frequency, typically 125× or 128× the PCM frame rate.
5	I^2S Interface	
6	USB	Requires an accurate 60 MHz clock. Also requires a HCLK Clock >30 MHz.
7	DSP	Up to 96 MHz
8	AAI	Requires an accurate frequency to match the bit clock and frame sync required by the external codec.

Figure 14-5 is block diagram of the auxiliary clock generation logic.

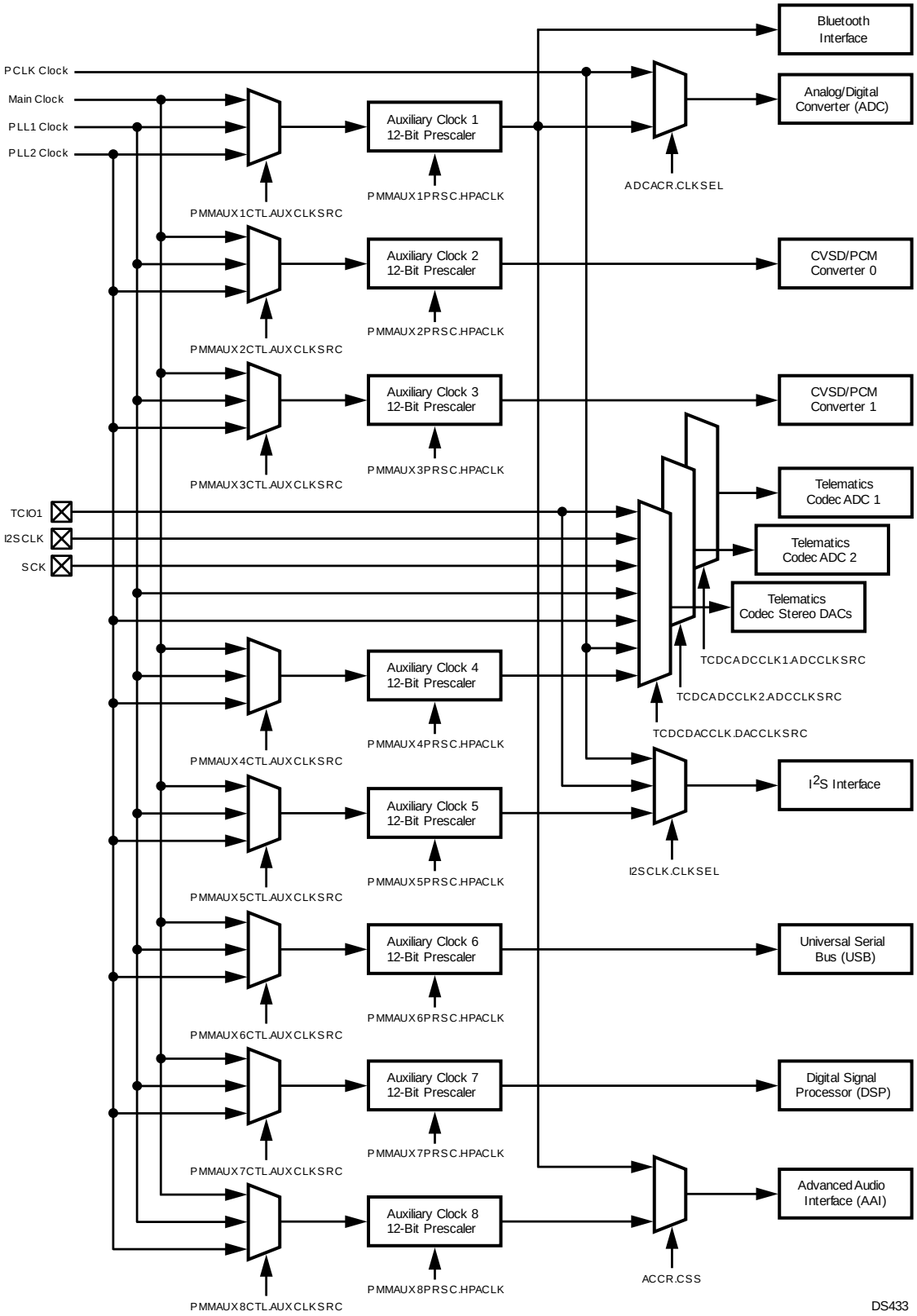


Figure 14-5. Auxiliary Clock Generators

14.8 Clock Generation Registers

Table 14-6 lists the clock generation registers.

Table 14-6. Clock and Reset Registers

NAME	ADDRESS	DESCRIPTION
PMMCKCTL	FF A400h	Clock and Reset Control Register
PMMSR	FF A408h	Clock Status Register
PMMPRSHC	FF A40Ch	HCLK Clock Prescaler Register
PMMPRSPC	FF A410h	PCLK Clock Prescaler Register
PMMPRSSC	FF A414h	Low Frequency Clock Prescaler Register
PMMPLL1CTL1	FF A420h	PLL1 Control Register 1
PMMPLL1CTL2	FF A424h	PLL1 Control Register 2
PMMPLL1MDIV	FF A428h	PLL1 M Divider Prescaler Register
PMMPLL1NDIV	FF A42Ch	PLL1 N Divider Prescaler Register
PMMPLL1NMOD	FF A434h	PLL1 N Divider Prescaler Register
PMMPLL1PDIV	FF A430h	PLL1 P Divider Prescaler Register
PMMPLL1STUP	FF A438h	PLL1 Start-Up Counter Register
PMMPLL2CTL1	FF A440h	PLL2 Control Register 1
PMMPLL2CTL2	FF A444h	PLL2 Control Register 2
PMMPLL2MDIV	FF A448h	PLL2 M Divider Prescaler Register
PMMPLL2NDIV	FF A44Ch	PLL2 N Divider Prescaler Register
PMMPLL2NMOD	FF A454h	PLL2 N Divider Prescaler Register
PMMPLL2PDIV	FF A450h	PLL2 P Divider Prescaler Register
PMMPLL2STUP	FF A458h	PLL2 Start-Up Counter Register
PMMAUX1CTL	FF A500h	Auxiliary Clock 1 Control Register
PMMAUX1PRSC	FF A504h	Auxiliary Clock 1 Prescaler Register
PMMAUX2CTL	FF A510h	Auxiliary Clock 2 Control Register
PMMAUX2PRSC	FF A514h	Auxiliary Clock 2 Prescaler Register
PMMAUX3CTL	FF A520h	Auxiliary Clock 3 Control Register
PMMAUX3PRSC	FF A524h	Auxiliary Clock 3 Prescaler Register
PMMAUX4CTL	FF A530h	Auxiliary Clock 4 Control Register
PMMAUX4PRSC	FF A534h	Auxiliary Clock 4 Prescaler Register
PMMAUX5CTL	FF A540h	Auxiliary Clock 5 Control Register
PMMAUX5PRSC	FF A544h	Auxiliary Clock 5 Prescaler Register
PMMAUX6CTL	FF A550h	Auxiliary Clock 6 Control Register
PMMAUX6PRSC	FF A554h	Auxiliary Clock 6 Prescaler Register
PMMAUX7CTL	FF A560h	Auxiliary Clock 7 Control Register
PMMAUX7PRSC	FF A564h	Auxiliary Clock 7 Prescaler Register
PMMAUX8CTL	FF A570h	Auxiliary Clock 8 Control Register
PMMAUX8PRSC	FF A574h	Auxiliary Clock 8 Prescaler Register

14.8.1 Clock and Reset Control Register (PMMCKCTL)

The PMMCKCTL register is a byte-wide, read/write register that controls the clock selection and contains the power-on reset status bit. At reset, the PMMCKCTL register is initialized as described below:

7 – 5	4 – 3	2	1	0
Reserved	FCLKSRC	SCLK	Reserved	POR

POR — The Power-On-Reset bit is set when a power-on condition has been detected. This bit can only be cleared by software, not set. Writing a 1 to this bit will be ignored, and the previous value of the bit will be unchanged.
0 – Software cleared this bit.
1 – Software has not cleared his bit since the last reset.

SCLK — The Slow Clock Select bit controls the clock source used for the Slow Clock. If set while the low-frequency oscillator is not stable, this bit will read as 1 but the clock source will not switch until the low-frequency oscillator is stable. This bit cannot be cleared while the high-frequency oscillator is not stable. At reset, this bit is cleared.
0 – Slow Clock driven by prescaled Main Clock.
1 – Request asserted to drive Slow Clock from the 32.768 kHz oscillator.

FCLKSRC — The Fast Clock Source field selects the clock source used to generate HCLK Clock. After reset, the Main Clock is selected. Requesting a switch to a PLL Clock while the corresponding PMMPLLn.CTL2.PLLCLKSTB bit is clear (PLL Clock not stable) will be stalled until the PLL Clock is stable. When switching between PLL Clocks, there must be an intermediate step in which Main Clock is selected. At reset, this field is initialized to 11b.
00 – PLL1.
01 – PLL2.
10 – Reserved.
11 – Main Clock.

14.8.2 Clock Status Register (PMMSR)

The PMMSR register is a byte-wide read-only register that holds the two bits which indicate whether the high-frequency oscillator and the low-frequency oscillator are stable.

7 – 2	1	0
Reserved	MCLKSTB	SCLKSTB

SCLKSTB — The Slow Clock Stable bit indicates whether the low-frequency oscillator is producing a stable clock.
0 – Low-frequency oscillator is unstable, disabled, or not oscillating.
1 – Low-frequency oscillator is available.

MCLKSTB — The Main Clock Stable bit indicates whether the high-frequency oscillator is producing a stable clock.
0 – High-frequency oscillator is unstable, disabled, or not oscillating.
1 – High-frequency oscillator is available.

14.8.3 HCLK Clock Prescaler Register (PMMPRSHC)

The PMMPRSHC register is a 16-bit read/write register that holds the 12-bit prescaler available to generate HCLK Clock. The register is initialized to 001Fh at reset.

15 – 12	11 – 0
Reserved	HPHCLK

HPHCLK The Half Periods High Frequency Clock field holds the divisor (expressed in half-periods of the input clock source) for the prescaler used to generate HCLK Clock. The clock source is divided by ((HPHCLK + 1) ÷ 2) to obtain the HCLK Clock. The field is biased by 1, so the default value of this field specifies 32 half-periods, which is a divisor of 16. Zero is an undefined value for this field.

14.8.4 PCLK Clock Prescaler Register (PMMPRSPC)

The PMMPRSPC register is a byte-wide, read/write register that holds the divisor for generating PCLK Clock from HCLK Clock. The maximum PCLK Clock frequency is 48 MHz. The register is initialized to 00h at reset.

7	2	1	0
Reserved		DIVPCLK	

DIVPCLK The Divisor PCLK Clock field has four defined values:
00b – ÷1
01b – ÷2
10b – ÷4
11b – Reserved.

14.8.5 Slow Clock Prescaler Register (PMMPRSSC)

The PMMPRSSC register is a 16-bit, read/write register that holds the 14-bit prescaler available to generate Slow Clock from Main Clock. The register is initialized to 02DBh at reset.

31	14	13	0
Reserved		HPSCLK	

HPSCLK The Half Periods Slow Clock field holds the divisor (expressed in half-periods of the input clock source) for the prescaler used to generate Slow Clock. The Main Clock is divided by ((HPSCLK + 1) ÷ 2) to obtain the Slow Clock. At reset, the HPSCLK register is initialized to 02DBh, which generates a Slow Clock rate of 32.786885 kHz. This is about 0.5% faster than a Slow Clock generated from an external 32.768 kHz crystal network. Zero is an undefined value for this field.

14.8.6 PLLn Control Register 1 (PMMPLLnCTL1)

The PMMPLLnCNT1 registers are 16-bit, read/write registers that hold control fields for the corresponding PLL. The registers are initialized to 00h at reset.

15	5	4	3	2	1	0
Reserved		PLLIPSEL		PLLDCOE		PLLEN

PLLEN The PLL Enable field controls whether the PLL is enabled or disabled. Before software can power down a PLL used to generate HCLK Clock or an Auxiliary Clock, the clocks must be switched to another clock source.
00 – PLL disabled.
01 – PLL enabled.
10 – Reserved.
11 – Reserved.

PLLDCOE The PLL Direct Clock Output Enable bit controls the corresponding PLL output. The PLL Clock output is driven only when this bit is set and the PLL Clock is stable.
0 – PLL Clock output disabled.
1 – PLL Clock is enabled when stable.

PLLIPSEL The PLL Input Select field selects the input clock source for the corresponding PLL.
00 – Main Clock.
01 – I2SCLK pin.
10 – Reserved.
11 – Reserved

14.8.7 PLLn Control Register 2 (PMMPLLnCTL2)

The PMMPLLnCNT2 registers are 16-bit, read/write registers that hold control fields for the corresponding PLL. The registers are initialized to 00h at reset.

15	3	2	1	0
Reserved		HCCPLL	DPLLC	PLLCLKSTB

PLLCLKSTB The PLL Clock Stable bit indicates when the corresponding PLL Clock is stable.
0 – PLL Clock not stable.
1 – PLL Clock is stable.

DPLLC The Disable PLL Clock Enable bit controls whether the corresponding PLL is disabled when entering the Power Save or Idle modes. This bit is cleared when a hardware wake-up event occurs.
0 – PLL may be enabled in Power Save and Idle modes.
1 – PLL is always disabled in Power Save and Idle modes.

HCCPLL The Hardware Clock Control PLL bit enables the hardware clock control mechanism, See Section 16.5 for more information.
0 – Hardware clock control disabled.
1 – Hardware clock control enabled.

14.8.8 PLLn M Divider Register (PMMPLLnMDIV)

The PMMPLLnMDIV registers are byte-wide, read/write registers that hold the 8-bit M divisor for the corresponding PLL. The registers are initialized to 00h at reset.

7	0
MDIV	

MDIV The M Divisor field specifies an integer divisor. See Section 14.4.1 for more information.

14.8.9 PLLn N Divider Register (PMMPLLnNDIV)

The PMMPLLnNDIV registers are byte-wide, read/write registers that hold the 8-bit integer part of the N divisor. The registers are initialized to 00h at reset.

7	0
NDIV	

NDIV The N Divisor field specifies the integer part of the N divisor. See Section 14.4.1 for more information.

14.8.10 PLLn N Mod Register (PMMPLLnNMOD)

The PMMPLLnNMOD registers are 16-bit, read/write registers that hold the 5-bit fractional part of the N divisor for the corresponding PLL and a control field for the level of dithering in effect. The registers are initialized to 00h at reset.

15	14	13	8	7	0
NMOD_DITH		Reserved		NMOD	

NMOD The N Mod field specifies the fractional part of the N divisor. See Section 14.4.1 PLL Programming for more information.

NMOD_DITH The N Mod Dithering field specifies the level of dithering to be applied to the PLL Clock to suppress tone artifacts that may occur in some audio applications. The default level is sufficient for a wide range of applications.
00 – Medium (default).
01 – Low.
10 – High.
11 – No dithering.

14.8.11 PLLn P Divider Register (PMMPLLnPDIV)

The PMMPLLnPDIV registers are byte-wide, read/write registers that hold the 8-bit P divisor for the corresponding PLL. The register is initialized to 00h at reset.

7	0
PDIV	

PDIV The P Divisor field specifies an integer divisor. See Section 14.4.1 PLL Programming for more information.

14.8.12 PLLn Start-Up Counter Register (PMMPLLnSTUP)

The PMMPLLnSTUP registers are 16-bit, read/write registers that specify a 10-bit integer number of clock periods that must occur before the corresponding PLL Clock is considered stable. The PLL Clock output is not driven unless the PLL is stable and the PMMPLLnCTL1.PLLDCOE bit is set. The registers are initialized to 007Fh at reset.

15	10	9	0
Reserved		CLKCNT	

CLKCNT The CLKCNT field specifies the number of PLL Clock cycles which must occur before the PLL is considered stable. The number of cycles is (256 × (CLKCNT + 1)), therefore the default is 32,768 cycles.

14.8.13 Auxiliary Clock n Control Register (PMMAUXnCTL)

The PMMAUXnCTL registers are byte-wide, read/write registers that control the Auxiliary Clock generators. The registers are initialized to 06h at reset, except PMMAUX7CTL which is initialized to 07h.

7	3	2	1	0
Reserved		AUXCLKSRC		AUXCLKEN

AUXCLKEN The Auxiliary Clock Enable bit enables the corresponding Auxiliary Clock output.
0 – PLL Clock output disabled.
1 – PLL Clock is enabled when stable.

AUXCLKSRC The Auxiliary Clock Source field selects the input clock source for the corresponding Auxiliary Clock generator. When switching from one PLL to another, there must be an intermediate step in which Main Clock is selected.
00 – PLL1 Clock.
01 – PLL2 Clock.
10 – Reserved.
11 – Main Clock.

14.8.14 Auxiliary Clock n Prescaler Register (PMMAUXnPRSC)

The PMMAUXnPRSC registers are 16-bit read/write registers that hold the 12-bit clock divisors (expressed in halfclocks) for the prescalers used to generate the corresponding Auxiliary Clocks from Main Clock, PLL1 Clock, or PLL2 Clock (as selected by the PMMAUXnCTL.AUXCLKSRC field). The registers are initialized to 00FFh at reset, except PMMAUX7PRSC which is initialized to 0001h.

15	12	11	0
Reserved		HPACLK	

HPACLK The Half Periods Per Auxiliary Clock field specifies the number of half-clocks used for generating the Auxiliary Clock. The clock source is divided by ((HPACLK + 1) ÷ 2) to obtain the auxiliary clock. Zero is an undefined value for this field.

15 Reset

There are five sources of reset:

- *Power-On Reset*—on-chip power-on detector and timer.
- *External Reset*—assertion of the RESET input.
- *Software Reset*—enabled by writing special code sequences to the SWRESET register.
- *Timing and Watchdog Module (TWM)*—overflow of the watchdog timer
- *SDI Reset*—reset from the Serial Debug Interface (SDI)

15.1 Power-On Reset

The on-chip Power On Reset (POR) circuit generates a positive edge on the internal reset signal when VCC rises above V_{TRIP} (typically 1.38V). The VCC rise time from 0V to V_{TRIP} must not exceed the maximum t_{TRIP} specification. After triggering the POR circuit, the rise time from V_{TRIP} to a stable VCC must not exceed the maximum t_D specification. After reset has occurred, the POR is specified to rearm when VCC falls below 400 mV (although it may rearm at a higher voltage, up to 600 mV), and it will be retriggered when VCC again rises above V_{TRIP}. There must be a delay of at least 950 µs before retriggering the POR.

VCC ramp-up must reach its nominal level before or simultaneous with IOVCC ramp-up, otherwise spikes may be driven on GPIO-capable pins.

15.2 Reset Input Timing

The CP3SP33 has specific timing requirements that must be met to prevent improper program behavior, such as corruption of an external flash memory programming operation in progress when the reset is received. This timing sequence shown in Figure 15-1.

All reset circuits must ensure that this timing sequence is always maintained during power-up and power-down. The design of the power supply also affects how this sequence is implemented.

The power-up sequence is:

1. The $\overline{\text{RESET}}$ pin must be held low until *both* IOVCC and VCC have reached the minimum levels specified in the DC Characteristics section. VCC must reach its nominal level at or before IOVCC.
2. After both of these supply voltage rails have met this condition, then the $\overline{\text{RESET}}$ pin may be driven high. At power-up an internal 14-bit counter is set to 3FFFh and begins counting down to 0 after the crystal oscillator becomes stable. When this counter reaches 0, the onchip $\overline{\text{RESET}}$ signal is driven high unless the external $\overline{\text{RESET}}$ pin is still being held low. This prevents the CP3SP33 from coming out of reset with an unstable clock source.

The power-down sequence is:

1. The $\overline{\text{RESET}}$ pin must be driven low as soon as *either* the IOVCC or VCC voltage rail reaches the minimum levels specified in the DC Characteristics.

2. The $\overline{\text{RESET}}$ pin must then be held low until the Main Clock is stopped. The Main Clock will decay with the same profile as IOVCC.

Meeting the power-down reset conditions ensures that software will not be executed at voltage levels that may cause incorrect program execution or corruption of the flash memories. This situation must be avoided because the Main Clock decays with the IOVCC supply rather than stopping immediately when IOVCC falls below the minimum specified level.

The external reset circuits presented in the following sections provide varying levels of additional fault tolerance and expandability and are presented as possible examples of solutions to be used with the CP3SP33. It is important to note, however, that any design for the reset circuit and power supply must meet the timing requirements shown in Figure 15-1.

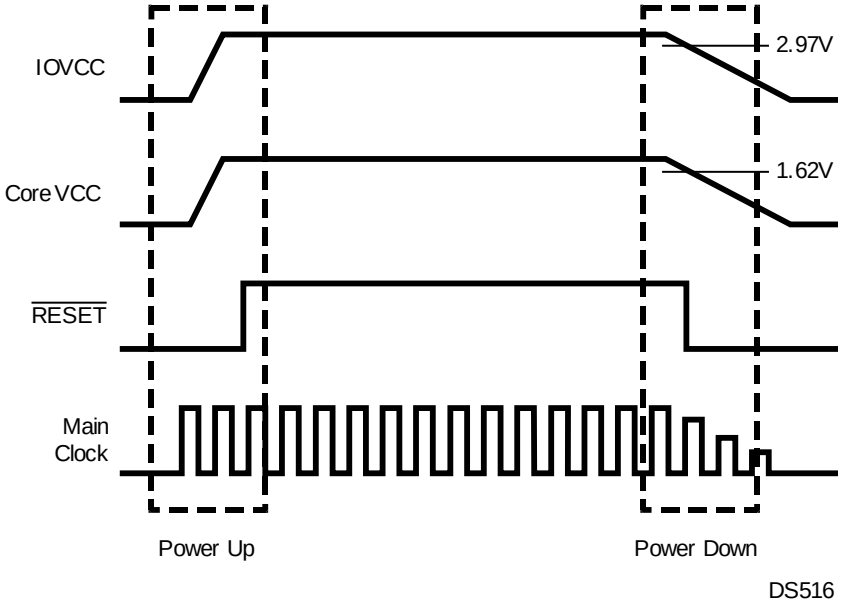


Figure 15-1. Power-On Reset Timing

15.2.1 Simple External Reset

A simple external reset circuit with brown-out and glitch protection based on the LM809 3-Pin Microprocessor Reset Circuit is shown in Figure 15-2. The LM809 produces a 240-ms logic low reset pulse when the power supply rises above a threshold voltage. Various reset thresholds are available for the LM809, however the options for 2.93V and 3.08V are most suitable for a CP3SP33 device operating from an IO-VCC at 3.0V to 3.3V.

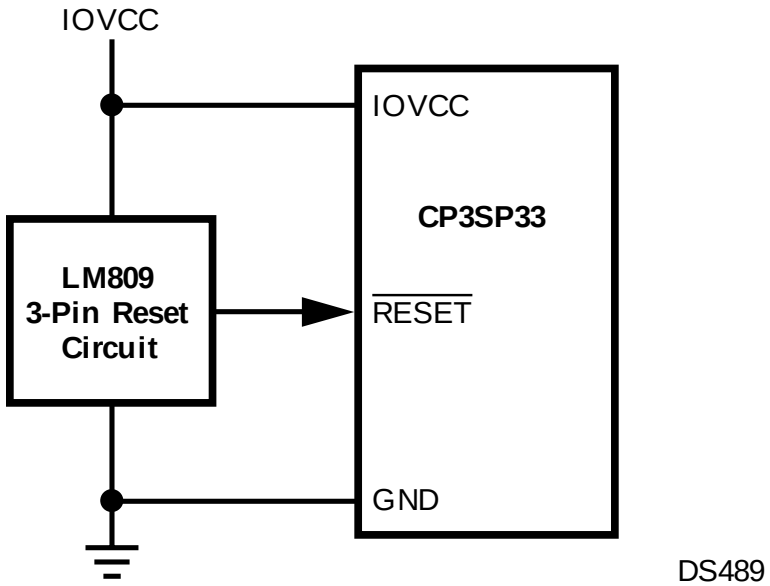


Figure 15-2. Simple External Reset

15.2.2 Manual and SDI External Reset

An external reset circuit based on the LM3724 5-Pin Microprocessor Reset Circuit is shown in Figure 15-3. The LM3724 produces a 190-ms logic low reset pulse when the power supply rises above a threshold voltage or a manual reset button is pressed. Various reset thresholds are available for the LM3724, however the option for 3.08V is most suitable for a CP3SP33 device operating from an IOVCC at 3.3V.

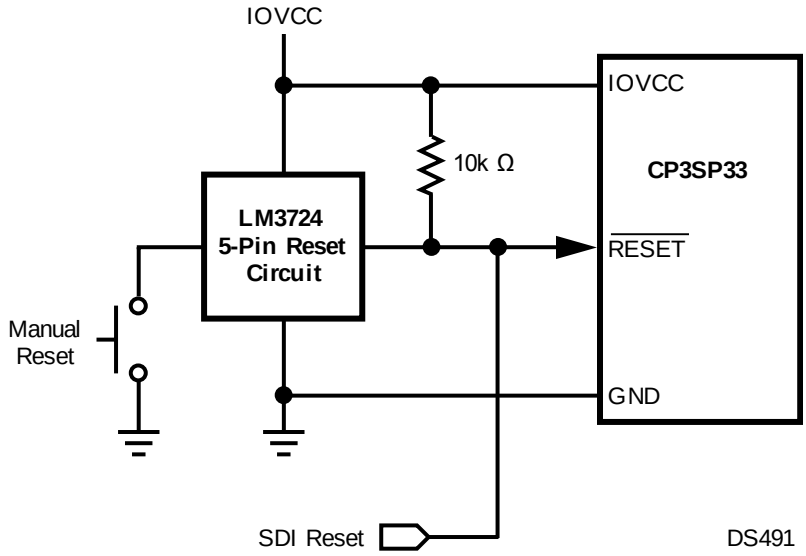


Figure 15-3. Manual and SDI External Reset

The LM3724 provides a debounced input for a manual pushbutton reset switch. It also has an open-drain output which can be used for implementing a wire-OR connection with a reset signal from a serial debug interface. This circuit is typical of a design to be used in a development or evaluation environment, however it is a good recommendation for all general CP3SP33 designs. If an SDI interface is not implemented, an LM3722 with active pullup may be used.

15.2.3 Fault-Tolerant External Reset

An external reset circuit based on the LM3710 Microprocessor Supervisory Circuit is shown in Figure 15-4. It provides a high level of fault tolerance in that it provides the ability to monitor both the VCC supply for the core logic and the IO-VCC supply. It also provides a low-voltage indication for the IOVCC supply and an external watchdog timer.

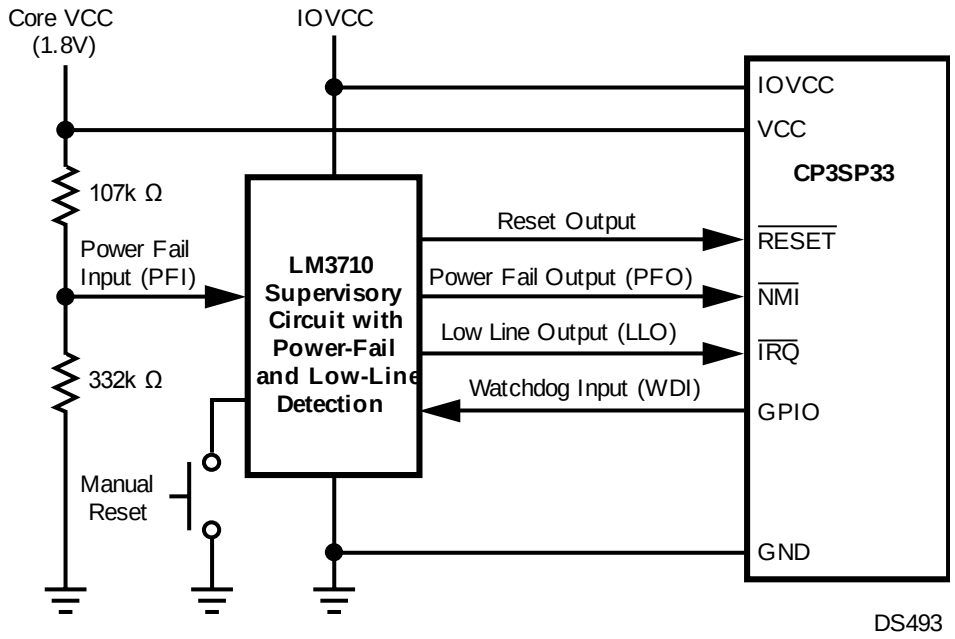


Figure 15-4. Fault-Tolerant External Reset

The signals shown in Figure 15-4 are:

- *Core VCC*—the 1.8V power supply rail for the core logic.
- *IOVCC*—the 3.0–3.3V power supply rail for the I/O logic.
- *Watchdog Input (WDI)*—this signal is asserted by the CP3SP33 at regular intervals to indicate normal operation. A general-purpose I/O (GPIO) port may be used to provide this signal. If the internal watchdog timer in the CP3SP33 is used, then the LM3704 Microprocessor Supervisory Circuit can provide the same features as the LM3710 but without the watchdog timer.
- *$\overline{\text{RESET}}$*—an active-low reset signal to the CP3SP33. The LM3710 is available in versions with active pullup or an open-drain RESET output.
- *Power-Fail Input (PFI)*—this is a voltage level derived from the Core VCC power supply rail through a simple resistor divider network.
- *Power-Fail Output (PFO)*—this signal is asserted when the voltage on PFI falls below 1.225V. PFO is connected to the non-maskable interrupt (NMI) input on the CP3SP33. A system shutdown routine can then be invoked by the NMI handler.
- *Low Line Output (LLO)*—this signal is asserted when the main IOVCC level fails below a warning threshold voltage but remains above a reset detection threshold. This signal may be routed to the $\overline{\text{NMI}}$ input on the CP3SP33 or to a separate interrupt input.

These additional status and feedback mechanisms allow the CP3SP33 to recover from software hangs or perform system shutdown functions before being placed into reset.

The standard reset threshold for the LM3710 is 3.08V with other options for different watchdog timeout and reset timeouts. The selection of these values are much more application-specific. The combination of a watchdog timeout period of 1600 ms and a reset period of 200 ms is a reasonable starting point.

16 Power Management Module

The Power Management Module (PMM) improves the efficiency of the CP3SP33 by changing the operating mode (and therefore the power consumption) according to the required level of device activity. The device implements four power modes:

- *Active Mode*—CPU and core bus peripherals operate from HCLK Clock, which is generated by a prescaler from Main Clock, PLL1 Clock, or PLL2 Clock. Either PLL may be powered down if its output is not used to gene ate HCLK Clock or any Auxiliary Clock.
- *Power-Save Mode*—CPU and core bus peripherals operate from Slow Clock. Any of the PLLs and high-frequency oscillator may be powered down, if its output is not used to generate Slow Clock or any Auxiliary Clock.
- *Idle Mode*—CPU and most or all peripherals are powered down. A watchdog timeout (from a timer running on Slow Clock), external reset, or MIWU input can break out of this mode.
- *Halt Mode*—only an external reset or MIWU input can break out of this mode.

Table 16-1 summarizes the differences between power modes: the state of the high-frequency oscillator (on or off), PLLs, HCLK Clock source (clock used by CPU and core bus peripherals), and clock source used by the Timing and Watchdog Module (TWM).

Table 16-1. Power Mode Operating Summary

MODE	HIGH-FREQUENCY OSCILLATOR	PLLs	HCLK CLOCK SOURCE	TWM CLOCK SOURCE
Active	On	On or Off	Prescaler	Slow Clock
Power Save	On or Off	On or Off	Slow Clock	Slow Clock
Idle	On or Off	On or Off	None	Slow Clock
Halt	Off	Off	None	None

The low-frequency oscillator continues to operate in all four modes and power must be provided continuously to the device power supply pins. In Halt mode, however, Slow Clock does not toggle, and as a result, the Timing and Watchdog Module does not operate. For the Power Save and Idle modes, the high-frequency oscillator can be turned on or off under software control, if the low-frequency oscillator is used to drive Slow Clock.

Software can configure either PLL to be powered down in Active, Power Save, and Idle mode, as long as it is not driving HCLK Clock or an Auxiliary Clock generator.

Table 16-2 shows the clock sources used by the CP3SP33 device modules and their behavior in each power mode.

Table 16-2. Module Activity Summary

MODULE	POWER MODE				CLOCK SOURCE(s)
	ACTIVE	POWER SAVE	IDLE	HALT	
CPU	On	On/Off	Off	Off	HCLK Clock
MIWU	On	On	On	Active	PCLK Clock
PMM	On	On	On	Active	Slow Clock
TWM	On	On	On	Off	Slow Clock
Bluetooth	On/Off	On/Off	On/Off	Off	Aux Clk 1
A/D Converter	On/Off	On/Off	On/Off	Off(1)	Aux Clk 1, PCLK Clock
CVSD/PCM 0	On/Off	On/Off	On/Off	Off	Aux Clk 2
CVSD/PCM 1	On/Off	On/Off	On/Off	Off	Aux Clk 3
Codec	On/Off	On/Off	On/Off	Off	Aux Clk 1, Aux Clk 4, PCLK Clock, PLL1 Clk, PLL2 Clk, I2SCLK, AAI SCK, TCIO1

(1) The Analog/Digital Converter (ADC) module is not automatically disabled by entering Halt mode, however its clock is stopped so no conversions may be performed in Halt mode. For maximum power savings, software must disable the ADC module before entering Halt mode.

Table 16-2. Module Activity Summary (continued)

MODULE	POWER MODE				CLOCK SOURCE(s)
	ACTIVE	POWER SAVE	IDLE	HALT	
I2S Interface	On/Off	On/Off	On/Off	Off	Aux Clk 5, PCLK Clock, TCIO1
USB	On/Off	On/Off	On/Off	Off	Aux Clk 6
DSP	On/Off	On/Off	On/Off	Off	Aux Clk 7
AAI	On/Off	On/Off	On/Off	Off	Aux Clk 1, Aux Clk 8
All Others	On/Off	On/Off	Off	Off	HCLK Clock, PCLK Clock

A module shown as On/Off in Table 16-2 may be enabled or disabled by software. A module shown as Active continues to operate even while its clock is suspended, which allows wake-up events to be processed during Idle and Halt modes.

16.1 Active Mode

In Active mode, the high-frequency oscillator is active and generates the 12-MHz Main Clock. If the PLL Clocks are not needed, the PLLs may remain powered off. Most devices on the CPU core bus are driven by HCLK Clock, and most APB bus devices are driven by PCLK Clock.

When entering Active mode from a mode in which the high-frequency oscillator, PLLs, or low-frequency clock were powered down, the mode switch will be stalled until the high-frequency oscillator, any enabled PLL, and the low-frequency oscillator (if enabled) are producing stable clocks.

The activity of peripheral modules is controlled by their enable bits. Power consumption can be reduced in Active mode by selectively disabling modules and by executing the WAIT instruction. When the WAIT instruction is executed, the CPU stops executing new instructions until it receives an interrupt signal.

16.2 Power Save Mode

In Power Save mode, Slow Clock is used as the HCLK Clock which drives the CPU and core bus modules. Power Save mode is intended for applications in which a low level of processing is required during low-power standby, for example if software needs to detect events for which no hardware wake-up signal is available.

If Slow Clock is driven by the 32.768 kHz oscillator and no on-chip module currently requires the 12-MHz Main Clock, software can disable the high-frequency oscillator to further reduce power consumption. The auxiliary clocks can be turned off under software control before switching to a reduced power mode, or they may remain active as long as they have an active clock source.

If a PLL Clock is not used to generate HCLK Clock or an auxiliary clock, the corresponding PLL can be powered down. Control over whether Main Clock and the PLLs are running can be controlled by the Bluetooth controller through the Hardware Clock Control function (described in Section), if enabled.

In Power Save mode, some modules are disabled or their operation is restricted. Other modules, including the CPU, continue to function normally, but operate at a reduced clock rate. See the module descriptions for details of each module's activity in Power Save mode.

16.3 Idle Mode

In Idle mode, the HCLK Clock and PCLK Clock are disabled and therefore the clock is stopped to most modules of the device. Idle mode is intended for applications in which no processing is required during low-power standby, but the capability is needed to break out of the mode due to activity on the Bluetooth interface or expiration of the watchdog timer. Idle mode can only be entered from Power Save mode.

The PLLs and the high-frequency oscillator may be disabled as controlled by register bits. The low-frequency oscillator remains active. The Power Management Module (PMM) and the Timing and Watchdog Module (TWM) continue to operate from the Slow Clock. The auxiliary clocks can be turned off under software control before switching to a low-power mode, or they remain active as long as they have an active clock source (Main Clock, PLL1 Clock, or PLL2 Clock, as selected by the PMMCKCTL.FCLKSRC bit). Alternatively, generation of Main Clock and the PLL Clocks can be controlled by the Bluetooth controller through the Hardware Clock Control function (described in Section 16.5), if enabled.

16.4 Halt Mode

In Halt mode, all the device clocks, including the Main Clock, PLL1 Clock, PLL2 Clock, HCLK Clock, PCLK Clock, and Slow Clock, are disabled. Halt mode is intended for applications in which an external wake-up signal can be used to power-up the system through the RESET input or an MIWU input.

The high-frequency oscillator and PLLs are turned off. This is the only mode in which the low-frequency oscillator may be disabled, however its circuitry is optimized to ensure lowest possible power consumption. This mode allows the device to reach the absolute minimum power consumption without losing its state (memory, registers, etc.).

16.5 Hardware Clock Control

The Hardware Clock Control (HCC) mechanism gives the Bluetooth Lower Link Controller (LLC) individual control over the high-frequency oscillator and the PLLs. The Bluetooth LLC can enter a Sleep mode for a specified number of low-frequency clock cycles. While the Bluetooth LLC is in Sleep mode and the CP3SP33 is in Power Save or Idle mode, the HCC mechanism may be used to control whether the high-frequency oscillator or PLLs are powered.

The PMMSTCTL.HCCM bit enables control by the Bluetooth LLC over the high-frequency oscillator, and the PMMPLLnCTL2.HCCPLL bits enable control over the corresponding PLLs. If the HCCM bit is set, then control is enabled over both the high-frequency oscillator and the PLLs, and the HCCPLL bits are ignored. (The PLLs cannot be enabled while the high-frequency oscillator is disabled.)

Altogether, three mechanisms control whether the high-frequency oscillator is active, and four mechanisms control whether the PLLs are active:

- *HCC bits*—the HCC bits (PMMSTCTL.HCCM and PMMPLLnCTL2.HCCPLL) allow the Bluetooth LLC to disable the high-frequency oscillator and PLLs during Power Save and Idle modes when the Bluetooth LLC goes into Sleep mode. The HCC bits are automatically cleared when Active mode is entered.
- *Disable bits*—the disable bits (PMMSTCTL.DMC and PMMPLLnCTL2.DPLLC) force the high-frequency oscillator and PLLs to be disabled when entering the Power Save and Idle modes. When set, these bits override the HCC mechanism. The DMC and DPLLC bits are automatically cleared by a hardware wake-up event.
- *Power Management Mode*—Halt mode disables the high-frequency oscillator and PLLs, without regard to any register bits. Active mode enables the high-frequency oscillator, again without regard to any bits. Entering Active mode will restart any PLLs enabled to run.
- *PLL enable fields*—The PMMPLLnCTL1.PLLEN fields enable the corresponding PLLs to run.

16.6 Switching Between Power Modes

Switching from a higher to a lower power consumption mode is performed by writing an appropriate value to the Power Management Control Register (PMMSTCTL). Switching from a lower power consumption mode to the Active mode is usually triggered by a hardware event. Figure 16-1 shows the four power modes and the events that trigger a transition from one mode to another.

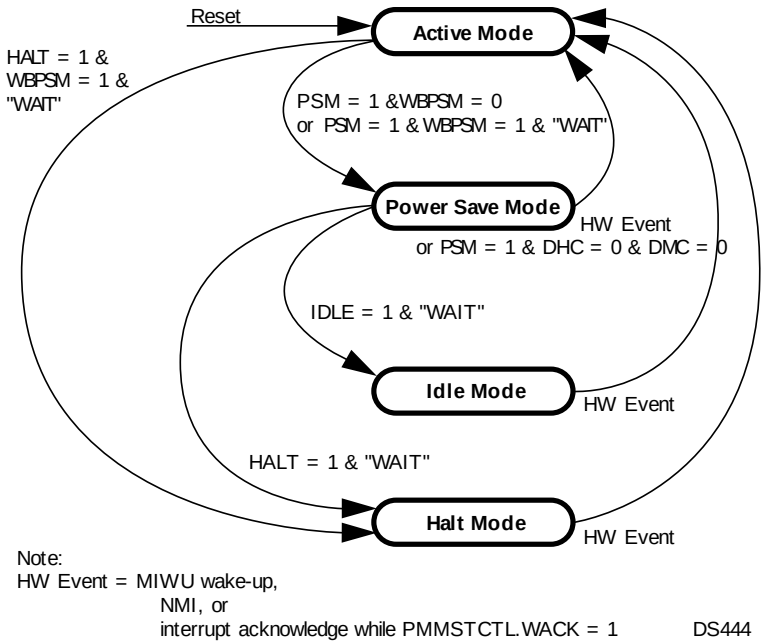


Figure 16-1. Power Mode State Diagram

Software changes the power mode in either of two ways, as controlled by the PMMSTCTL.WBPSM bit:

- *Immediate transition (WBPSM = 0)*—new clock sources take effect as soon as stable clocks are available. Only allowed when going from Active to Power Save mode.
- *Transition at next WAIT instruction (WBPSM = 1)*—device continues to operate in Active mode until it executes a WAIT instruction. At execution of the WAIT instruction, the device enters the new mode, and the CPU waits for the next interrupt event.

A transition to Idle or Halt mode must use the second method (WBPSM = 1).

Some of the power-up transitions are based on the occurrence of a wake-up event:

- *Multi-Input Wake-Up Event*—most general-purpose I/O port pins, certain peripheral pins (that don't share functionality with a GPIO port), and certain peripheral events can be programmed in the MIWU module to trigger a wake-up event.
- *Non-Maskable Interrupt (NMI)*—an NMI interrupt is a wake-up event.
- *Interrupt Acknowledge Cycle*—when the WACK bit in the PMMSTCTL register is set, any interrupt acknowledge cycle is a wake-up event.

Once a wake-up event is detected, it is latched until an interrupt acknowledge cycle or reset occurs.

A wake-up event causes a transition to the Active mode and restores normal clock operation, but does not start execution of the program. It is the interrupt handler associated with the wake-up source (MIWU, NMI, or any interrupt) that causes program execution to resume.

16.6.1 Active Mode to Power Save Mode

A transition from Active mode to Power Save mode is performed by writing a 1 to the PMMSTCTL.PSM bit. The transition to Power Save mode occurs immediately or at the execution of the next WAIT instruction, depending on the state of the PMMSTCTL.WBPSM bit. The PSM bit operates differently depending on this mode:

- *WBPSM = 0*—PSM bit reads as 1 after the transition to the Power Save mode occurs. If the high-frequency oscillator or PLLs are not producing a stable output, this transition may be stalled. In this mode, the PSM bit indicates when the transition has been completed.
- *WBPSM = 1*—PSM bit reads as 1 after it is written with 1, even before the WAIT instruction has executed and the transition to Power Save mode has occurred.

16.6.2 Entering Idle Mode

Idle mode can only be entered from Power Save mode. Idle mode is entered by writing 1 to the PMMSTCTL.IDLE bit and then executing a WAIT instruction.

16.6.3 Disabling the High-Frequency Clock

When the low-frequency oscillator is used to generate the Slow Clock, power consumption can be reduced further in the Power Save or Idle mode by disabling the high-frequency oscillator. This is accomplished by writing a 1 to the PMMSTCTL.DMC bit before executing the WAIT instruction that puts the device in the Power Save or Idle mode. The high-frequency clock is turned off only after the device enters the Power Save or Idle mode.

The CPU operates from Slow Clock in Power Save mode. It can turn off the high-frequency oscillator at any time by writing a 1 to the PMMSTCTL.DMC bit. The high-frequency oscillator is always enabled in Active mode and is always disabled in Halt mode, without regard to the PMMSTCTL.DMC bit.

Immediately after power-up and entry into Active mode, software must wait for the low-frequency oscillator to become stable before it can put the device in Power Save mode. It should monitor the PMMSR.SCLKSTB bit for this purpose. Once this bit is set, the low-frequency oscillator is stable and Power Save mode can be entered.

Slow Clock must come from the low-frequency oscillator or an external clock (driven on X2CKI), not derived from Main Clock, if the high-frequency oscillator will be disabled.

16.6.4 Entering Halt Mode

Halt mode can only be entered from the Active and Power Save modes. In Active mode, the PMMSTCTL.WBPSM bit must be set before entering Halt mode. Halt mode is entered by writing 1 to the PMMSTCTL.HALT bit and then executing a WAIT instruction.

16.6.5 Software-Controlled Transition to Active Mode

A transition from Power Save mode to Active mode can be accomplished by either a software method or a hardware wake-up event. The software method is to write a 0 to the PMMSTCTL.PSM bit. The value of the register bit changes only after the transition to the Active mode is completed.

If the high-frequency oscillator is disabled for Power Save operation, the oscillator must be enabled and allowed to stabilize before the transition to Active mode. To enable the high-frequency oscillator, clear the PMMSTCTL.DMC bit. Before clearing the PMMSTCTL.PSM bit, software must poll the PMMSR.MCLKSTB bit to determine when the oscillator has stabilized.

16.6.6 Wake-Up Transition to Active Mode

A hardware wake-up event switches the device directly from Power Save, Idle, or Halt mode to Active mode. When a wake-up event occurs, the on-chip hardware performs the following steps:

1. Clears the PMMSTCTL.DMC, PMMCSTCTL.DSC, and PMMPLLnCTL2.DPLLC bits, which enable the high-frequency oscillator, low-frequency oscillator, and PLL Clocks (if any were disabled).
2. Waits for the PMMSR.MCLKSTB and PMMSR.SCLKSTB bits to become set. If either of the PLLs were enabled before entering the low-power mode, waits for the corresponding PMMPLLnCTL2.PLLCLKSTB bits to become set.
3. Switches the device into Active mode.

16.6.7 Power Mode Switching Protection

The Power Management Module has several mechanisms to protect the device from malfunctions caused by missing or unstable clock signals.

The PMMSR.MCLKSTB, PMMSR.SCLKSTB, and PMMPLLnCTL2.PLLCLKSTB bits indicate the current status of the high-frequency oscillator, low-frequency oscillator, and PLLs, respectively. Software can check the appropriate bit before switching to a power mode that requires a specific clock. A set bit indicates an operating, stable clock. A clear bit indicates a clock that is disabled, not available, or not yet stable.

During a power mode transition, if there is a request to switch to a mode which uses a clock with a clear stability bit, the switch is delayed until the bit is set by the hardware.

When the system is built without an external crystal network for the low-frequency clock, Main Clock or a PLL Clock is divided by a prescaler to produce the low-frequency clock. In this situation, Main Clock is disabled only in the Halt mode, and cannot be disabled for the Power Save or Idle mode.

Following reset, Main Clock drives Slow Clock through a prescaler. The prescaler value results in a Slow Clock rate of 32.786.885 Hz.

NOTE

For correct operation in the absence of a low-frequency crystal, the X2CKI pin must be tied low (not left floating) so that the hardware can detect the absence of the crystal.

16.7 Power Management Register

Table 16-3 shows the power management register.

Table 16-3. Power Management Register

NAME	ADDRESS	DESCRIPTION
PMMSTCTL	FF A404h	Power Management State Control Register

16.7.1 Power Management State Control Register (PMMSTCTL)

The PMMSTCTL register is a byte-wide, read/write register that controls the operating power mode (Active, Power Save, Idle, or Halt) and enables or disables the high-frequency oscillator in the Power Save and Idle modes. At reset, the register are cleared. The format of the register is shown below. At reset, the register is cleared.

7	6	5	4	3	2	1	0
DSC	WACK	HCCM	DMC	WBPSM	HALT	IDLE	PSM

PSM — When the Power Save Mode bit is clear, the system is in Active mode. If the WBPSM bit is clear, writing 1 to the PSM bit causes the device to start the switch to Power Save mode. If the WBPSM bit is set when the PSM bit is written with 1, entry into Power Save mode is delayed until execution of a WAIT instruction. The PSM bit becomes set after the switch to Power Save mode is complete. The PSM bit can be cleared by software, and it can be cleared by hardware when a hardware wakeup event is detected.
0 – Device is not in Power Save mode.
1 – Device is in Power Save mode.

IDLE — The Idle Mode bit indicates whether the device has entered Idle mode. If the WBPSM bit is clear and the device is in Power Save mode, writing 1 to the IDLE bit causes the device to start the switch to IDLE mode. If the WBPSM bit is set when the IDLE bit is written with 1, entry into Idle mode is delayed until execution of a WAIT instruction. The IDLE bit can be set and cleared by software. It is also cleared by hardware when a hardware wake-up event is detected. When set, the IDLE bit overrides the PSM bit.
0 – Device is not in Idle mode.
1 – Device is in Idle mode.

HALT The Halt Mode bit indicates whether the device is in Halt mode. If the WBPSM bit is clear and the device is in Power Save mode, writing 1 to the HALT bit causes the device to start the switch to Halt mode. If the WBPSM bit is set when the HALT bit is written with 1, entry into Halt mode is delayed until execution of a WAIT instruction. When in Halt mode, the PMM stops HCLK Clock and then turns off the high-frequency oscillator and PLLs. The low-frequency oscillator also may be stopped in Halt mode by setting the PMMSTCTL.DSC bit. The HALT bit can be set and cleared by software. Halt mode is exited by a hardware wake-up event. When this signal is set high, the oscillator is started. After the oscillator has stabilized, the HALT bit is cleared by hardware. When set, the HALT bit overrides the IDLE and PSM bits.
0 – Device is not in Halt mode.
1 – Device is in Halt mode.

WBPSM When the Wait Before Power Save Mode bit is clear, a switch from Active mode to Power Save mode only requires setting the PSM bit. When the WBPSM bit is set, a switch from Active mode to Power Save, Idle, or Halt mode is performed by setting the PSM, IDLE, or HALT bit, respectively, and then executing a WAIT instruction. Also, if the DMC or DPLLC bits are set, the high-frequency oscillator or PLLs may be disabled only after a WAIT instruction is executed and the Power Save, Idle, or Halt mode is entered.
0 – Mode transitions may occur immediately.
1 – Mode transitions are delayed until the next WAIT instruction is executed.

DMC The Disable Main Clock bit may be used to disable Main Clock and the high-frequency oscillator in Power Save and Idle modes. In Active mode, Main Clock is enabled without regard to the DMC value. In Halt mode, Main Clock is disabled without regard to the DMC value. Disabling Main Clock will also disable the PLL Clocks. The DMC bit is cleared by hardware when a hardware wake-up event is detected.
0 – Main Clock is only disabled in Halt mode or when disabled by the HCC mechanism.
1 – Main Clock is also disabled in Power Save and Idle modes, unless Slow Clock is generated from Main Clock or PLL Clock.

HCCM The Hardware Clock Control for Main Clock bit may be used in Power Save and Idle modes to disable Main Clock and the high-frequency oscillator conditionally, depending on whether the Bluetooth LLC is in Sleep mode. Disabling Main Clock has the side-effect of disabling the PLL Clocks. The DMC bit must be clear for this mechanism to operate. The HCCM bit is automatically cleared when the device enters Active mode.
0 – Main Clock is disabled in Power Save or Idle mode only if the DMC bit is set.
1 – Main Clock is also disabled if the Bluetooth LLC is idle.

WACK The Wake-Up on Interrupt Acknowledge bit is set to enable a mode in which the system will wake up when an interrupt acknowledge occurs. This could be used in Power Save mode without using the WAIT instruction, so that an interrupt could be used to power up the system quickly, rather than needing the software to do it. If the WACK bit is clear, then an interrupt acknowledge will not cause a wake up from a low power mode.
0 – Interrupt acknowledge does not wake up from low-power mode.
1 – Interrupt acknowledge wakes up from low-power mode.

DSC The Disable Slow Clock bit is used to disable the low-frequency oscillator in Halt mode. This bit would typically be set for maximum power savings, but the low-frequency oscillator is optimized for very low power so it may not be necessary to disable it. Leaving the oscillator running avoids a start-up delay for the low-frequency oscillator. The DSC bit is cleared when a hardware wake-up event occurs.
0 – Low-frequency oscillator is not disabled.
1 – Low-frequency oscillator is disabled when Halt mode is entered.

17 Multi-Input Wake-Up

The Multi-Input Wake-Up (MIWU) module allows most general-purpose I/O port pins, certain peripheral pins (that do not share functionality with a GPIO port), and certain peripheral events to break the system out of a low-power mode and return to Active mode. Each MIWU input can be programmed to assert a wake-up signal for exiting from a low-power mode, and each input can be independently programmed to assert an interrupt request on any of eight maskable interrupts assigned to the MIWU module.

Eight types of registers control the generation of interrupts and wake-up events, and one register indicates the status of pending interrupt requests:

- *WKRPNDx*—two 32-bit registers indicate which MIWU inputs have detected rising edges since they were last cleared (either by reset or writing to WKCLRx).
- *WKFPNDx*—two 32-bit registers indicate which MIWU inputs have detected falling edges since they were last cleared (either by reset or writing to WKCLRx).
- *WKCLRx*—writing 1 to bits in either of these two 32-bit registers clears the corresponding bits in the WKRPNDx and WKFPNDx registers. The WKCLRx registers are write-only.
- *WKRENx*—two 32-bit registers hold bits for enabling a wake-up event when a rising edge is detected on the corresponding MIWU input.
- *WKFENx*—two 32-bit registers hold bits for enabling a wake-up event when a falling edge is detected on the corresponding MIWU input.
- *WKREINx*—two 32-bit registers hold bits for enabling an interrupt when a rising edge is detected on the corresponding MIWU input.
- *WKFEINx*—two 32-bit registers hold bits for enabling an interrupt when a falling edge is detected on the corresponding MIWU input.
- *WKICTLx*—four 32-bit registers provide two-bit fields to select one of four MIWU interrupt channels available when an enabled interrupt occurs. MIWU inputs 31:0 can only assert MIWU interrupts 3:0, while MIWU inputs 63:32 can only assert MIWU interrupts 7:4.
- *WKISTAT*—indicates pending MIWU interrupts.

The MIWU module is always active, including in Halt mode when all device clocks are stopped. Therefore, detecting an external trigger condition and setting bits in WKRPNDx and WKFPNDx do not require an active HCLK Clock or PCLK Clock.

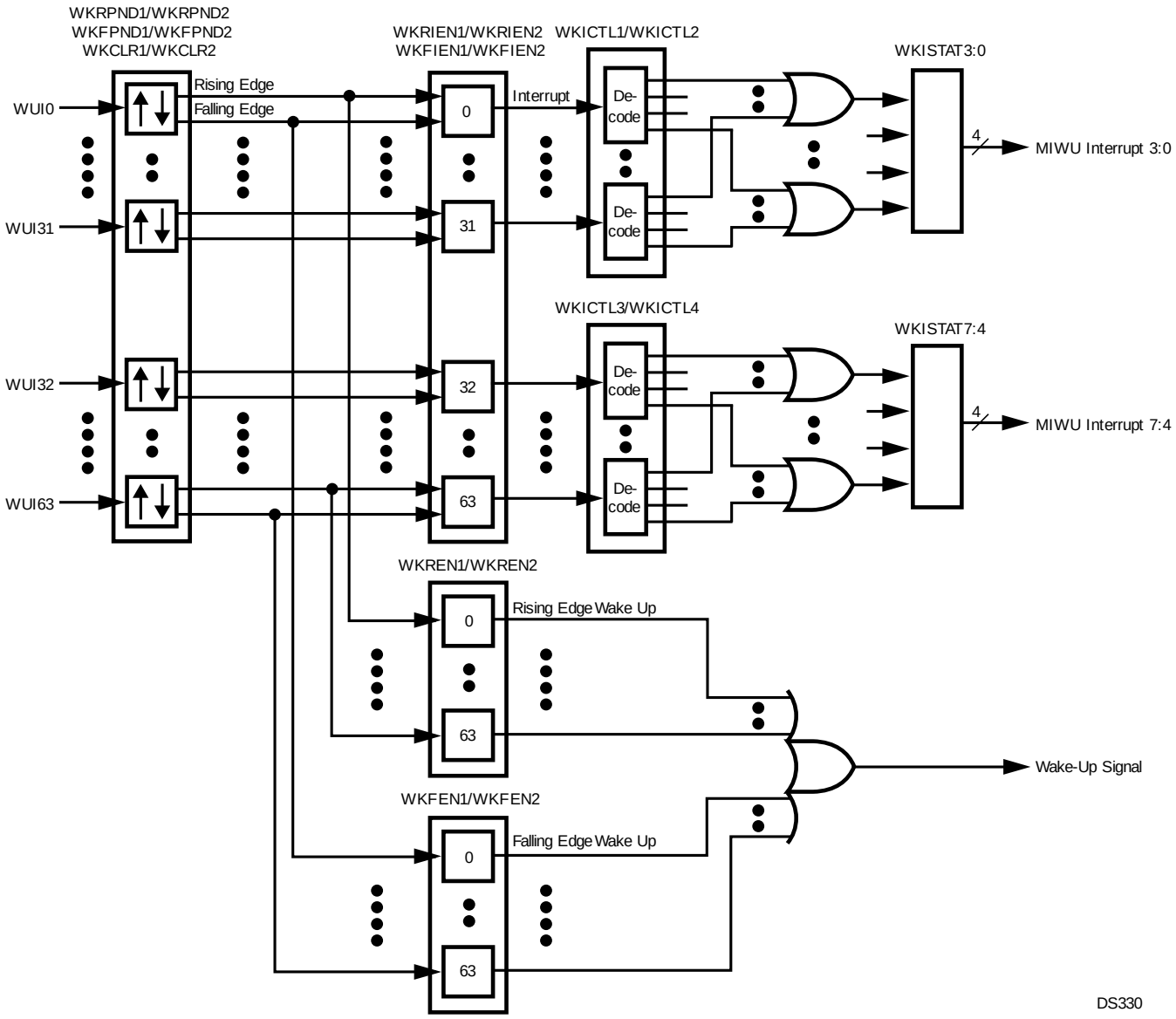


Figure 17-1. Multi-Input Wake-Up Module Block Diagram

Table 17-1 lists the sources connected to the MIWU inputs. A source which is a GPIO port pin cannot be used unless it is enabled for use as an input, either by enabling an alternate function in which the pin is used as an input or by setting its corresponding bit in a PxIEN register. If this is not done, the input logic for the pin is disabled.

Table 17-1. MIWU Sources

MIWU CHANNEL	SOURCE		MIWU CHANNEL	SOURCE
WUI0	PE0		WUI32	PF0
WUI1	PE1		WUI33	PF1
WUI2	PE2		WUI34	PF2
WUI3	PE3		WUI35	PF3
WUI4	PE4		WUI36	PF4
WUI5	PE5		WUI37	PF5
WUI6	PE6		WUI38	PF6

Table 17-1. MIWU Sources (continued)

MIWU CHANNEL	SOURCE		MIWU CHANNEL	SOURCE
WUI7	PE7		WUI39	PF7
WUI8	PE8		WUI40	PF8
WUI9	PE9		WUI41	PF9
WUI10	PE10		WUI42	PF10
WUI11	PE11		WUI43	PF11
WUI12	PE12		WUI44	PF12
WUI13	PE13		WUI45	PF13
WUI14	PE14		WUI46	PF14
WUI15	PE15		WUI47	PF15
WUI16	PG0		WUI48	PH0
WUI17	PG1		WUI49	PH1
WUI18	PG2		WUI50	PH2
WUI19	PG3		WUI51	DSP
WUI20	PG4		WUI52	ACCESS.bus 0
WUI21	PG5		WUI53	A/D Converter
WUI22	PG6		WUI54	Bluetooth LLC
WUI23	PG7		WUI55	Reserved
WUI24	PG8		WUI56	USB
WUI25	PG9		WUI57	Reserved
WUI26	PG10		WUI58	ACCESS.bus 1
WUI27	PG11		WUI59	Reserved
WUI28	PG12		WUI60	TWM T0OUT
WUI29	PG12		WUI61	RTC RTCEVT3
WUI30	PG14		WUI62	RTC RTCEVT2
WUI31	PG15		WUI63	RTC RTCEVT1

17.1 Multi-Input Wake-Up Registers

Table 17-2 lists the MIWU registers.

Table 17-2. Multi-Input Wake-Up Registers

NAME	ADDRESS	DESCRIPTION
WKRPND1	FF C020h	Rising Edge Pending Register 1
WKRPND2	FF C024h	Rising Edge Pending Register 2
WKFPND1	FF C030h	Falling Edge Pending Register 1
WKFPND2	FF C034h	Falling Edge Pending Register 2
WKCLR1	FF C040h	Clear Pending Register 1
WKCLR2	FF C044h	Clear Pending Register 2
WKREN1	FF C000h	Rising Edge Enable Register 1
WKREN2	FF C004h	Rising Edge Enable Register 2
WKFEN1	FF C010h	Falling Edge Enable Register 1
WKFEN2	FF C014h	Falling Edge Enable Register 2
WKRIEN1	FF C050h	Rising Edge Interrupt Enable Register 1
WKRIEN2	FF C054h	Rising Edge Interrupt Enable Register 2
WKFIEN1	FF C060h	Falling Edge Interrupt Enable Register 1
WKFIEN2	FF C064h	Falling Edge Interrupt Enable Register 2
WKICTL1	FF C070h	Interrupt Control Register 1

Table 17-2. Multi-Input Wake-Up Registers (continued)

NAME	ADDRESS	DESCRIPTION
WKICTL2	FF C074h	Interrupt Control Register 2
WKICTL3	FF C078h	Interrupt Control Register 3
WKICTL4	FF C07Ch	Interrupt Control Register 4
WKISTAT	FF C090h	Interrupt Status Register

17.1.1 Rising Edge Pending Register n (WKRPNDn)

The WKRPNDn registers are 32-bit, read/write registers that indicate whether a rising edge has been detected on the corresponding MIWU input. Bits 31:0 of WKRPND1 correspond to MIWU inputs 31:0 Bits 31:0 of WKRPND2 correspond to MIWU inputs 63:32. Writing 1 to bits in the WKCLRn registers clears the corresponding bits in the WKRPNDn registers. The WKRPNDn registers are cleared at reset. The register format is shown below.

31	0
WKRPD	

WKRPD The Wake-Up Rising Edge Pending bits indicate whether a rising edge has occurred on the corresponding MIWU inputs since the bits were last cleared.
0 – No rising edge occurred.
1 – Rising edge occurred.

17.1.2 Falling Edge Pending Register n (WKFPNDn)

The WKFPNDn registers are 32-bit, read/write registers that indicate whether a falling edge has been detected on the corresponding MIWU input. Bits 31:0 of WKFPND1 correspond to MIWU inputs 31:0 Bits 31:0 of WKFPND2 correspond to MIWU inputs 63:32. Writing 1 to bits in the WKCLRn registers clears the corresponding bits in the WKFPNDn registers. The WKFPNDn registers are cleared at reset. The register format is shown below.

31	0
WKFPD	

WKFPD The Wake-Up Falling Edge Pending bits indicate whether a falling edge has occurred on the corresponding MIWU inputs since the bits were last cleared.
0 – No falling edge occurred.
1 – Falling edge occurred.

17.1.3 Clear Pending Register n (WKCLRn)

The WKCLRn registers are 32-bit, write-only registers that clear bits in the WKRPNDn and WKFPNDn registers. Writing 1 to a WKCLRn bit clears the corresponding bit in each of the other registers. WKCLR1 clears bits in WKRPND1 and WKFPND1, and WKCLR2 clears bits in WKRPND2 and WKFPND2. The register format is shown below.

31	0
WKCL	

WKCL The Wake-Up Clear bits are used to selectively clear bits in WKRPNDn and WKFPNDn.
0 – Writing 0 has no effect.
1 – Writing 1 clears the corresponding bit position in WKRPNDn and WKFPNDn.

17.1.4 Rising Edge Enable Register n (WKRENn)

The WKRENn registers are 32-bit, read/write registers that enable a wake-up event when a rising edge is detected on the corresponding MIWU input. Bits 31:0 of WKREN1 correspond to MIWU inputs 31:0 Bits 31:0 of WKREN2 correspond to MIWU inputs 63:32. The WKRENn registers are cleared at reset. The register format is shown below.

31	0
WKREN	

WKREN The Wake-Up Rising Edge Enable bits enable a wake-up event when a rising edge occurs on the corresponding MIWU input.
0 – No wake-up event enabled.
1 – Wake-up event enabled.

17.1.5 Falling Edge Enable Register n (WKFENn)

The WKFENn registers are 32-bit, read/write registers that enable a wake-up event when a falling edge is detected on the corresponding MIWU input. Bits 31:0 of WKFEN1 correspond to MIWU inputs 31:0 Bits 31:0 of WKFEN2 correspond to MIWU inputs 63:32. The WKFENn registers are cleared at reset. The register format is shown below.

31	0
WKFEN	

WKFEN The Wake-Up Falling Edge Enable bits enable a wake-up event when a falling edge occurs on the corresponding MIWU input.
0 – No wake-up event enabled.
1 – Wake-up event enabled.

17.1.6 Rising Edge Interrupt Enable Register n (WKRIENn)

The WKRIENn registers are 32-bit, read/write registers that enable an interrupt when a rising edge is detected on the corresponding MIWU input. Bits 31:0 of WKRIEN1 correspond to MIWU inputs 31:0 Bits 31:0 of WKRIEN2 correspond to MIWU inputs 63:32. The WKRIENn registers are cleared at reset. The register format is shown below.

31	0
WKRIEN	

WKRIEN The Rising Edge Interrupt Enable bits enable an interrupt when a rising edge occurs on the corresponding MIWU input.
0 – No interrupt enabled.
1 – Interrupt enabled.

17.1.7 Falling Edge Interrupt Enable Register n (WKFIENn)

The WKFIENn registers are 32-bit registers, read/write that enable an interrupt when a falling edge is detected on the corresponding MIWU input. Bits 31:0 of WKFIEN1 correspond to MIWU inputs 31:0 Bits 31:0 of WKFIEN2 correspond to MIWU inputs 63:32. The WKFIENn registers are cleared at reset. The register format is shown below.

31	0
WKFIEN	

WKFIEN The Falling Edge Interrupt Enable bits enable an interrupt when a falling edge occurs on the corresponding MIWU input.
0 – No interrupt enabled.
1 – Interrupt enabled.

17.1.8 Interrupt Control Register n (WKICTLn)

The WKICTLn registers are 32-bit, read/write registers that provide 2-bit fields which select the MIWU interrupt channels used by the associated MIWU channels. Each MIWU input can only activate one of four interrupt channels. MIWU inputs 31:0 can only activate MIWU interrupt channels 3:0, and MIWU inputs 63:32 can only activate MIWU interrupt channels 7:4.

The WKICTL1 register selects interrupts for MIWU inputs 15:0, WKICTL2 selects interrupts for MIWU inputs 31:16, WKICTL3 selects interrupts for inputs 47:32, and WKICTL4 selects interrupts for inputs 63:48. At reset, the WKICTLn registers are cleared. The register format of WKICTL1 is shown below (the other registers have a similar format).

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
WKIN TR7	WKIN TR6	WKIN TR5	WKIN TR4	WKIN TR3	WKIN TR2	WKIN TR1	WKIN TR0

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
WKIN TR15	WKIN TR14	WKIN TR13	WKIN TR12	WKIN TR11	WKIN TR10	WKIN TR9	WKIN TR8

WKINTR The Interrupt Request fields select which of four available MIWU interrupt channels are activated for the corresponding MIWU input.
00 – Selects MIWU interrupt channel 0 or 4.
01 – Selects MIWU interrupt channel 1 or 5.
10 – Selects MIWU interrupt channel 2 or 6.
11 – Selects MIWU interrupt channel 3 or 7.

17.1.9 Interrupt Status Register (WKISTAT)

The WKISTAT register is a 32-bit read-only register that indicates the status of the 8 MIWU interrupt channels. No interrupt channels are active following reset. The register format is shown below.

31 8	7 0
Reserved	WKIST

WKIST The Interrupt Status bits indicate which MIWU interrupt channels are currently asserting an interrupt request.
0 – No interrupt request is being asserted.
1 – An interrupt request is being asserted.

18 Input/Output Ports

The CP3SP33 has 64 software-configurable general-purpose I/O pins (36 on the FBGA-144 package), organized into four ports, named Port E, Port F, Port G, and Port H. All 16 pins on Ports E, F, and G and three pins on 8-bit Port H have independently programmable MIWU channels, which give them the capability to interrupt the CPU and wake up the system from low-power modes.

In addition to their general-purpose I/O function, most of the pins on Ports E, F, and G have alternate functions for use with on-chip peripheral modules such as the UARTs and timers. Architecturally, there may be two alternate functions for a port pin, and software must specify which function to select (A or B), even when only one function is implemented. The alternate functions of all I/O pins are shown in Table 18-2. To avoid glitches, software should first select the alternate function, then enable the alternate function, and finally initialize the module which uses the alternate function.

The I/O pin characteristics are fully programmable. Each pin can be configured to operate as a TRI-STATE output, push-pull output, input with weak pullup, input with weak pull-down, or high-impedance input. Different pins within the same port can be individually configured to operate in different modes.

Figure 18-1 shows the I/O port pin logic. The register bits, multiplexers, and buffers allow the port pin to be configured into the various operating modes.

To reduce power consumption, input buffers configured for general-purpose I/O are only enabled when they are read. When configured for an alternate function, the input buffers are enabled continuously. To minimize power consumption, input signals to enabled buffers must be held within 0.2 volts of the VCC or GND voltage.

The electrical characteristics and drive capabilities of the input and output buffers are described in **Section 36**.

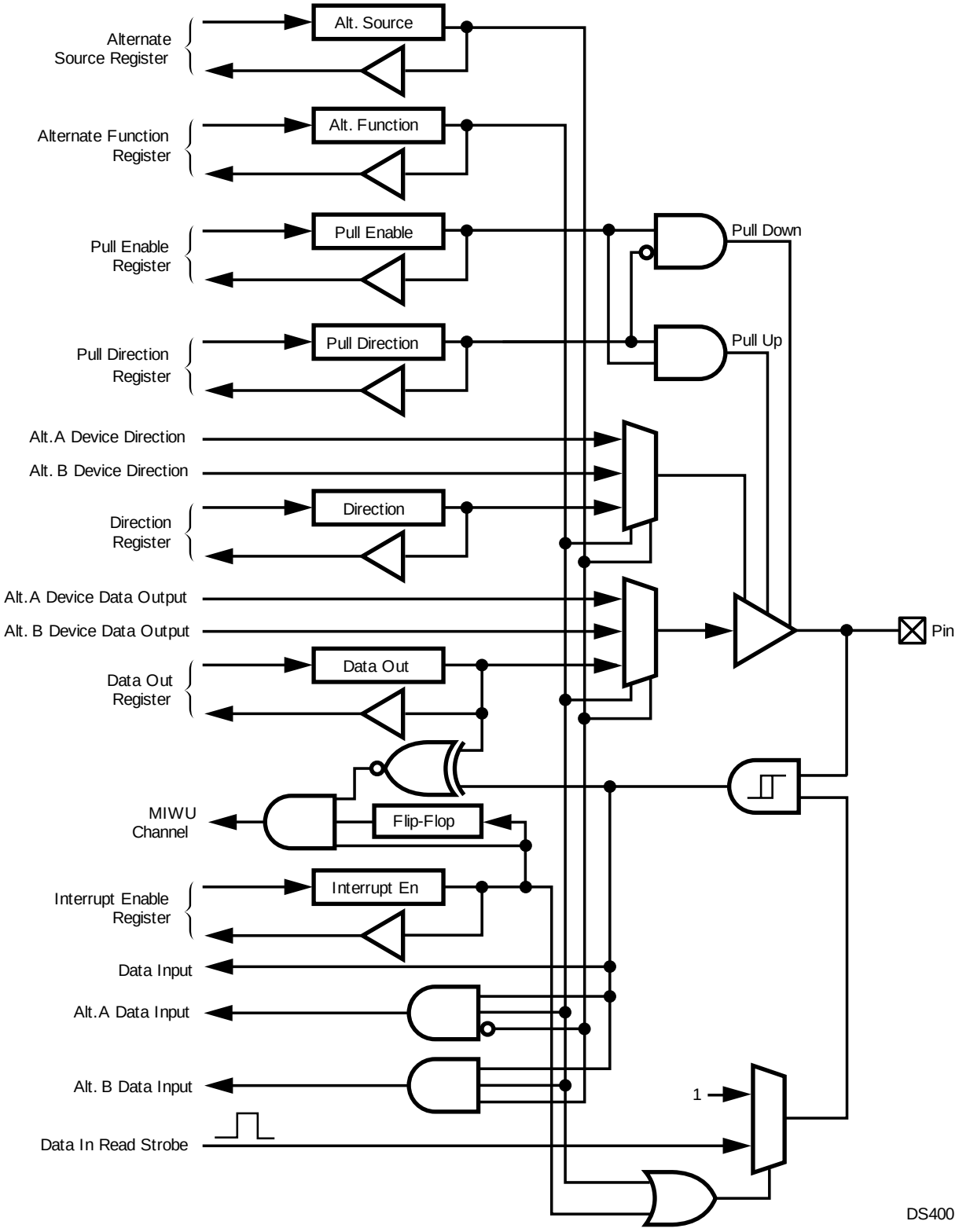


Figure 18-1. I/O Port Pin Logic

18.1 Open-Drain Operation

A port pin can be configured to operate as an inverting open-drain output buffer. To do this, software must clear the bit in the PxDOUT register and use the PxDIR bit to set the value of the port pin. In open-drain operation, the output buffer is toggled between driving logic low and TRI-STATE (high-impedance mode). This is achieved by using the PxDIR register to switch between driving a low output and configuring the port pin as an input. If desired, the internal weak pullup or pulldown can be enabled to pull the signal to a harmless state when the output buffer is in TRI-STATE mode.

18.2 Port Registers

Each port has a set of memory-mapped registers:

- *PxDIR*—specifies whether the port is an input or output, except when using an alternate function. When an alternate function is used, peripheral controls port direction.
- *PxDIN*—indicates the states on the port pins.
- *PxDOUT*—holds data driven on the port pins, when the port is configured as an output and no alternate function is used. When an alternate function is used, the peripheral device controls the data output. This register also specifies states which cause assertion of interrupts, when interrupts are enabled (see PxIEN register).
- *PxIEN*—enables an interrupt when the state on the port pin matches the state in PxDOUT register.
- *PxALT*—enables the port alternate function.
- *PxALTS*—selects between alternate functions (A and B)
- *PxWPU*—enables a weak pullup or pulldown
- *PxPDR*—selects the pullup/pulldown direction.

Table 18-1. Port Registers

NAME	ADDRESS	DESCRIPTION
PEDIR	FF C404h	Port E Direction Register
PEDIN	FF C408h	Port E Data Input Register
PEDOUT	FF C40Ch	Port E Data Output Register
PEIEN	FF C41Ch	Port E Interrupt Enable Register
PEALT	FF C400h	Port E Alternate Function Register
PEALTS	FF C418h	Port E Alternate Function Source Register
PEWPU	FF C410h	Port E Weak Pullup/Pulldown Enable Register
PEPDR	FF C420h	Port E Weak Pullup/Pulldown Direction Register
PFDIR	FF C804h	Port F Direction Register
PFDIN	FF C808h	Port F Data Input Register
PFDOUT	FF C80Ch	Port F Data Output Register
PFIEN	FF C81Ch	Port F Interrupt Enable Register
PFALT	FF C800h	Port F Alternate Function Register
PFALTS	FF C818h	Port F Alternate Function Source Register
PFWPU	FF C810h	Port F Weak Pullup/Pulldown Enable Register
PFPDR	FF C820h	Port F Weak Pullup/Pulldown Direction Register
PGDIR	FF 6404h	Port G Direction Register
PGDIN	FF 6408h	Port G Data Input Register
PGDOUT	FF 640Ch	Port G Data Output Register
PGIEN	FF 641Ch	Port G Interrupt Enable Register
PGALT	FF 6400h	Port G Alternate Function Register
PGALTS	FF 6418h	Port G Alternate Function Source Register
PGWPU	FF 6410h	Port G Weak Pullup/Pulldown Enable Register
PGPDR	FF 6420h	Port G Weak Pullup/Pulldown Direction Register

Table 18-1. Port Registers (continued)

NAME	ADDRESS	DESCRIPTION
PHDIR	FF CC04h	Port H Direction Register
PHDIN	FF CC08h	Port H Data Input Register
PHDOUT	FF CC0Ch	Port H Data Output Register
PHIEN	FF CC1Ch	Port H Interrupt Enable Register
PHALT	FF CC00h	Port H Alternate Function Register
PHALTS	FF CC18h	Port H Alternate Function Source Register
PHWPU	FF CC10h	Port H Weak Pullup/Pulldown Enable Register
PHPDR	FF CC20h	Port H Weak Pullup/Pulldown Direction Register

In the descriptions of the port registers, the lower-case letter “x” represents the port designation, either E, F, G, or H. For example, “PxDIR register” means any one of the port direction registers: PEDIR, PFDIR, PGDIR, or PHDIR.

All of the port registers are 16-bit read/write registers, except for the port data input registers, which are read-only registers. Each register bit controls the function of the corresponding port pin. For example, PGDIR.2 (bit 2 of the PGDIR register) controls the direction of port pin PG2.

18.2.1 Port Direction Register (PxDIR)

The PxDIR register selects whether the corresponding port pin is used for input or output. A reset operation clears the port direction registers, which initializes the pins as inputs.

15	0
PxDIR	

PxDIR The Port Direction bits select the direction of the corresponding port pin.
0 – Input.
1 – Output.

18.2.2 Port Data Input Register (PxDIN)

The PxDIN register is a read-only register that returns the current state on each port pin. The CPU can read this register at any time, even when the pin is configured as an output.

15	0
PxDIN	

PxDIN The Port Data In bits indicate the state on the corresponding port pin.
0 – Pin is low.
1 – Pin is high.

18.2.3 Port Data Output Register (PxDOUT)

The PxDOUT register holds the data to be driven on output port pins. When the pins are configured as outputs and no alternate function is enabled, writing to this register changes the output value. Reading the register returns the last value written to the register.

When an interrupt is enabled through the PxIEN register, the PxDOUT register specifies the signal level which asserts the interrupt.

A warm reset (software reset or watchdog reset) leaves the register contents unchanged. At power-up, the PxDOUT registers contain undefined data.

15	0
PxDOUT	

PxDOUT The Port Data Out bits hold the data to be driven on pins configured as outputs in general-purpose I/O mode. When interrupts are enabled in the PxIEN register, the PxDOUT bits specify the signal levels which assert interrupts.
0 – Drive output pin low.
1 – Drive output pin high.

18.2.4 Port Interrupt Enable Register (PxIEN)

The PxIEN registers enable the port input logic so that the corresponding pins can be used as wake-up inputs to the MIWU. All 16 pins on Ports E, F, and G and three pins on 8-bit Port H have independently programmable MIWU channels, which give them the capability to interrupt the CPU and wake up the system from low-power modes. If an alternate function is enabled which uses a pin as an input, the port input logic is already enabled, so it is not required to set the corresponding bit in its PxIEN register. The PxDOUT register specifies the signal level which asserts the wake-up signal to the MIWU. At reset, the register is cleared to 0000h.

15	0
PxIEN	

PxIEN The Port Interrupt Enable bits enable the port input logic so that the corresponding port pins can be used as MIWU inputs.
0 – Interrupt disabled.
1 – Interrupt enabled.

18.2.5 Port Alternate Function Register (PxALT)

The PxALT registers control whether the port pins are used for general-purpose I/O or for their alternate function. Each port pin can be controlled independently.

A clear bit in the alternate function register causes the corresponding pin to be used for general-purpose I/O. In this configuration, the port pin output buffer is controlled by the direction register (PxDIR) and the data output register (PxDOUT). The input buffer is visible to software as the data input register (PxDIN).

A set bit in the alternate function register (PxALT) causes the corresponding pin to be used for its peripheral I/O function. When the alternate function is selected, the port direction and output data are controlled by the peripheral device.

A reset clears the port alternate function registers, which initializes the pins as general-purpose I/O ports.

15	0
PxALT	

PxALT The Port Alternate Function bits control whether the corresponding port pins are general-purpose I/O ports or are used in their alternate function by an on-chip peripheral.
0 – General-purpose I/O selected.
1 – Alternate function selected.

18.2.6 Port Alternate Function Select Register (PxALTS)

The PxALTS register selects between two peripheral devices available for the alternate function of a port pin. These bits are ignored unless the corresponding PxALT bits are set. Each port pin can be controlled independently.

15	0
PxALTS	

PxALTS — The Alternate Function Select bits select among two alternate functions. Table 18-2 shows the mapping of the PxALTS bits to the alternate functions. Unused PxALTS bits must be clear.

Table 18-2. Alternate Function Select

PORT PIN	PxALTS = 0 (Device A)	PxALTS = 1 (Device B)	PORT PIN	PxALTS = 0 (Device A)	PxALTS = 1 (Device B)
PE0	USART0 RXD0	TIO0_1	PG0	I2SCLK	Reserved
PE1	USART0 TXD0	Reserved	PG1	I2SWS	Reserved
PE2	USART0 RTS0	TIO0_3	PG2	I2SSDI	Reserved
PE3	USART0 CTS0	Reserved	PG3	I2SSDO	Reserved
PE4	USART0 CKX0	TIO0_2	PG4	MSK1	Reserved
PE5	TA0	NMI	PG5	MDIDO1	Reserved
PE6	CAN1RX	Reserved	PG6	MDODI1	Reserved
PE7	CAN1TX	Reserved	PG7	MWCS1	TA1
PE8	MSK	Reserved	PG8	SRFS	UART3 RTS3
PE9	MDIDO0	Reserved	PG9	SCK	DIGMIC1
PE10	MDODI0	Reserved	PG10	SFS	TCIO1
PE11	MWCS0	TIO0_7	PG11	STD	DIGMIC2
PE12	SCL0	TIO0_5	PG12	SRD	Reserved
PE13	IDPULLUP	TIO1_1	PG13	SRCLK	UART3 CTS3
PE14	DRVVBUS	TIO1_3	PG14	UART3 RXD3	TB1
PE15	IDDIG	TIO1_5	PG15	UART3 TXD3	Reserved
PF0	RFSYNC	Reserved	PH0	TIO1_4	Reserved
PF1	SCLK	Reserved	PH1	SDA0	TIO1_6
PF2	SDAT	Reserved	PH2	CAN0RX	TIO1_7
PF3	SLE	Reserved	PH3	CAN0TX	TIO1_8
PF4	BTSEQ1	Reserved	PH4	Reserved	Reserved
PF5	BTSEQ2	Reserved	PH5	Reserved	Reserved
PF6	BTSEQ3	Reserved	PH6	Reserved	Reserved
PF7	ASYNC	TB0	PH7	Reserved	Reserved
PF8	UART1 RXD1	TIO0_4	PH8	Reserved	Reserved
PF9	UART1 TXD1	Reserved	PH9	Reserved	Reserved
PF10	UART1 RTS1	TIO0_6	PH10	Reserved	Reserved
PF11	UART1 CTS1	Reserved	PH11	Reserved	Reserved
PF12	UART2 RXD2	TIO0_8	PH12	Reserved	Reserved
PF13	UART2 TXD2	Reserved	PH13	Reserved	Reserved
PF14	UART2 RTS2	TIO1_2	PH14	Reserved	Reserved
PF15	UART2 CTS2	Reserved	PH15	Reserved	Reserved

18.2.7 Port Weak Pullup/Pulldown Enable Register (PxWPU)

The PxWPU register controls whether a weak pullup or pulldown device is enabled on the output buffer. The pullup or pulldown device, if enabled by the register bit, operates whenever the port output buffer is disabled (TRI-STATE mode). Enabling the pullup or pulldown device is unaffected by whether the port pin is in GPIO or alternate function mode. A reset operation clears the PxWPU registers, which disables all pullups and pulldowns.

15		0
	PxWPU	

PxWPU The Weak Pullup/Pulldown Enable bits control whether the weak pullup/pulldown device is enabled.
0 – Weak pullup/pulldown disabled.
1 – Weak pullup/pulldown enabled.

18.2.8 Port Weak Pullup/Pulldown Direction Register (PxPDR)

The PxPDR register selects whether a weak pullup or pulldown device is enabled on the output buffer. The pullup or pulldown device, if enabled in the PxWPU register, operates whenever the port output buffer is disabled (TRI-STATE mode). The PxPDR bits are ignored unless the corresponding PxWPU bits are set. At reset, the registers are initialized to FFFFh, which selects pullups.

15		0
	PxPDR	

PxPDR The Pullup/Pulldown Direction bits select between a pullup and a pulldown device.
0 – Pulldown selected.
1 – Pullup selected.

19 Bluetooth Controller

The integrated hardware Bluetooth Lower Link Controller (LLC) complies to the Bluetooth Specification Version 1.2 and integrates the following functions:

- 7K-byte dedicated Bluetooth data RAM
- 1K-byte dedicated Bluetooth sequencer RAM
- Support of all Bluetooth 1.2 packet types
- Support for fast frequency hopping of 1600 hops/s
- Access code correlation and slot timing recovery circuit
- Power Management Control Logic
- BlueRF-compatible interface to connect with National's LMX5252 and other RF transceiver chips

For a detailed description of the interface to the LMX5252, consult the LMX5252 data sheet which is available from the Texas Instruments wireless group. National provides software libraries for using the Bluetooth LLC. Documentation for the software libraries is also available from National Semiconductor.

19.1 RF Interface

The CP3SP33 interfaces to the LMX5251 or LMX5252 radio chips though the RF interface.

Figure 19-1 shows the interface between the CP3SP33 and the LMX5251 radio chip.

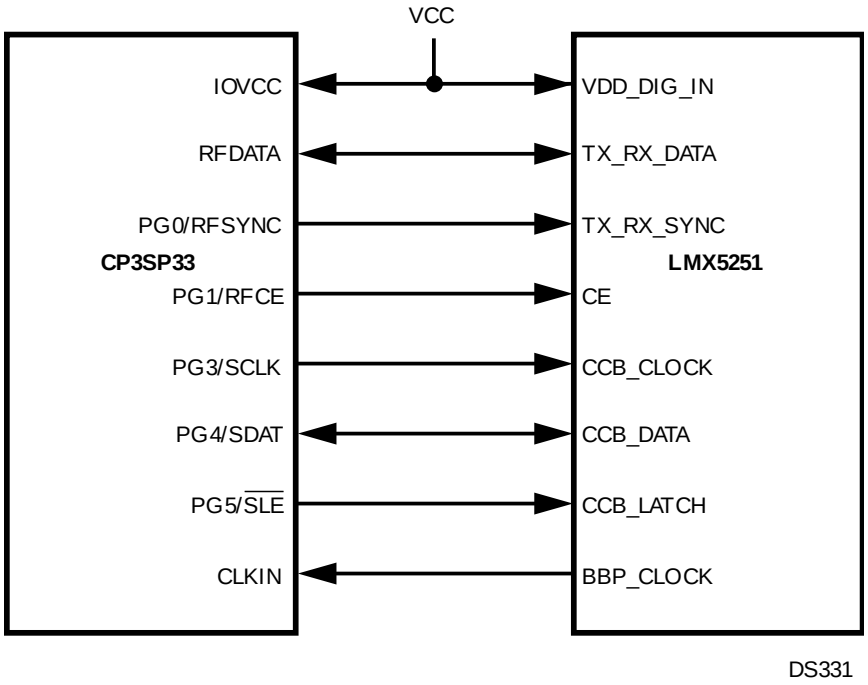


Figure 19-1. LMX5251 Interface

Figure 19-2 shows the interface between the CP3SP33 and the LMX5252 radio chip.

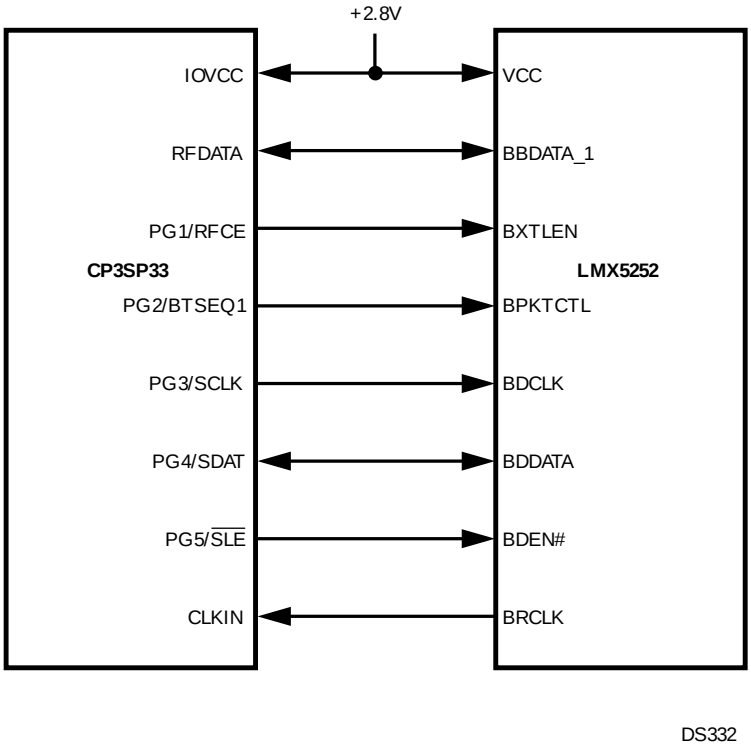


Figure 19-2. LMX5252 Interface

The CP3SP33 implements a BlueRF-compatible interface, which may be used with other RF transceiver chips.

19.1.1 RF Interface Signals

The RF interface signals are grouped as follows:

- Modem Signals (CLKIN, RFDATA, and RFSYNC)
- Control Signal (RFCE)
- Serial Interface Signals (SCLK, SDAT, and $\overline{\text{SLE}}$)
- Bluetooth Sequencer Status Signals (BTSEQ1, BTSEQ2, and BTSEQ2)

Software must enable the alternate functions of the RF interface signals to allow them to be controlled by the Bluetooth module.

19.1.1.1 CLKIN

The CLKIN pin is the input signal for the 12-MHz clock signal. The radio chip uses this signal internally as the 12× oversampling clock and provides it externally to the CP3SP33 for use as the Main Clock.

19.1.1.2 RFDATA

The RFDATA signal is the multiplexed Bluetooth data receive and transmit signal. The data is provided at a bit rate of 1 Mbit/s with 12× oversampling, synchronized to the 12 MHz CLKIN. The RFDATA signal is a dedicated RF interface pin. This signal is driven to a logic high level after reset.

19.1.1.3 RFSYNC

In receive mode (data direction from the radio chip to the CP3SP33), the RFSYNC signal acts as the frequency correction/DC compensation circuit control output to the radio chip. The RFSYNC signal is driven low throughout the correlation phase and driven high when synchronization to the received access code is achieved.

In transmit mode (data direction from the CP3SP33 to the radio chip), the RFSYNC signal enables the RF output of the radio chip. When the RFSYNC pin is driven high, the RF transmitter circuit of the radio chip is enabled, corresponding to the settings of the power control register in the radio chip

The RFSYNC signal is the alternate function of the general-purpose I/O pin PG0. At reset, this pin is in TRI-STATE mode. Software must enable the alternate function of the PG0 pin to give control over this signal to the RF interface.

19.1.1.4 RFCE

The RFCE signal is the chip enable output to the external RF chip. When the RFCE signal is driven high, the RF chip power is controlled by the settings of its power control registers. When the RFCE signal is driven low, the RF chip is powered-down. However, the serial interface is still operational and the CP3SP33 can still access the RF chip internal control registers.

The RFCE signal is the alternate function of the general-purpose I/O pin PG1. At reset, this pin is in TRI-STATE mode. Software must enable the alternate function of the PG1 pin to give control over this signal to the RF interface.

During Bluetooth power-down phases, the CP3SP33 provides a mechanism to reduce the power consumption of an external RF chip by driving the RFCE signal of the RF interface to a logic low level. This feature is available when the Power Management Module of the CP3SP33 has enabled the Hardware Clock Control mechanism. (However, the current version of the radio chip does not implement a power-reduction mode.)

19.1.1.5 SCLK

The SCLK signal is the serial interface shift clock output. The CP3SP33 always acts as the master of the serial interface and therefore always provides the shift clock. The SCLK signal is the alternate function of the general-purpose I/O pin PG3. At reset, this pin is in TRI-STATE mode. Software must enable the alternate function of the PG3 pin to give control over this signal to the RF interface.

19.1.1.6 SDAT

The SDAT signal is the multiplexed serial data receive and transmit path between the radio chip and the CP3SP33.

The SDAT signal is the alternate function of the general-purpose I/O pin PG4. At reset, this pin is in TRI-STATE mode. Software must enable the alternate function of the PG4 pin to give control over this signal to the RF interface.

$\overline{SLE}$
The $\overline{SLE}$ pin is the serial load enable output of the serial interface of the CP3SP33.

During write operations (to the radio chip registers), the data received by the shift register of the radio chip is copied into the address register on the next rising edge of SCLK after the $\overline{SLE}$ signal goes high.

During read operations (read from the registers), the radio chip releases the SDAT line on the next rising edge of SCLK after the SLE signal goes high

$\overline{SLE}$ is the alternate function of the general-purpose I/O pin PG5. At reset, this pin is in TRI-STATE mode. Software must enable the alternate function of the PG5 pin to give control over this signal to the RF interface.

19.1.1.7 BTSEQ[3:1]

The BTSEQ[3:1] signals indicate internal states of the Bluetooth sequencer, which are used for interfacing to some external devices.

19.2 Serial Interface

The radio chip register set can be accessed by the CP3SP33 through the serial interface. The serial interface uses three pins of the RF interface: SDAT, SCLK, and $\overline{SLE}$.

The serial interface of the CP3SP33 always operates as the master, providing the shift clock (SCLK) and load enable ($\overline{SLE}$) signal to the radio chip. The radio chip always acts as the slave.

A 25-bit shift protocol is used to perform read/write accesses to the radio chip internal registers. The complete protocol is comprised of the following sections:

- 3-bit Header Field
- Read/Write Bit
- 5-bit Address Field
- 16-bit Data Field

Header
The 3-bit header contains the fixed data 101b (except for Fast Write Operations).

Read/Write Bit
The header is followed by the read/write control bit (R/W). If the Read/Write bit is clear, a write operation is performed and the 16-bit data portion is copied into the addressed radio chip register.

Address
The address field is used to select one of the radio chip internal registers.

Data
The data field is used to transfer data to or from a radio chip register. The timing is modified for reads, to transfer control over the data signal from the CP3SP33 to the radio chip.

Figure 19-3 shows the serial interface protocol format.

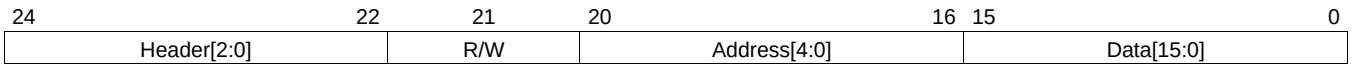


Figure 19-3. Serial Interface Protocol Format

Data is transferred on the serial interface with the most significant bit (MSB) first.

Write Operation

When the R/W bit is clear, the 16 bits of the data field are shifted out of the CP3SP33 on the falling edge of SCLK. Data is sampled by the radio chip on the rising edge of SCLK. When SLE is high, the 16-bit data are copied into the radio chip register on the next rising edge of SCLK. The data is loaded in the appropriate radio chip register depending on the state of the four address bits, Address[4:0]. Figure 19-4 shows the timing for the write operation.

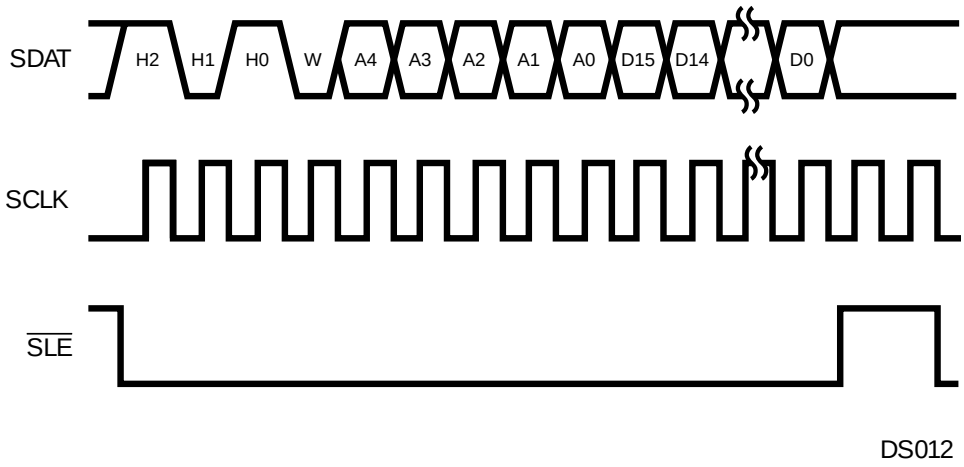


Figure 19-4. Serial Interface Write Timing

Read Operation

When the R/W bit is set, data is shifted out of the radio chip on the rising edge of SCLK. Data is sampled by the CP3SP33 on the falling edge of SCLK. On reception of the read command (R/W = 1), the radio chip takes control of the serial interface data line. The received 16-bit data is loaded by the CP3SP33 after the first falling edge of SCLK when $\overline{SLE}$ is high. When $\overline{SLE}$ is high, the radio chip releases the SDAT line again on the next rising edge of SCLK. The CP3SP33 takes control of the SDAT line again after the following rising edge of SCLK. Which radio chip register is read, depends on the state of the four address bits, Address[4:0]. The transfer is always 16 bits, without regard to the actual size of the register. Unimplemented bits contain undefined data. Figure 19-5 shows the timing for the read operation .

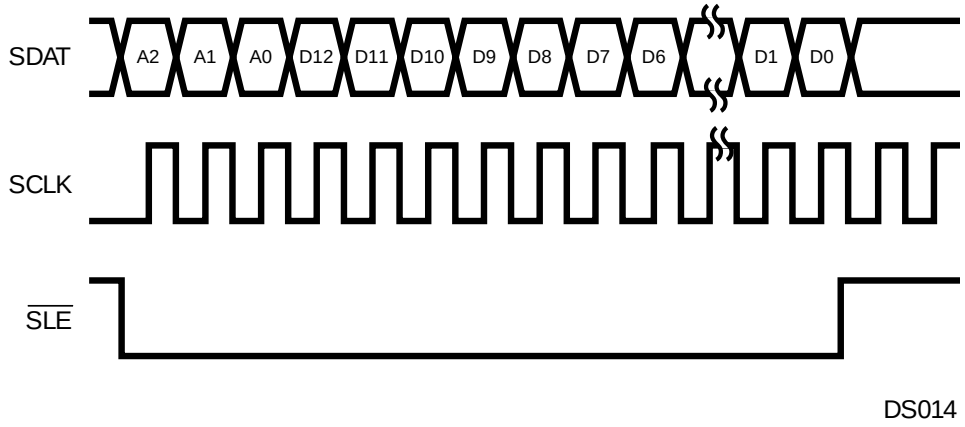


Figure 19-5. Serial Interface 16-bit Fast-Write Timing

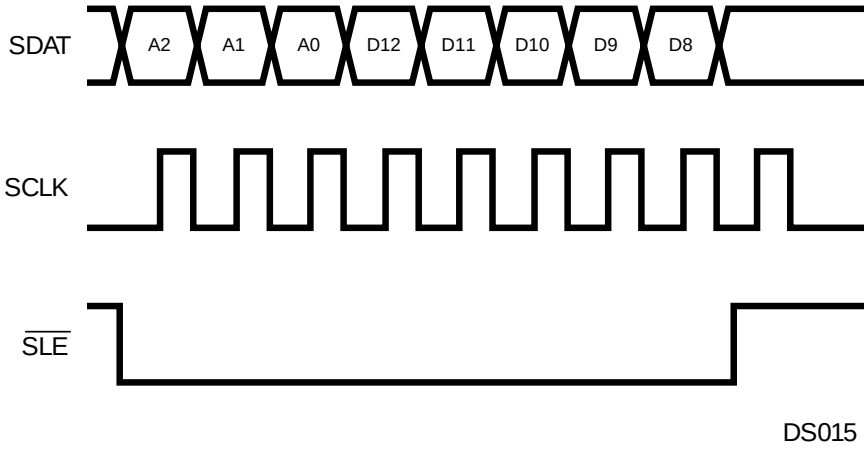


Figure 19-6. Serial Interface 8-bit Fast-Write Timing

32-Bit Write Operation

On the LMX5252, a 32-bit register is loaded by writing to the same register address twice. The first write loads the high word (bits 31:16), and the second write loads the low word (bits 15:0). The two writes must be separated by at least two clock cycles. For a 4-MHz clock, the minimum separation time is 500 ns.

The value read from a 32-bit register is a counter value, not the contents of the register. The counter value indicates which words have been written. If the high word has been written, the counter reads as 0000h. If both words have been written, the counter reads as 0001h. The value returned by reading a 32-bit register is independent of the contents of the register.

Figure 19-7 and Figure 19-8 show the timing for 32-bit register writing and reading.

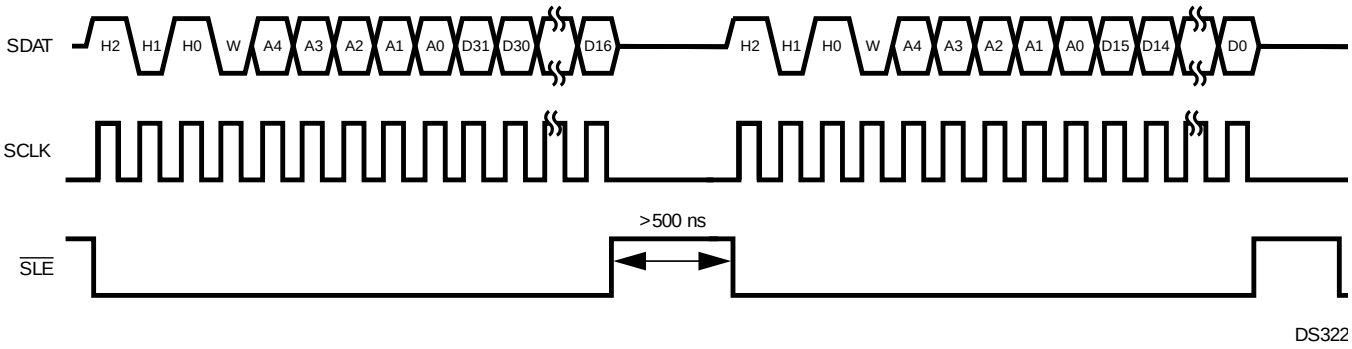


Figure 19-7. 32-Bit Write Timing

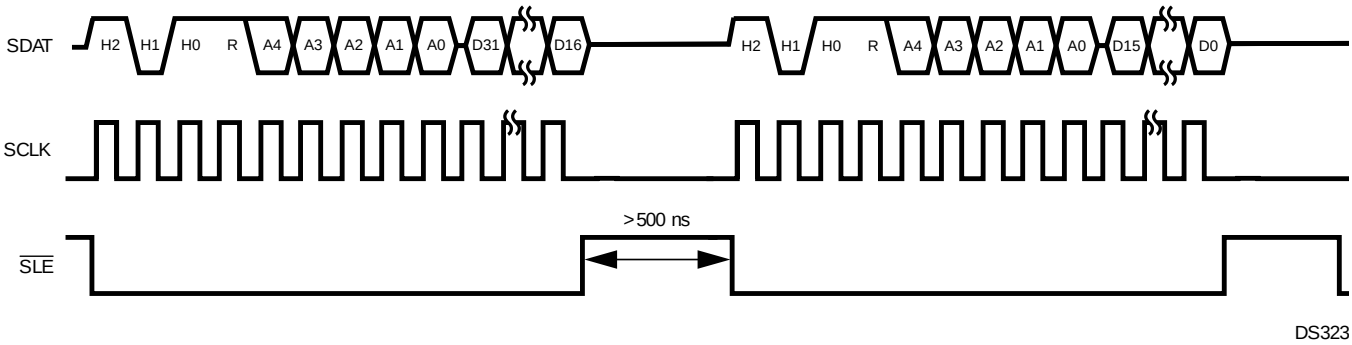


Figure 19-8. 32-Bit Read Timing

An example of a 32-bit write is shown in Table 40. In this example, the 32-bit value FFFF DC04h is written to register address 0Ah. In cycle 1, the high word (FFFFh) is written. In the first part of cycle 2, the CP3SP33 drives the header, R/W bit, and register address for a read cycle. In the second part of cycle 2, the LMX5252 drives the counter value. The counter value is 0, which indicates one word has been written. In cycle 3, the low word (DC04h) is written. In the first part of cycle 4, the CP3SP33 drives the header, R/W bit, and register address for a read cycle. In the second part of cycle 4, the LMX5252 drives the counter value. The counter value is 1, which indicates two words have been written.

Table 19-1. Example of 32-Bit Write with Interleaved Reads

CYCLE	SERIAL DATA ON SDAT	DESCRIPTION
1	101 0 01010 1111111111111111	Write cycle driven by CP3SP33. Data is FFFFh. Address is 0Ah.
2	101 1 01010	First part of read cycle driven by CP3SP33. Address is 0Ah.
	0000000000000000	Second part of read cycle driven by LMX5252. Counter value is 0.
3	101 0 01010 1101110000000100	Write cycle driven by CP3SP33. Data is DC04h. Address is 0Ah.
4	101 1 01010	First part of read cycle driven by CP3SP33. Address is 0Ah.
	0000000000000001	Second part of read cycle driven by LMX5252. Counter value is 1.

19.3 LMX5251 Power-Up Sequence

To power-up a Bluetooth system based on the CP3SP33 and LMX5251 devices, the following sequence must be performed.

1. Apply VDD to the LMX5251.
2. Apply IOVCC and VCC to the CP3SP33.
3. The RESET_N pin of the radio chip is directly connected to the Vuc pin of the radio chip and the RFVcc pin of the Pulsar. As the voltage on the RESET_N (also Vuc and RFVcc) is applied after applying a voltage to the Vdd pin, the LMX5251 is properly reset.
4. After (Power-On) reset of the Pulsar the RFDATA pin of the Pulsar is driven high. The RFCE, RFSYNC and SDAT pins of te Pulsar are set into high-impedance state. However, there are pullup/pulldown resistors built into the S_CLK, S_DATA, S_LE, and Rx/Tx_Synch pins of the radio chip to assure the correct initial voltage levels for a proper power-up sequence.
5. The RFDATA pin driven to logic high level causes the radio chip to enable its oscillator and after an oscillator start-up delay, the radio chip outputs a stable 12 MHz clock (BB_CLK/BRCLK) to the Pulsar.
6. By receiving a stable clock from the radio chip, the Pulsar is operational and the CR16 enables the alternate function of the GPIO pin, used by the RF interface.
7. Now the Bluetooth LLC can directly control the RF interface pins and sets the pins to the logic high levels required during the power-up phase. The RFCE pin driven high forces the radio chip to switch from “power-up” to “normal” mode and to disable it’s internal pullup/pulldown resistors of the RF interface.
8. In the “normal” mode the oscillator of the RF chip is controlled via the RFCE pin. Driving the RFCE pin to logic high level enables the oscillator and the RF chip outputs the BBCLK.
9. Drive the RESET# pin of the LMX5251 high a minimum of 2 ms after the LMX5251 and CP3000 supply rails are powered up. This resets the LMX5251 and CP3SP33.
10. After internal Power-On Reset (POR) of the CP3SP33, the RFDATA pin is driven high. The RFCE, RFSYNC, and SDAT pins are in TRI-STATE mode. Internal pullup/pulldown resistors on the CCB_CLOCK (SCLK), CCB_DATA (SDAT), CCB_LATCH (SLE), and TX_RX_SYNC (RFSYNC) inputs of the LMX5251 pull these signals to states required during the power-up sequence.
11. When the RFDATA pin is driven high, the LMX5251 enables its oscillator. After an oscillator start-up delay, the LMX5251 drives a stable 12-MHz BBP_CLOCK (CLKIN) to the CP3SP33.
12. The Bluetooth baseband processor on the CP3SP33 now directly controls the RF interface pins and drives the logic levels required during the power-up phase. When the RFCE pin is driven high, the LMX5251 switches from “power-up” to “normal” mode and disables the internal pullup/pulldown resistors on its RF interface inputs.
13. In “normal” mode, the oscillator of the LMX5251 is controlled by the RFCE signal. Driving RFCE high enables the oscillator, and the LMX5251 drives its BBP_CLOCK (CLKIN) output.

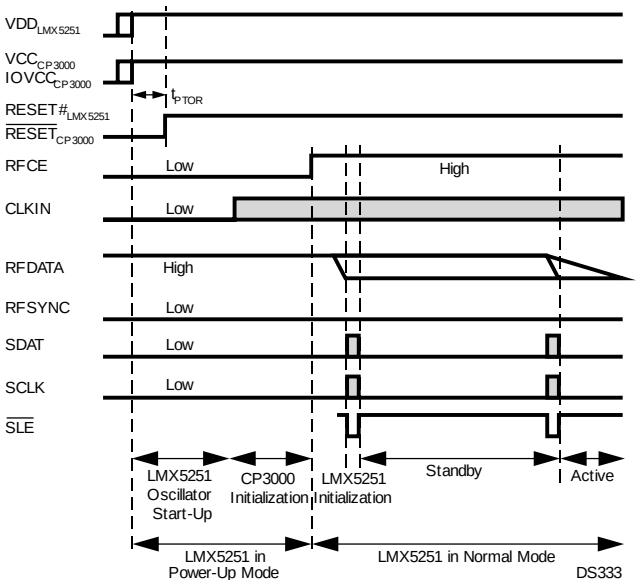


Figure 19-9. LMX5251 Power-Up Sequence

19.4 LMX5252 Power-Up Sequence

A Bluetooth system based on the CP3SP33 and LMX5252 devices has the following states:

- **Off**—When the LMX5252 enters Off mode, all configuration data is lost. In this state, the LMX5252 drives BPOR low.
- **Power-Up**—When the power supply is on and the LMX5252 RESET# input is high, the LMX5252 starts up its crystal oscillator and enters Power-Up mode. After the crystal oscillator is settled, the LMX5252 sends four clock cycles on BRCLK (CLKIN) before driving BPOR high.
- **RF Init**—The baseband controller on the CP3SP33 now drives RFCE high and takes control of the crystal oscillator. The baseband performs all the needed initialization (such as writing the registers in the LMX5252 and crystal oscillator trim).
- **Idle**—The baseband controller on the CP3SP33 drives RFDATA low when the initialization is ready. The LMX5252 is now ready to start transmitting, receiving, or enter Sleep mode.
- **Sleep**—The LMX5252 can be forced into Sleep mode at any time by driving RFCE low. All configuration settings are kept, only the Bluetooth low power clock is running (B3k2).
- **Wait XTL**—When RFCE goes high, the crystal oscillator becomes operational. When it is stable, the LMX5252 enters Idle mode and drives BRCLK (CLKIN).

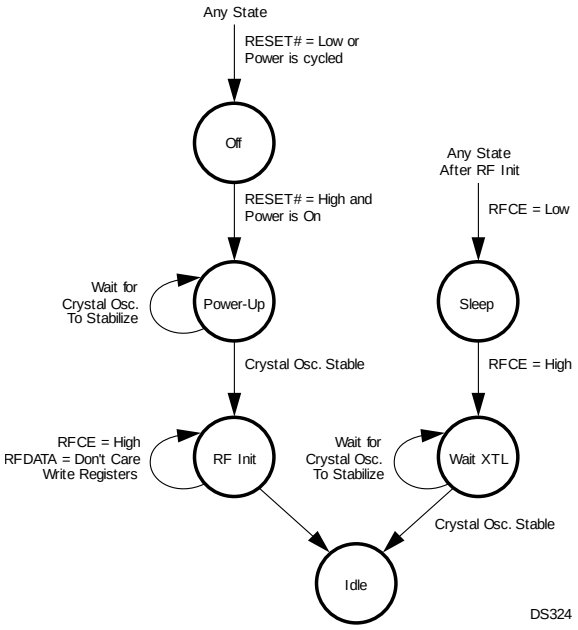


Figure 19-10. LMX5252 Power States

The power-up sequence for a Bluetooth system based on the CP3SP33 and LMX5252 devices is shown in Figure 19-11.

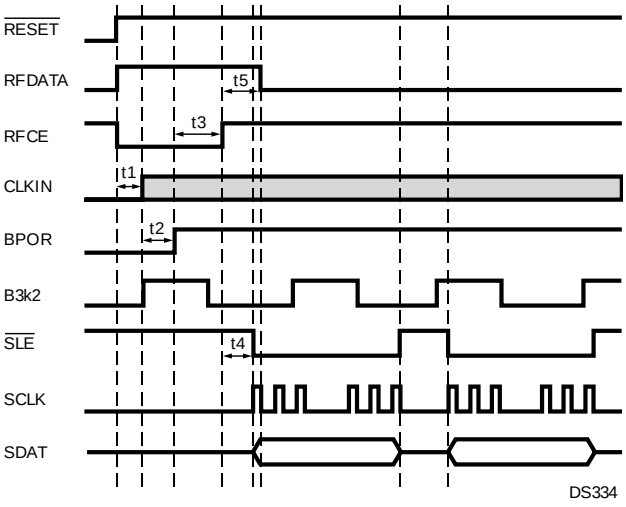


Figure 19-11. LMX5252 Power-Up Sequence

19.5 Bluetooth Sleep Mode

The Bluetooth controller is capable of putting itself into a sleep mode for a specified number of Slow Clock cycles. In this mode, the controller clocks are stopped internally. The only circuitry which remains active are two counters (counter N and counter M) running at the Slow Clock rate. These counters determine the duration of the sleep mode.

The sequence of events when entering the LLC sleep mode is as follows:

1. The current Bluetooth counter contents are read by the CPU.
2. Software “estimates” the Bluetooth counter value after leaving the sleep mode
3. The new Bluetooth counter value is written into the Bluetooth counter register.

4. The Bluetooth sequencer RAM is updated with the code required by the Bluetooth sequencer to enter/exit Sleep mode.
5. The Bluetooth sequencer RAM and the Bluetooth LLC registers are switched from the HCLK Clock domain to the local 12 MHz Bluetooth clock domain. At this point, the Bluetooth sequencer RAM and Bluetooth LLC registers cannot be updated by the CPU, because the CPU no longer has access to the Bluetooth LLC.
6. Hardware Clock Control (HCC) is enabled, and the CP3SP33 enters a low-power mode (Power Save or Idle mode). While in Power Save mode, the Slow Clock drives the HCLK Clock. While in Idle mode, the HCLK Clock is turned off.
7. The Bluetooth sequencer checks if HCC is enabled. If HCC is enabled, the sequencer asserts HCC to the PMM. On the next rising edge of the low-frequency clock, the 1MHz clock and the 12 MHz clock are stopped locally within the Bluetooth LLC. At this point, the Bluetooth sequencer is stopped.
8. The M-counter starts counting. After M + 1 Slow Clock cycles, the HCC signal to the PMM is deasserted.
9. The PMM restarts the 12 MHz Main Clock (and the PLL, if required). The N-counter starts counting. After N + 1 Slow Clock cycles, the Bluetooth clocks (1 MHz and 12 MHz) are turned on again. The Bluetooth sequencer starts operating.
10. The Bluetooth sequencer waits for the completion of the sleep mode. When completed, the Bluetooth sequencer asserts a wake-up signal to the MIWU (see Section 17.0 Section 17).
11. The PMM switches the HCLK Clock to the high-frequency clock and the CP3SP33 enters Active mode again. HCC is disabled. The Bluetooth sequencer RAM and Bluetooth LLC registers are switched back from the local 12 MHz Bluetooth clock to the HCLK Clock. At this point, the Bluetooth sequencer RAM and Bluetooth LLC registers are once again accessible by the CPU. If enabled, an interrupt is issued to the CPU.

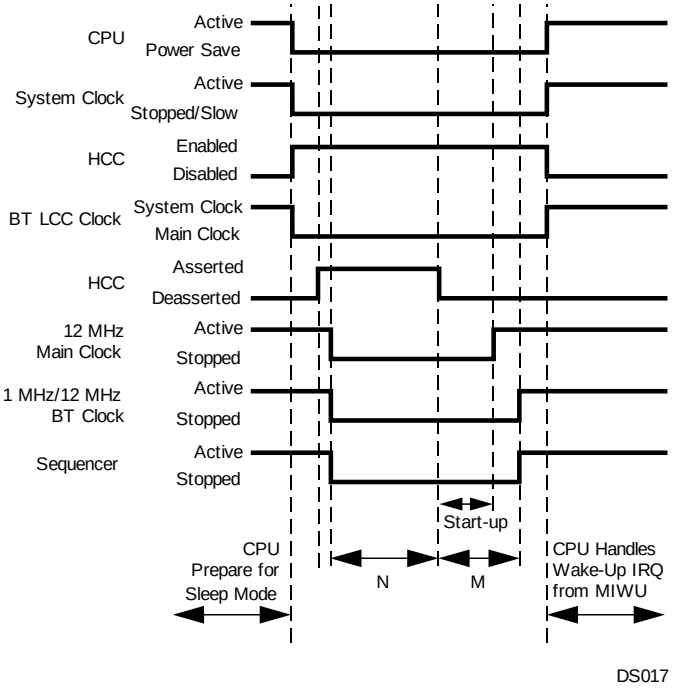


Figure 19-12. Bluetooth Sleep Mode Sequence

19.6 Bluetooth Global Registers

Table 19-2 shows the memory map of the Bluetooth LLC global registers.

Table 19-2. Memory Map of Bluetooth Global Registers

ADDRESS	DESCRIPTION
0001 2080h–0001 20C8h	Global LLC Configuration
0001 20C9h–0001 20FFh	Unused

19.7 Bluetooth Sequencer RAM

The sequencer RAM is a 1K memory-mapped section of RAM that contains the sequencer program. This RAM can be read and written by the CPU in the same way as the Static RAM space and can also be read by the sequencer in the Bluetooth LLC. Arbitration between these devices is performed in hardware.

19.8 Bluetooth Shared Data RAM

The shared data RAM is a 7K memory-mapped section of RAM that contains the link control data, RF programming look-up table, and the link payload. This RAM can be read and written in the same way as the Static RAM space and can also be read by the sequencer in the Bluetooth LLC. Arbitration between these devices is performed in hardware. Table 19-3 shows the memory map of the Bluetooth LLC shared Data RAM.

Table 19-3. Memory Map of Bluetooth Shared RAM

ADDRESS	DESCRIPTION
0001 2500h–0001 26D9h	RF Programming Look-up Table
0001 26DAh–0001 26FFh	Unused
0001 2700h–0001 273Fh	Link Control 0
0001 2740h–0001 277Fh	Link Control 1
0001 2780h–0001 27BFh	Link Control 2
0001 27C0h–0001 27FFh	Link Control 3
0001 2800h–0001 283Fh	Link Control 4
0001 2840h–0001 287Fh	Link Control 5
0001 2880h–0001 28BFh	Link Control 6
0001 28C0h–0001 28FFh	Link Control 7
0001 2900h–0001 36FFh	Link Payload 0–6

20 Telematics Codec

The telematics codec provides dual input channels for voice recognition, telematics, voice-over-IP (VoIP), and Bluetooth applications, and it provides high-quality stereo output for music playback and voice DAC functions.

The ADC has these features:

- Two differential or single-ended analog microphone input
- Two digital microphone interfaces
- Programmable microphone gain and muting
- SNR of 70 dB for ADC path
- Fixed 125× oversampling rate
- 8 kHz to 24 kHz
- Selectable choice of high-pass filters

The stereo DAC has these features:

- >85 dB SNR
- Selectable mono and stereo modes

- Exact sample rates of 8, 12, 16, 24, and 48 kHz derived from 12-MHz clock
- Exact sample rates of 22.05, 24, 32, 44.1, 48, 88.2, 96, 176.4, and 192 kHz from on-chip PLL
- 125× oversampling rate mode using 2, 3, 4, 6, or 12 MHz clock
- 128× oversampling rate mode
- 64× and 32× oversampling rate modes for high-quality 96/192 kHz audio
- On-chip fully differential signaling for analog circuitry to ensure high PSRR and low crosstalk
- Mute and low-power modes on outputs
- Programmable sidetone from ADC
- Click and pop reduction and DC protection circuits
- Zero-crossing detection for mode changing
- Power-management sequencer to protect external components (such as speakers) from power fluctuations
- Programmable 5-tap FIR filter for tone control and compensation

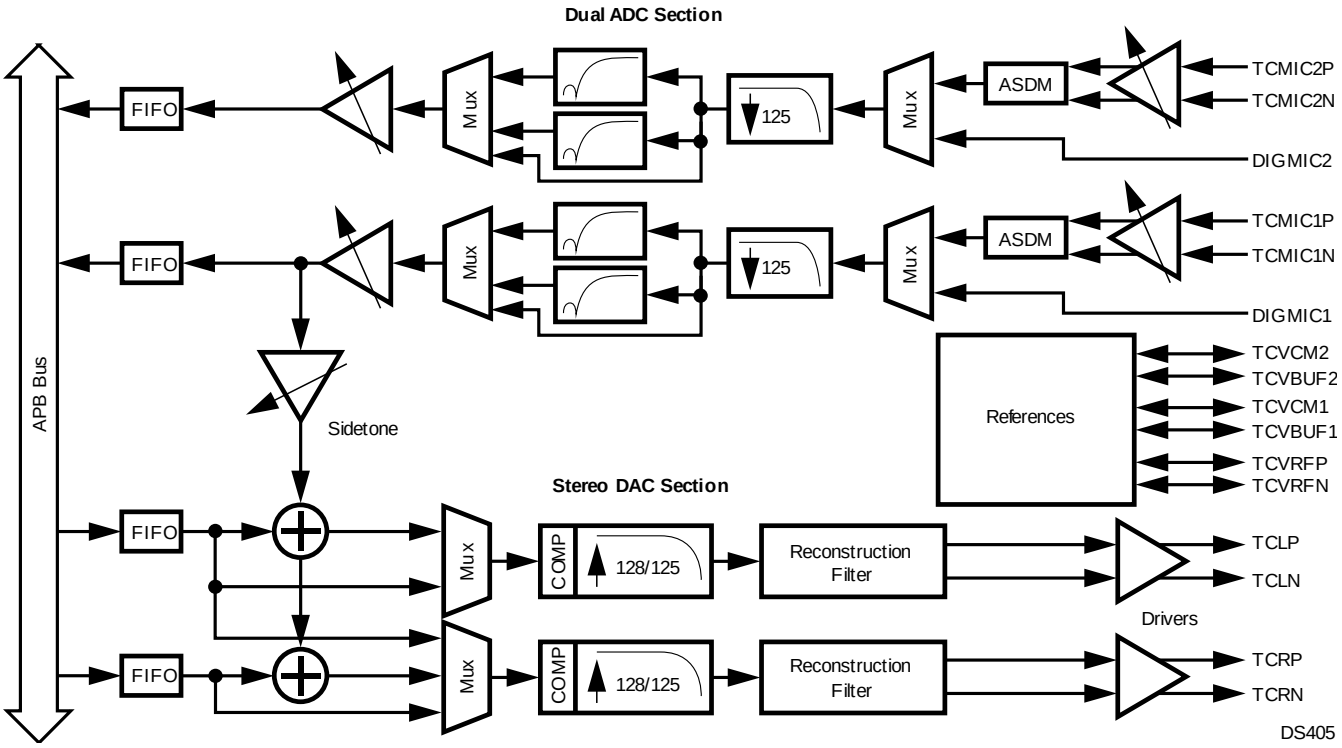


Figure 20-1. Telematics Codec

20.1 CODEC ADC

The ADCs are designed for operation at 8 or 16 kHz and meet all typical specifications at this level. They can be used at sample rates up to 24 kHz by increasing the input clock from 1 MHz up to 3 MHz. The ADCs are 125×-oversampled, second-order, analog sigma-delta modulator (ASDM) designed specifically for audio applications. They use IIR filters to obtain large stopband attenuations. The frequency response matches that used for Bluetooth and GSM at 8 kHz, and it is suitable for G.722 applications at 16 kHz.

The ADCs have selectable high-pass filters to support pop/wind rejection at higher sampling rates. The clock should be set as shown in Table 20-1.

Table 20-1. Codec ADC Clock Frequency

OVERSAMPLING RATE	SAMPLING FREQUENCY (kHz)	CODEC ADC CLOCK FREQUENCY (MHz)
125	8	1
125	12	1.5
125	16	2
125	24	3

20.2 CODEC DAC

The stereo DAC operates in one of four oversampling modes (32×, 64×, 125×, or 128×), as selected in the DACOSR field of the TCDCBASIC register. This allows a 12-MHz clock for 8, 16, 24, or 48 kHz audio or an external clock for any other sampling rate up to 192 kHz as shown in Table 20-2.

Table 20-2. Codec DAC Clock Frequency

DACOSR FIELD	OVERSAMPLING RATE	SAMPLING FREQUENCY (kHz)	CODEC DAC CLOCK FREQUENCY (MHz)
0	125	8	2
0	125	12	3
0	125	16	4
0	125	24	6
0	125	32	8
0	125	48	12
1	128	22.05	5.6448
1	128	32	8.192
1	128	44.1	11.2896
1	128	48	12.288
10	64	88.2	11.2896
10	64	96	12.288
11	32	176.4	22.5792
11	32	192	24.576

The codec stereo DAC is a CD-quality third-order “bit-stream” DAC similar to those found on most commercial CD players. The stop band attenuation is greater than 76 dB and system pass-band ripple can be kept to within 0.01 dB. Typical SNRs are 85dB without a weighting filter.

To meet the requirements of typical automotive telematics applications, the analog signal path is fully differential. For best audio performance, the DAC should operate synchronously to the CPU.

20.3 Compensation Filter

To allow compensation of roll-off in the DAC and analog filter sections, an FIR compensation filter is applied to the input data at the original sample rate. The filter can also be used for precise digital gain and simple tone controls, although a DSP or CPU should be used if more powerful tone control is required. Figure 20-2 shows the filter stages.

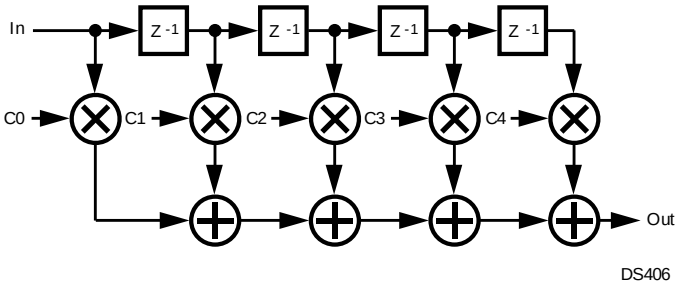


Figure 20-2. Compensation FIR Filter

By default, the filter applies about 2 dB of compensation at 20 kHz. Five taps is sufficient to allow passband equalization and ripple cancellation to around ±0.01 dB.

Because the filter must always be phase linear, the coefficients are symmetrical. Coefficients C0, C1, and C2 are programmable, C3 is equal to C1, and C4 is equal to C0. The maximum power of this filter must not exceed that of the examples given in Table 20-3.

Table 20-3. Default Filter Coefficients

SAMPLING FREQUENCY (kHz)	OVER-SAMPLING RATE	C0	C1	C2	C3	C4
48	125	343	–2291	26984	–2291	343
48	128	61	–371	25699	–371	61

20.4 Reconstruction Filter

The output of the stereo DAC is passed through low-pass reconstruction filters to remove quantization noise (the “sigma-delta tail”). Because the DAC is optimized for high-quality audio (44.1 kHz or higher sample rate), the cutoff frequency of the filter is placed at 24 kHz.

At low sample rates (for example, 8 kHz), quantization noise will appear below this cutoff frequency. If the application requires this noise to be removed, several solutions are possible. For example, low-pass filters may be added to the DAC outputs. Even simple RC filters will provide substantial noise reduction. Audio devices intended for voice (such as headsets) often provide an effective filter because of their limited bandwidth.

In software, digital interpolation may be used to upsample the data for a higher sample rate. This solution requires no additional hardware, but it consumes more CPU bandwidth and adds delay to the audio path.

20.5 Peripheral Bus Interface

The codec is a slave device on the shared audio peripheral APB bus. As shown in Figure 20-1, the codec interface has four FIFOs, two read FIFOs for the two ADC channels and two write FIFOs for the stereo DAC channels. Four independent DMA requests are available corresponding to the four channels. One interrupt request (IRQ28) is available for triggering by any channel.

20.5.1 FIFOs

The ADC and DAC trigger depths are controlled independently to support differing sampling rates. The full depth of the FIFOs does not need not be used, and a lesser depth may be used to reduce latency.

Each ADC FIFO remains in the same state when its ADC is disabled and is only cleared during a reset or flush condition. The ADC FIFOs should be flushed before the ADCs are unmuted to clear any old samples, or the codec can be placed in reset using the Soft Reset bit (SFTRST bit in the TCDCDEBUG register). This can be used to start the codec from a known state.

The DAC FIFOs are automatically flushed when the DAC is disabled. When the DAC is muted, the DAC FIFOs should be flushed before the DAC is unmuted.

20.5.2 Interrupts

Interrupt requests on IRQ28 can be generated from several sources, such as FIFO overrun/underrun, power-cycle events, and audio signal zero-crossings and clipping events (useful for changes to the DSP or audio stream). The interrupt enable register (TCDCIRQEN) and FIFO trigger control register (TCDCFIFO) are used to enable events which assert interrupt requests. The pending/clear register (TCDCIRQPNDCLR) indicates which events have occurred. Writing 1 to a bit in the TCDCIRQPNDCLR register clears the event.

Because the interrupt request can be handled by either the CPU or DSP interrupt controllers, the interrupt request is passed through the Audio Subsystem Controller (ASC), which is programmed to route it to the interrupt controller. To use the codec interrupt, it must be enabled at three levels:

- Codec module—the interrupt must be enabled in the TCDCIRQEN register.
- *ASC module*—the codec interrupt is assigned to ASC interrupt channel 8, so bit 8 in the ASCINTSEL register must be clear to select the CPU interrupt controller or set to select the DSP interrupt controller.
- *Interrupt controller*—the interrupt request must be enabled in the interrupt controller. For the CPU interrupt controller, this is interrupt request IRQ28.

20.5.3 DMA

A DMA request or interrupt request may be enabled when a FIFO reaches its trigger level. Low trigger levels are recommended for triggering DMA requests, to reduce unnecessary latency. DMA requests and IRQ28 interrupt requests are mutually exclusive, however interrupt requests IRQ49, IRQ48, IRQ47, and IRQ46 may be enabled for DMA requests on channels 10, 11, 12, and 13, respectively.

Because the codec DMA requests can be handled by either the CPU or DSP DMA controllers, the DMA requests are passed through the Audio Subsystem Controller (ASC), which is programmed to route them to the DMA controller. To use the codec DMA requests, they must be enabled at three levels:

- *Codec module*—the codec DMA requests must be enabled in the TCDCFIFO register.
- *ASC module*—the codec DMA requests are assigned to ASC DMA channels 10 (ADC1), 11 (ADC2), 12 (left DAC), and 13 (right DAC). These channels are controlled by bits 10, 11, 12, and 13 in the ASCDMASEL0 register. The bits must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be enabled in the ASCDDMASELn registers.
- *DMA controller*—the DMA requests must be enabled in the DMA controller. For the CPU DMA controller, these are DMA requests 10 (ADC1), 11 (ADC1), 12 (left DAC), and 13 (right DAC).

If a DMA request is disabled while it is asserted, the request will remain asserted, which blocks lower-priority requests to the DMA controller. To avoid this situation, software can either check that the DMA request is not asserted before disabling the request, or it can disable the request in the DMA controller.

20.6 Freeze Mode

When Freeze mode is entered, the codec will exhibit the following behavior:

- The contents of the ADC and DAC FIFOs and the FIFO output data do not change.
- The DAC outputs are muted.

20.7 Reset

The ADC, DACs, DSP, and analog modules should not be reset asynchronously in normal applications, because this may damage external electro-mechanical devices such as speakers.

If a watchdog reset occurs, the codec's integral power management will not power down the analog circuitry in a safe manner. When software needs to reset the codec, it should set and then clear the SFTRST bit in the TCDCDEBUG register.

20.8 DC Protection Monitor

If the analog outputs are used to drive a loudspeaker differentially, then DC offsets on the outputs can damage the drivers. If the DAC FIFO output has not changed for a programmable number of samples, the DC protection monitor triggers a DAC FIFO flush. If the number of DC protection monitor events exceeds a programmable limit, the monitor can also shut down the stereo DAC. If the DAC is not being used for audio applications and a constant DC output is required, the monitor can be disabled.

20.9 Sidetone Injection

Sidetone injection from the ADC1 channel can be enabled individually for each DAC channel. The level of sidetone attenuation is programmable in the SIDETONEATTEN field of the TCDCDSP register from 0 to –36 dB in increments of 3 dB. If the sidetone signal for either channel causes clipping, a warning flag is set (STCLPL or STCLPR in the TCDCDSP register).

If the sidetone is used, ADC1 and the stereo DAC must run from the same clock at the same sample and oversampling rate. This can be enabled by setting the CLKTIE bit in the TCDCADC1CLK register and selecting a DAC oversampling rate of 125.

20.10 Telematics Codec Register Set

Table 20-4 lists the telematics codec registers.

Table 20-4. Telematics Codec Registers

NAME	ADDRESS	DESCRIPTION
TCDCBASIC	FF 4400h	Codec Basic Configuration Register
TCDCDACSTATUS	FF 4404h	Codec DAC Status Register
TCDCADCSTATUS	FF 4408h	Codec ADC Status Register
TCDCDSP	FF 440Ch	Codec DSP Set-Up Register
TCDCADCANA1	FF 4410h	Codec ADC1 Analog Control Register
TCDCADCANA2	FF 4414h	Codec ADC2 Analog Control Register
TCDCADC1CLK	FF 4418h	Codec ADC1 Clock Register
TCDCADC2CLK	FF 441Ch	Codec ADC2 Clock Register
TCDCDACCLK	FF 4420h	Codec DAC Clock Register
TCDCFIFO	FF 4424h	Codec FIFO Trigger Control Register
TCDCIRQEN	FF 4428h	Codec Interrupt Enable Register
TCDCIRQPNDCLR	FF 442Ch	Codec Interrupt Pending/Clear Register
TCDCCOMPC0	FF 4430h	Codec Compensation Filter C0/4 Tap Register
TCDCCOMPC1	FF 4434h	Codec Compensation Filter C1/3 Tap Register
TCDCCOMPC2	FF 4438h	Codec Compensation Filter C2 Tap Register
TCDCADC1	FF 4448h	Codec ADC1 Data Register
TCDCADC2	FF 444Ch	Codec ADC2 Data Register
TCDCLEFT	FF 4450h	Codec Left-Channel DAC Data Register
TCDCRIGHT	FF 4454h	Codec Right-Channel DAC Data Register
TCDCDEBUG	FF 443Ch	Codec Debug Register
TCDCMONITOR	FF 4458h	Codec Monitor Control and Status Register

20.10.1 Codec Basic Configuration Register (TCBCBASIC)

The TCDCBASIC register is a 16-bit read/write register that controls the basic operation of the codec. At reset, this register is cleared to 0000h.

7	6	5	4	2	1	0
MUTEL	DACOSR		DACSTMODE		ADC2ON	ADC1ON

15	14	13	12	11	10	9	8
DFS	AFS	FLSDAC	FLSADC	Reserved		SEM	MUTER

ADC1ON — The ADC1 On bit controls whether ADC1 is enabled to run. When written with 1, ADC1 begins its power-up sequence. Check the ADC1UP bit in the TCDCIRQPNDCLR register to determine whether ADC1 has completed its power-up sequence.
0 – ADC1 disabled.
1 – ADC1 enabled.

ADC2ON — The ADC2 On bit controls whether ADC2 is enabled to run. When written with 1, ADC2 begins its power-up sequence. Check the ADC2UP bit in the TCDCIRQPNDCLR register to determine whether ADC2 has completed its power-up sequence.
0 – ADC2 disabled.
1 – ADC2 enabled.

DACSTMODE — The DAC Sidetone Mode field controls the basic function of the DAC, sidetone input, and output stage. The sidetone is taken from ADC1.

DACSTMODE FIELD	LEFT CHANNEL	RIGHT CHANNEL
000	Off	Off
001	Left	Right
010	Left	Off
011	Left + Sidetone	Off
100	Left + Sidetone	Left
101	Left + Sidetone	Right
110	Left	Right + Sidetone
111	Left + Sidetone	Right + Sidetone

DOCOSR — The DAC Oversampling Rate field selects 32×, 64×, 125×, or 128× oversampling rate for the stereo DAC. Note that the ADCs have a fixed oversampling rate of 125×.
00 – 125×
01 – 128×
10 – 64×
11 – 32×

MUTEL — The Mute Left-Channel DAC bit controls whether the left-channel DAC will be muted after the next zero-crossing occurs.
0 – Left-channel DAC mute off.
1 – Left-channel DAC mute enabled.

MUTER — The Mute Right-Channel DAC bit controls whether the right-channel DAC will be muted after the next zero-crossing occurs.
0 – Right-channel DAC mute off.
1 – Right-channel DAC mute enabled.

SEN — The Status Enable bit controls the clock for the bus interface. When the bus interface is driven with a clock, the power-up and power-down interrupts for the ADCs and DACs may be used.
0 – Bus interface clocks disabled.
1 – Bus interface clocks enabled.

FLSADC — The Flush ADC FIFO bit can be used to flush the ADC FIFOs. Writing 1 to this bit clears the FIFOs.
0 – No effect.
1 – Flush the ADC FIFOs.

FLSDAC — The Flush DAC bit can be used to flush the stereo DAC FIFOs. Writing 1 to this bit clears the FIFOs. This should be used when deliberately pausing the DAC data stream or any other time when the DAC FIFOs will be underrun deliberately. If the DAC FIFOs are underrun and this bit is not set, then the DAC is continuously driven with the oldest data and this will result in a DC component on the DAC outputs.
0 – No effect.
1 – Flush the DAC FIFOs.

AFS — The ADC FIFO Status Mode bit controls the encoding of the ADC1FIFO and ADC2FIFO fields in the TCDCADCSTATUS register. See the description of the TCDCADCSTATUS re ister for more information.

DFS — The DAC FIFO Status Mode bit controls the encoding of the LEFTFIFO and RIGHTFIFO fields in the TCDCDACSTATUS register. See the description of the TCDCDACSTATUS register for more information.

20.10.2 Codec DAC Status Register (TCDCDACSTATUS)

The TCDCDACSTATUS register is a 16-bit, read-only register that indicates the status of the DAC FIFOs and the DAC analog output stage. After DAC operation is enabled in the TCDCDACCLK and TCDCBASIC registers, the DACSTATUS bit will become set to indicate that the DAC is operational. The left and right DAC FIFOs are clear while the DAC is disabled. While the DAC is enabled, the FIFOs can be flushed by using the FLSDAC bit in the TCDCBASIC register or by resetting the codec using the Soft Reset bit (SFTRST bit in the TCDCDEBUG register). The DAC FIFOs should be flushed before the DACs are unmuted to clear any old samples. After reset, this register is initialized to 1010h, which indicates the DAC FIFOs contain no valid words and the analog output stage is powered down.

15	14 – 13	12 – 8	7 – 5	4 – 0
DACSTATUS	Reserved	RIGHTFIFO	Reserved	LEFTFIFO

LEFTFIFO — The Left-Channel DAC FIFO Status field indicates the current status of the FIFO for the left-channel DAC. After reset, the default mode for this field is to report the number of empty words available in the FIFO, so it indicates 10h (empty). By setting the DFS bit in the TCDCBASIC register, this indication is reversed and the field reports the number of valid words of data, as shown below:

FIFO STATUS FIELD	DFS BIT	DESCRIPTION
00h	0	Full (16 valid words)
01h to 0Fh	0	15 to 1 valid words
10h	0	Empty (no valid data)
1Eh	0	Underrun error (read while empty)
1Fh	0	Overrun error (written while full)
00h	1	Empty (no valid data)
01h to 0Fh	1	1 to 15 valid words
10h	1	Full (16 valid words)
1Eh	1	Underrun error (read while empty)
1Fh	1	Overrun error (written while full)

RIGHTFIFO The Right-Channel DAC FIFO Status field indicates the current status of the FIFO for the right-channel DAC. After reset, the default mode for this field is to report the number of empty words available in the FIFO, so it indicates 10h (empty). By setting the DFS bit in the TCDCBASIC register, this indication is reversed and the field reports the number of valid words of data. The encoding of the field is the same as that shown for the LEFTFIFO field.

DACSTATUS The DAC Status bit indicates whether the DAC output stages are enabled, which may occur due to remaining charge on the bypass capacitor. Software must check that this bit has cleared before entering disabling the clock to the codec (or entering a low-power mode that disables the clock).
0 – DAC off.
1 – DAC on.

20.10.3 Codec ADC Status Register (TCDCADCSTATUS)

The TCDCADCSTATUS register is a 16-bit, read-only register that indicates the current status of the ADC FIFOs, ADCs, and ADC analog input stage. After ADC operation is enabled in the TCDCADCCLKn and TCDCBASIC registers, the corresponding ADCnSTATUS bits will become set to indicate that the ADCs are operational. The ADC FIFOs are unchanged while the ADCs are disabled. While the ADCs are enabled, the FIFOs can be flushed by using the FLSADC bit in the TCDCBASIC register or by resetting the codec using the Soft Reset bit (SFTRST bit in the TCDCDEBUG register). After reset, this register is initialized to 1010h, which indicates the ADC FIFOs contain no valid words and the analog input stage is powered down.

15	14	13	12 8	7 5	4 0
ADC2STATUS	ADC1STATUS	Reserved	ADC2FIFO	Reserved	ADC1FIFO

ADC1FIFO The ADC1 FIFO Status field indicates the current status of the ADC1 FIFO. After reset, the default mode for this field is to report the number of empty words in the FIFO, so it indicates 10h (empty). By setting the AFS bit in the TCDCBASIC register, this indication is reversed and the field reports the number of valid words of data, as shown below:

ADC FIFO STATUS FIELD	AFS BIT	DESCRIPTION
00h	0	Full (16 valid words)
01h to 0Fh	0	15 to 1 valid words
10h	0	Empty (no valid data)
1Eh	0	Underrun error (read while empty)
1Fh	0	Overrun error (written while full)
00h	1	Empty (no valid data)
01h to 0Fh	1	1 to 15 valid words
10h	1	Full (16 valid words)
1Eh	1	Underrun error (read while empty)
1Fh	1	Overrun error (written while full)

ADC2FIFO The ADC2 FIFO Status field indicates the current status of the ADC2 FIFO. After reset, the default mode for this field is to report the number of valid words in the FIFO, so it indicates 10h (empty). By setting the AFS bit in the TCDCBASIC register, this indication is reversed and the field reports the number of valid words of data. The encoding of the field is the same as that shown for the ADC1FIFO field.

ADC1STATUS The ADC1 Status bit indicates whether ADC1 is enabled. If ADC1 is enabled, its clock must not be disabled.
0 – ADC1 disabled.
1 – ADC1 enabled.

ADC2STATUS The ADC2 Status bit indicates whether ADC2 is enabled. If ADC2 is enabled, its clock must not be disabled.
0 – ADC2 disabled.
1 – ADC2 enabled.

20.10.4 Codec DSP Set-Up Register (TCDCDSP)

The TCDCDSP register is a 16-bit, read/write register that configures some of the DSP portions of the module. After reset, this register is cleared to 0000h.

15	14	13	12 9	8	7 5	4	3	2	1	0
Reserved	STCLPR	STCLPL	SIDETONEATTEN	CLKPH	DIGMICGAIN	CUST COMP	ADC2 DITOFF	ADC1 DITOFF	DAC DITOFF	DAC DITON

DACDITON The DAC Dither On bit enables dithering the output, without regard to the audio content.
0 – Dithering may be on or off.
1 – Dithering always on.

DACDITOFF The DAC Dither Off bit disables dithering the output, without regard to the audio content. If set, this bit overrides the DACDITON bit.
0 – Dithering may be on or off.
1 – Dithering always off.

ADC1DITON The ADC1 Dither Off bit disables dithering the ADC1 ASDM input.
0 – Dithering may be on or off.
1 – Dithering always off.

ADC2DITON The ADC2 Dither Off bit disables dithering the ADC2 ASDM input.
0 – Dithering may be on or off.
1 – Dithering always off.

CUSTCOMP The CUSTCOMP bit selects between default compensation filter coefficients and custom values in the TCDCCOMP0, TCDCCOMP1, and TCDCCOMP2 registers. Default values vary with the over-sampling rate to keep the reconstruction filter output flat up to 20kHz.
0 – Default coefficients are used.
1 – Custom coefficients are used.

DIGMICGAIN The DIGMICGAIN field selects a scaling factor for the DIGMIC1 input data. It can be used to provide extra gain for small analog signals when the microphone pre-amplifier gain is not sufficient. It is controllable in 2 dB steps between 0 dB and 14 dB.

DIGMICGAIN FIELD	GAIN (db)
000	0
001	2
010	4
011	6
100	8
101	10
110	12
111	14

CLKPH The Clock Phase bit can be used to invert the phase of the output clock for use with non-standard digital microphones.
0 – Normal clock.
1 – Inverted clock.

SIDETONEATTEN The Sidetone Attenuation field specifies the gain used in the sidetone feedback. This should only be used if the DAC and ADC are operating at the same sample rates.

SIDETONEATTEN FIELD	GAIN (db)
0000	Off (Mute)
0001	–36
0010	–33
0011	–30
0100	–28
0101	–26
0110	–24
0111	–22
1000	–20
1001	–18
1010	–15
1011	–12
1100	–9
1101	–66
1110	–3
1111	0

STCLPL The Sidetone Clipping Left Channel bit indicates that the sidetone applied to the left DAC channel is too large, and the signal has clipped. More headroom should be provided in the DAC data or the level of sidetone should be reduced. Write 1 to the bit to clear it.
0 – No clipping has occurred.
1 – Clipping has occurred.

STCLPR The Sidetone Clipping Right Channel bit indicates that the sidetone applied to the right DAC channel is too large, and the signal has clipped. More headroom should be provided in the DAC data or the level of sidetone should be reduced. Write 1 to the bit to clear it.
0 – No clipping has occurred.
1 – Clipping has occurred.

20.10.5 Codec ADC Analog Control Register n (TCDCADCANAn)

The TCDCADCANAn registers are 16-bit, read/write register that configure the analog portions of the ADCs. After reset, these registers are cleared to 0000h.

15	14 13	12	11	10 8	7	6	5 4	3 0
Reserved	HPF	MUTE	MICSEL	Reserved	Reserved	MICMODE	Reserved	MICGAIN

MICGAIN The Microphone Gain field selects the gain of the microphone pre-amp. For a MICGAIN of 0 dB, the input should not exceed 500 mVpp (third harmonic distortion ~ –70 dB). For lower distortion, reduce the maximum input signal by 3 dB, a 350 mVpp input gives a third harmonic at ~ –80 dB. This applies to both differential and single-ended configurations. For a differential input, the peak-to-peak voltage is the difference between the maximum negative voltage on TCMICnN and the maximum positive voltage on TCMICnP. For a single-ended input, the peak-to-peak voltage is the difference between the minimum and maximum voltage on TCMICnP.

MICGAIN FIELD	GAIN (db)	MAX. INPUT (mVpp)	RECOMMENDED INPUT (mVpp)
0	0	500	354
1	1.8	406	287
10	4.1	312	220
11	6	251	177
100	7.8	204	144
101	10.1	156	111
110	12	126	88.8
111	14.5	94.2	66.6
1000	16.1	78.3	55.4
1001	18.1	62.2	44
1010	20.6	46.7	33
1011	22.1	39.3	27.8
1100	24.1	31.2	22
1101	26.6	23.4	16.5
1110	28.2	19.5	13.8
1111	30.1	15.6	11.1

MICMODE The Microphone Mode field configures the analog microphone input.
0 – Differential (TCMICnP and TCMICnN inputs).
1 – Single-ended (TCMICnP input only).

MICSEL The Microphone Select bit selects between the analog and digital microphone inputs. When a digital microphone is selected, the TCIO1 pin is configured as an output clock at 125× oversampling rate, without regard to the GPIO registers. The digital microphone is a four-pin device which will take the ADC clock and produce a sigma-delta stream suitable for the decimator.
0 – Analog input.
1 – Digital input.

MUTE The Mute bit enables muting the microphone input. For ADC1, muting is applied to both analog and digital inputs. For ADC2, only the analog input is muted.
0 – Muting disabled.
1 – Muting enabled.

HPF The High Pass Filter field controls the high-pass filter applied to the output of the ADC, typically used for eliminating wind/road/pop noise. The HPF has two options, to enable the ADC to be used at 8 kHz and 16 kHz sample rates without altering the HPF response.

HPF FIELD	DESCRIPTION	
	8 kHz	16 kHz
00	Bypass	Bypass
01	–0.5dB Point at 300 Hz Notch at 55 Hz (recommended)	–0.5dB Point at 600 Hz Notch at 110 Hz
10	–0.5dB Point at 150 Hz Notch at 27 Hz	–0.5dB Point at 300 Hz Notch at 55 Hz (recommended)
11	Reserved	Reserved

From 16 kHz to 24 kHz the HPF roll-off options will be too high frequency for most applications, so it is recommended that the filter is bypassed above 16 kHz. Simple voice applications rarely require sample rates above 16 kHz. It is recommended that for higher quality applications in which 24 kHz recording is required, a signal conditioning algorithm to remove unwanted LF should be used.

20.10.6 Codec ADC Clock Control Register n (TCDCADCnCLK)

The TCDCADCnCLK registers are 16-bit, read/write registers that configure the ADC clock sources. These registers must only be programmed when the corresponding ADC is disabled. At reset, these registers are cleared to 0000h.

15 12	11	10 8	7 0
Reserved	CLKTIE	ADCCLKSRC	ADCCLKDIV

ADCCLKDIV The ADC Clock Divisor field specifies the ADC clock period in terms of half-cycles of the input clock. The field is biased by 1, so to get a period of 4 half-cycles, load this field with 3. A value of zero disables the clock.

ADCCLKSRC The ADC Clock Source field selects the clock source for the ADC clock divisor. Independent clock sources can be programmed for each ADC and the DAC, to allow each section of the codec to run from different clocks. If the sidetone is used, ADC1 and the DAC must run from the same clock source at the same sample and oversampling rate. Although PLL1 and PLL2 clocks are available as ADC clock sources, it is recommended to use Auxiliary Clock 4 when a timebase derived from a PLL is desired.

ADCCLKSRC FIELD	SOURCE
000	PCLK Clock
001	TCIO1 Input
010	PLL2 Clock
011	I2SCLK Input
100	AAI SCK Input
101	Auxiliary Clock 4
110	PLL1 Clock
111	Clock Disabled

CLKTIE The Clock Tie bit is used to drive the ADC clock from the DAC clock, which causes the ADCCLKDIV and ADCCLKSRC fields to be ignored. Only the input clock source and clock divisor are affected by the CLKTIE bit.
0 – ADC clock independent of DAC clock.
1 – ADC clock driven by DAC clock.

20.10.7 Codec DAC Clock Control Register (TCDCDACCLK)

The TCDCDACCLK register is a 16-bit, read/write register that configures the DAC clock source. This register must only be programmed when the DAC is disabled. At reset, this register is cleared to 0000h.

15 14	13	12 11	10 8	7 0
Reserved	SEL6_144	DACRNG	DACCLKSRC	DACCLKDIV

DACCLKDIV The DAC Clock Divisor field specifies the DAC clock period in terms of half-cycles of the input clock. The field is biased by 1, so to get a period of 4 half-cycles, load this field with 3. A value of zero disables the clock.

DACCLKSRC The DAC Clock Source field selects the clock source for the DAC clock divisor. Independent clock sources can be programmed for each ADC and the DAC, to allow each section of the codec to run from different clocks. If the sidetone is used, ADC1 and the DAC must run from the same clock source at the same sample and oversampling rate. Although PLL1 and PLL2 clocks are available as ADC clock sources, it is recommended to use Auxiliary Clock 4 when a timebase derived from a PLL is desired.

DACCLKSRC FIELD	SOURCE
000	PCLK Clock
001	TCIO1 Input
010	PLL2 Clock
011	I2SCLK Input
100	AAI SCK Input
101	Auxiliary Clock 4
110	PLL1 Clock
111	Clock Disabled

DACRNG The DACRNG field specifies the expected input clock range of the DAC. This is used by the power management unit to ensure correct timing during power up and power down. If the exact clock range is not available, use the next fastest.

DACRNG FIELD	RANGE (MHz)	TYPICAL SAMPLE RATE (kHz)
00	2	8
01	6	24
10	12	48
11	24	96

SEL6_144 The Select 6 MHz bit is used to select the 6 MHz sync to clock the reconstruction filter, rather than the 12 MHz clock.
0 – 12 MHz clock.
1 – 6 MHz sync.

20.10.8 Codec FIFO Trigger Control Register (TCDCFIFO)

The TCDCFIFO register is a 16-bit, read/write register that configures the events which assert DMA and interrupt requests. At reset, this register is cleared to 0000h.

15	14	13	12	11 – 8	7 – 4	3 – 0
DMAR	DMAL	DMAADC2	CMAADC1	DACFIFOTRIG	ADC2FIFOTRIG	ADC1FIFOTRIG

ADC1FIFOTRIG The ADC FIFO Trigger Level field specifies how many empty words are in the ADC1 FIFO when an event is triggered.

ADC2FIFOTRIG The ADC FIFO Trigger Level field specifies how many empty words are in the ADC2 FIFO when an event is triggered.

DACFIFOTRIG The DAC FIFO Trigger Level field specifies how many valid words are remaining in either of the DAC FIFOs when an event is triggered.

DMAADC1 The DMA ADC1 bit enables a DMA request in response to the ADC1 trigger event. If the ADC1FIFO bit in the TCDCIRQEN register is set, the DMAADC1 bit is ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – ADC1 DMA request disabled.
1 – ADC1 DMA request enabled.

DMAADC2 The DMA ADC2 bit enables a DMA request in response to the ADC2 trigger event. If the ADC2FIFO bit in the TCDCIRQEN register is set, the DMAADC2 bit is ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – ADC2 DMA request disabled.
1 – ADC2 DMA request enabled.

DMAL — The DMA Left Channel bit enables a DMA request in response to the left channel DAC FIFO asserting the DAC trigger event. If the LFTFIFO bit in the TCDCIRQEN register is set, the DMAL bit is ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – Left channel DAC DMA request disabled.
1 – Left channel DAC DMA request enabled.

DMAR — The DMA Right Channel bit enables a DMA request in response to the right channel DAC FIFO asserting the DAC trigger event. If the RGTFIFO bit in the TCDCIRQEN register is set, the DMAR bit is ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – Right channel DAC DMA request disabled.
1 – Right channel DAC DMA request enabled.

20.10.9 Codec Interrupt Request Enable Register (TCDCIRQEN)

The TCDCIRQEN register is a 16-bit, read/write register that enables interrupt requests from the codec. The clock to the TCDCIRQPNDCLR register is enabled when any of the fields in this register are set. After reset, this register is cleared to 0000h.

7	6	5	4	3	2	1	0
ADCDOWN	DACUP	ADC2U	ADC1UP	RGTFIFO	LFTFIFO	ADC2FIFO	ADC1FIFO

15 14	13	12	11	10	9	8
Reserved	STCLP	MICCLP	ZXDR	ZXDL	DACDOWN	ADC2DOWN

ADC1FIFO — The ADC1 FIFO bit enables an interrupt when the ADC1 FIFO is filled to the trigger level specified in the TCDCFIFO register, when the FIFO is full, or when the FIFO is empty. Setting the ADC1FIFO bit causes the DMAADC1 bit to be ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – ADC1 FIFO interrupt disabled.
1 – ADC1 FIFO interrupt enabled.

ADC2FIFO — The ADC2 FIFO bit is used to trigger an interrupt when the ADC2 FIFO is filled to the trigger level specified in the TCDCFIFO register, when the FIFO is full, or when the FIFO is empty. Setting the ADC2FIFO bit causes the DMAADC2 bit to be ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – ADC2 FIFO interrupt disabled.
1 – ADC2 FIFO interrupt enabled.

LFTFIFO — The Left DAC FIFO bit enables an interrupt when the left-channel DAC FIFO is emptied to the trigger level specified in the TCDCFIFO register, when the FIFO is full, or when the FIFO is empty. Setting the LFTFIFO bit causes the DMAL bit to be ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – Left-channel FIFO interrupt disabled.
1 – Left-channel FIFO interrupt enabled.

RGTFIFO — The Right DAC FIFO bit enables an interrupt when the right-channel DAC FIFO is emptied to the trigger level specified in the TCDCFIFO register, when the FIFO is full, or when the FIFO is empty. Setting the RGTFIFO bit causes the DMAR bit to be ignored. (Interrupt and DMA requests are mutually exclusive.)
0 – Right-channel FIFO interrupt disabled.
1 – Right-channel FIFO interrupt enabled.

ADC1UP — The ADC1 Power-Up bit enables an interrupt when ADC1 has powered up.
0 – ADC1 power-up interrupt disabled.
1 – ADC1 power-up interrupt enabled.

ADC2UP	The ADC2 Power-Up bit enables an interrupt when ADC2 has powered up. 0 – ADC2 power-up interrupt disabled. 1 – ADC2 power-up interrupt enabled.
DACUP	The DAC Power-Up bit enables an interrupt when the stereo DAC has powered up. 0 – DAC power-up interrupt disabled. 1 – DAC power-up interrupt enabled.
ADC1DOWN	The ADC1 Power-Down bit enables an interrupt when ADC1 has powered down. 0 – ADC1 power-down interrupt disabled. 1 – ADC1 power-down interrupt enabled.
ADC2DOWN	The ADC2 Power-Down bit enables an interrupt when ADC2 has powered down. 0 – ADC2 power-down interrupt disabled. 1 – ADC2 power-down interrupt enabled.
DACDOWN	The DAC Power-Down bit enables an interrupt when the DAC has powered down. 0 – DAC power-down interrupt disabled. 1 – DAC power-down interrupt enabled.
ZXDL	The Zero-Crossing Detector Left Channel bit enables an interrupt when the left-channel reconstruction filter output crosses zero. This is useful if the gain of an external audio device is to be changed. Internal gains are controlled automatically on these events. 0 – Zero-crossing interrupt disabled. 1 – Zero-crossing interrupt enabled.
ZXDR	The Zero-Crossing Detector Right Channel bit enables an interrupt when the right-channel reconstruction filter output crosses zero. 0 – Zero-crossing interrupt disabled. 1 – Zero-crossing interrupt enabled.
MICCLP	The Microphone Clipping bit enables an interrupt when the DIGMIC1 input gain stage clips. 0 – Microphone clipping interrupt disabled. 1 – Microphone clipping interrupt enabled.
STCLP	The Sidetone Clipping bit enables an interrupt when either sidetone feed to the DACs clip. 0 – Sidetone clipping interrupt disabled. 1 – Sidetone clipping interrupt enabled.

20.10.10 Codec Interrupt Pending/Clear Register (TCDCIRQPNDCLR)

The TCDCIRQPNDCLR register is a 16-bit, read/write register that indicates the state of pending interrupt requests. The bits in this register correspond to the interrupt enable bits in the TCDCIRQEN register. A set bit in the TCDCIRQPNDCLR register indicates that the request is enabled and asserted. The TCDCIRQPNDCLR register is also used by the interrupt service routine to clear these requests. Writing 1 to a bit in this register clears the request. Writing 0 has no effect. After reset, this register is cleared to 0000h.

7	6	5	4	3	2	1	0
ADC1DOWN	DACUP	ADC2UP	ADC1UP	RGTFIFO	LFTFIFO	ADC2FIFO	ADC1FIFO

15	14	13	12	11	10	9	8
Reserved		STCLP	MICCLP	ZXDR	ZXDL	DACDOWN	ADC2DOWN

20.10.11 Codec Compensation Filter C0/4 Tap Register (TCDCCOMP0)

The TCDCCOMP0 register is a 16-bit, read/write register that specifies the C0 and C4 coefficients of the compensation filter, when the CUSTCOMP bit of the TCDCDSP register is set. When the CUSTCOMP bit is clear, the TCDCCOMP0 register is ignored. After reset, this register is cleared to 0000h

15	1
COMPC0	

20.10.12 Codec Compensation Filter C1/3 Tap Register (TCDCCOMP1)

The TCDCCOMP1 register is a 16-bit, read/write register that specifies the C1 and C3 coefficients of the compensation filter, when the CUSTCOMP bit of the TCDCDSP register is set. When the CUSTCOMP bit is clear, the TCDCCOMP1 register is ignored. After reset, this register is cleared to 0000h

15	1
COMPC1	

20.10.13 Codec Compensation Filter C2 Tap Register (TCDCCOMP2)

The TCDCCOMP2 register is a 16-bit, read/write register that specifies the C2 coefficient of the compensation filter, when the CUSTCOMP bit of the TCDCDSP register is set. When the CUSTCOMP bit is clear, the TCDCCOMP2 register is ignored. After reset, this register is cleared to 0000h

15	1
COMPC2	

20.10.14 Codec ADC Data Register n (TCDCADCn)

The TCDCADCn registers are 16-bit, read-only registers used to unload a word from the corresponding ADC FIFOs. After reset, these registers are cleared to 0000h

15	1
ADCDATA	

20.10.15 Codec Left Channel DAC Data Register (TCDCLEFT)

The TCDCLEFT register is a 16-bit, read/write register used to load a word into the left-channel DAC FIFO. After reset, this register is cleared to 0000h

15	1
LEFTDATA	

20.10.16 Codec Right Channel DAC Data Register (TCDCRIGHT)

The TCDCRIGHT register is a 16-bit, read/write register used to load a word into the right-channel DAC FIFO. After reset, this register is cleared to 0000h

15	1
RIGHTDATA	

20.10.17 Codec Debug Register (TCDCDEBUG)

The TCDCDEBUG register is a 16-bit, read/write register used to configure the TCIO1 pin and safely reset the codec. After reset, this register is cleared to 0000h.

15	8	7	6	4	3	0
Reserved		SFTRST	Reserved		GPIO	

GPIO The GPIO field configures the TCIO1 pin (alternate function of PG10).

GPIO FIELD	I/O	DESCRIPTION
0000	Input	Audio input clock at 256× or 512× oversampling rate
0001	Output	ADC1 Clock
0010	Output	ADC2 Clock
0011	Output	DIGMIC1 Clock
0100	Output	DAC Clock
0101	Output	ADC1 Serial Data Stream
0110	Output	ADC2 Serial Data Stream
0111	Output	Left Channel DAC Serial Data Stream
1000	Output	Right Channel DAC Serial Data Stream
1001	Output	Sidetone Left Overflow
1010	Output	Sidetone Right Overflow
1011	Output	DIGMIC Overflow
1100	Output	1
1101	Output	0
>1101	N/A	Reserved

SFTRST The Soft Reset bit is used to reset the codec independently of a system reset. The codec is safely reset by setting and then clearing this bit. After the bit is cleared, configuration registers remain unchanged. The ADC and DAC FIFOs are flushed. Pending interrupts and DMA requests are cleared. The ADCs and DAC, if enabled, enter their start-up routines to ensure no noise (click and pop) is heard.
0 – Normal operation.
1 – Soft reset.

20.10.18 Codec Monitor Control/Status Register (TCDCMONITOR)

The TCDCMONITOR register is a 16-bit, read/write register that configures the DC protection monitor. The monitor ensures that, if the system hangs, harmful DC outputs are not driven to external devices such as speakers. The monitor is enabled by default. When triggered, it will flush the DAC FIFOs. If the SDE bit is set and the number of DC protection events exceeds a programmable limit, the protection monitor will also shut down the stereo DAC. After reset, this register is cleared to 0000h.

15	14	13 … 11	10	9 … 0
Reserved	SDE	SDLIMIT	MD	DCLIMIT

DCLIMIT The DC Limit field specifies the number of times a sample can have the same value before the DC protection monitor is triggered to protect the external device. If the number is less than 16, a default value of 1023 is used.

MD The Monitor Disable bit allows software to disable the monitor. By default, the monitor is enabled until software sets this bit.
0 – Monitor enabled.
1 – Monitor disabled.

SDLIMIT The Shutdown Limit field is used to specify the number of DC protection events which may occur before the stereo DAC is shut down. To clear the shutdown condition, software must respond to the DACDOWN interrupt.

SDE The Shutdown Enable bit enables stereo DAC shutdown when the monitor is triggered. If a shutdown event occurs, the stereo DAC remains shut down until this bit is cleared.
0 – DAC shutdown disabled.
1 – DAC shutdown enabled.

20.11 Usage Notes

The telematics codec is designed for maximum flexibility, this section gives an overview of recommended system setups for different applications.

20.11.1 In-Car Bluetooth Telephony

In this mode, the codec acts as a bridge between a Bluetooth-equipped mobile telephone and the car's head unit. A GSM phone establishes a link and transmits voice-band audio data in the CVSD format, which is well-suited to a noisy environment. The data is received by the Bluetooth LLC and passed to the CVSD decoder. The CVSD converts the data into linear 16-bit samples which are transferred to and from the codec (possibly through the on-chip DSP). The codec then performs the digital-to-analog and analog-to-digital conversion and the analog interfaces with the car radio.

Any differences in sample rate should be across the CVSD link where they pose the least threat to audio quality. The DAC should be used in 8-kHz mode rather than implementing an SRC (Sample Rate Conversion) scheme. The codec should share a clock source with the Bluetooth LLC.

Data: bidirectional mono 8-kHz 16-bit linear format.

Audio clock for DAC: 2.000 MHz. DACOSR = 00

Audio clock for ADC: 1.000 MHz.

20.11.2 I^2S Music Playback

In this example, the ADC is not used and the data is provided from the I^2S interface. The I^2S interface should be configured as a slave to allow the sample rate to be derived from the external source device. If the source sample rate is not 48 kHz, then SRC is performed in the DSP coprocessor. The sample rate of the audio stream must be known or conveyed by other means. Data is most likely passed using interrupt requests. The SRC must cope with the rate conversion and allow for a tolerance in the input rate relative to the output rate, varying the algorithm to match the required output rate by knowing how often data arrives and applying skew on the interpolation/decimation when required. This can be avoided by using a clock synchronous to the audio source and using an oversampling rate other than 125.

Data: Stereo 44.1-kHz 16-bit linear format.

Audio clock for DAC: 12.288 MHz. DACOSR = 01.

20.11.3 Wireless Music Headphones

In this example, the ADC is not used and compressed data is provided from the Bluetooth link. The DSP must then convert this data into 48-kHz, 16-bit stereo audio for use by the DAC.

Data: Stereo 48-kHz 16-bit linear format.

Audio clock for DAC: 12.000 MHz. DACOSR = 00.

20.11.4 Wireless Voice Headset

In this example, the device is configured in the same way as for In-Car Bluetooth Telephony, but the output must drive a headphone amplifier rather than the differential line outputs.

20.11.5 MP3 Playback

This application uses the stereo DAC to playback audio from memory, so a PLL can be used to create an approximate 11.025 MHz clock for 44.1-kHz samples with 12.000 MHz for 48-kHz data.

In this example, the ADC can be used for the typical voice notes feature, 8-kHz recording data in companded format using the CVSD's G.711 circuits to compress the data, rather than using a DSP for compression.

20.11.6 VoIP and 3G Telephony

This application can require G.722 support with sample rates up to 16 kHz in 16-bit mode. The ADC and DAC must be used at these rates with the compression being handled by a DSP. The delay caused by the G.722 algorithm and FIFO depths should be minimized to ensure the group delay specifications are met.

20.12 Tuning the Compensation Filter

For unusual sample rates or where the circuit is used to drive a non-linear audio system, it is straightforward to calculate the required compensation filter coefficients. First, replace all coefficients with zero except C2, which should be replaced with a value of 20000 (to ensure that the output frequency response is not limited by headroom). This eliminates the effect of the compensation filter and a frequency sweep on the input to the DAC will produce the uncompensated response of the system. The inverse of this response can then be used to calculate the required coefficients using the inverse discrete fourier transform:

$$c_n = \frac{\Delta t}{2\pi} \int_0^{(2\pi)/(\Delta t)} H_D(w) \cos \frac{2\pi n w}{w_s} dw \tag{5}$$

In which H_D is the desired response (from ω up to ω_{cutoff}), and n is the tap number. In practice, this requires minor manual optimization of the coefficient LSBs to obtain acceptable ripple across the newly optimized passband.

20.13 Obtaining Maximum DAC SNR

The noise floor of the DAC increases at frequencies above Fs/2 and care should be taken at sample rates below 44.1 kHz, because this reduces SNR. For maximum SNR when the DAC is operated at 8 kHz, a low-pass filter can be used to limit the bandwidth of the analog audio output to 4 kHz before driving a system with wider bandwidth.

If the same DAC channel is used for both 8 kHz and 44.1/48 kHz sample rates, the 8-kHz signal can be interpolated in software to a higher rate. This saves the component cost of the external filter at the expense of group delay.

These modifications are not required in typical Bluetooth applications, because the signal noise floor is far higher than the DAC noise to 20 kHz, and speaker bandwidths are not much wider than 4–6 kHz. It becomes an issue when using digital attenuation on voice band signals before the DAC. This reduces SNR linearly and the relative noise can become audible. It is recommended that the DAC is operated at full dynamic range and attenuation is performed at the last stage in the analog signal path.

21 USB Controller

The dual-role USB controller may be used as either the host or the peripheral in point-to-point communications with another USB device. The USB controller is a peripheral on the CPU core AHB bus. An on-chip USB transceiver supports full-speed (12 Mbps) operation.

The USB controller complies with both the USB 2.0 specification (full-speed mode) and with the On-The-Go supplement to the specification. USB On-The-Go has been introduced to provide a low-cost connectivity solution for consumer devices such as mobile phones, PDAs, digital cameras, and MP3 players. Devices that are solely peripherals initiate transfers through a Session Request Protocol (SRP), while dual-role devices support both SRP and Host Negotiation Protocol (HNP).

The USB controller is configured for up to seven endpoint pipes, one bidirectional pipe for Endpoint 0 and unidirectional transmit and receive pipes for Endpoints 1, 2, and 3. The endpoints can be individually programmed for Bulk/Interrupt or Isochronous transfers. (Whether these endpoints are used for IN transactions or for OUT transactions at any time will depend on whether the device is currently being used as a USB peripheral or as the host for point-to-point communications with another USB peripheral.)

Each endpoint pipe has its own FIFO buffer. Table 21-1 shows the size of each FIFO, which is the maximum packet size which can be used with the corresponding endpoint pipe. In double-buffer mode, the maximum packet size is reduced by a factor of two.

Table 21-1. Endpoint FIFO Buffer Size

ENDPOINT	DIRECTION	FIFO SIZE (bytes)	TRANSFER TYPE
0	Bidirectional	64	Control
1	Transmit	256	Isochronous
1	Receive	256	Isochronous
2	Transmit	64	Bulk/Interrupt
2	Receive	64	Bulk/Interrupt
3	Transmit	64	Bulk/Interrupt
3	Receive	64	Bulk/Interrupt

The USB controller provides all encoding, decoding, and checking needed for sending and receiving USB packets. It interrupts the CPU only when endpoint data has been successfully transferred.

When acting as the host for point-to-point communications, the USB controller maintains a frame counter and automatically schedules SOF, Isochronous, Interrupt, and Bulk transfers. It also includes support for the SRP and HRP protocols for point-to-point communications, defined in the USB On-The-Go specification.

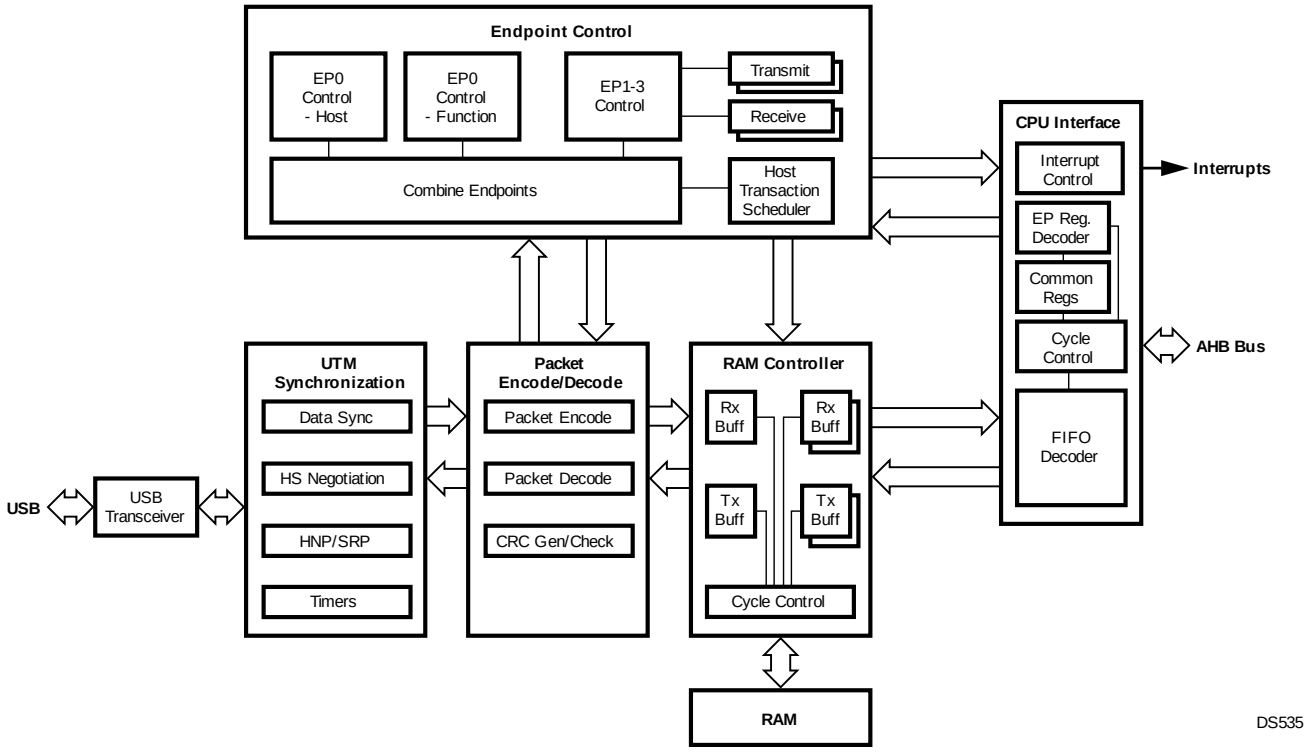


Figure 21-1. USB Controller Block Diagram

21.1 Modes of Operation

The USB controller has two main modes of operation:

- *Peripheral Mode*—the USB controller encodes, decodes, checks, and directs all USB packets sent and received. IN transactions are handled through the device's Tx FIFOs, and OUT transactions are handled through its Rx FIFOs. If a FIFO becomes full during an OUT transaction or empty during an IN transaction, a NAK handshake packet is automatically returned to the host. A NAK packet is also returned for any endpoint that has interrupts disabled. Peripheral mode is used when the device is acting as a peripheral to a standard USB host or as a peripheral in point-to-point communications. Control, Bulk, Isochronous, and Interrupt transactions are supported.
- *Host Mode*—the USB controller acts as the host in point-to-point communication with another USB device. In this mode, it can communicate with another device (but not a hub), that supports Control, Bulk, Isochronous, or Interrupt transactions. IN transactions are handled through the Rx FIFOs, and OUT transactions are handled through the Tx FIFOs. As well as encoding, decoding, and checking the USB packets being sent and received, the USB controller automatically schedules Isochronous endpoints and Interrupt endpoints to perform one transaction every n frames, in which n represents the polling interval that has been programmed for the endpoint. The remaining bus bandwidth is shared equally among the Control and Bulk endpoints. Only one downstream device can be supported in Host mode.

21.2 USB Connector Interface

21.2.1 USB 2.0

A full-speed USB 2.0 interface requires 22Ω series resistors on the D+ and D– data signals, to meet the impedance specification for these signals, as shown in Figure 21-2.

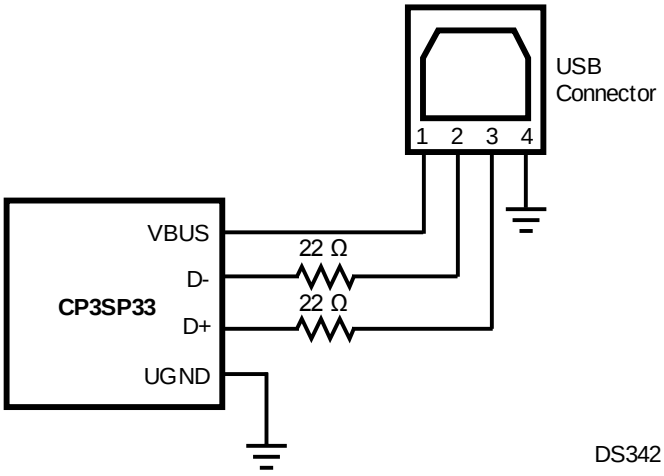


Figure 21-2. USB 2.0 Connector Interface

21.2.2 USB On-The-Go

Additional signals are used to implement USB On-The-Go features, as shown in Figure 21-3.

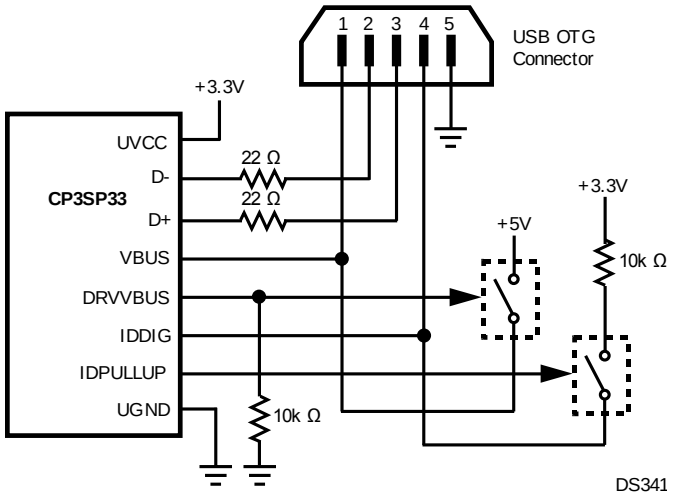


Figure 21-3. USB On-The-Go Connector Interface

For point-to-point communications, the device may also be required to power the VBUS to 5V as the A device of the connection (source of power and default host) or, as the B device (default peripheral), to be able to wake the A device by charging VBUS to 2V. A 100kΩ pulldown resistor on the DRVVBUS output is recommended to ensure that VBUS is not pulled high during reset.

Whether the USB controller initially operates in Host mode or in Peripheral mode depends on whether it is being used in an A device or a B device, which in turn depends on whether the IDDIG input is low or high. When the USB controller is operating as an A device, it is initially configured to operate in Host mode. When operating as a B device, the USB controller is initially configured to operate in Peripheral mode. However, the HOSTREQ bit in the DEVCTL register can be used to request that a B device becomes the Host the next time there is no activity on the USB bus.

The IDDIG input reflects the state of the ID pin of the device's mini-AB receptacle, with IDDIG being low indicating an A plug and operation as an A device, and IDDIG being high indicating a B plug and operation as a B device.

Information on whether the USB controller is acting as an A device or as a B device is indicated in the DEVCTL register, along with information about the level of VBUS relative to the high and low voltage thresholds used to signal Session Start and Session End.

21.3 USB Controller Register Set

The USB controller common registers affect all endpoints. In addition, for each endpoint there is an endpoint-specific register bank. There is an indexed addressing mechanism for accessing the endpoint-specific register banks, and there are also non-indexed images of each register bank. Either the indexed or the non-indexed images may be used to access these registers. The INDEX register selects the register bank accessible in the image at FF 0810h.

Table 21-2 lists the USB controller common registers, Table 21-3 lists the endpoint-specific registers in a bank, and Table 21-4 lists the register banks.

Table 21-2. USB Common Interface Registers

NAME	ADDRESS	DESCRIPTION
FADDR	FF 0800h	Function Address Register
POWER	FF 0801h	Power Management Register
INTRTX	FF 0802h	Interrupt Register for Endpoint 0 and Tx Endpoints 1-3
INTRRX	FF 0804h	Interrupt Register for Rx Endpoints 1-3

Table 21-2. USB Common Interface Registers (continued)

NAME	ADDRESS	DESCRIPTION
INTRTXE	FF 0806h	Interrupt Enable Register for INTRTX
INTRRXE	FF 0808h	Interrupt Enable Register for INTRRX
INTRUSB	FF 080Ah	Interrupt Register for USB Common Interrupts
INTRUSBE	FF 080Bh	Interrupt Enable Register for INTRUSB
FRAME	FF 080Ch	Frame Number Register
INDEX	FF 080Eh	Index Register
EP0FIFO	FF 0820h	Endpoint 0 FIFO
EP1FIFO	FF 0824h	Endpoint 1 FIFO
EP2FIFO	FF 0828h	Endpoint 2 FIFO
EP3FIFO	FF 082Ch	Endpoint 3 FIFO
DEVCTL	FF 0860h	USB OTG Device Control Register
VCTRL	FF 0C00h	USB Transceiver Control Register
VSTATUS	FF 0C02h	USB Transceiver Status Register

Table 21-3. USB Endpoint-Specific Interface Registers

NAME	ADDRESS	DESCRIPTION
TXMAXP	base + 00h	Transmit Maximum Packet Size Register
CSR0	base + 02h	Control and Status Register (Endpoint 0)
TXCSR	base + 02h	Transmit Control and Status Register (Endpoints 1-3)
RXMAXP	base + 04h	Receive Maximum Packet Size Register
RXCSR	base + 06h	Receive Control and Status Register
COUNT0	base + 08h	Receive Count Register (Endpoint 0)
RXCOUNT	base + 08h	Receive Count Register (Endpoints 1-3)
TXTYPE	base + 0Ah	Transmit Transfer Type Register
NAKLIMIT0	base + 0Bh	NAK Limit Register (Endpoint 0)
TXINTERVAL	base + 0Bh	Transmit Polling Interval Register (Endpoints 1-3)
RXTYPE	base + 0Ch	Receive Transfer Type Register
RXINTERVAL	base + 0Dh	Receive Polling Interval Register (Endpoints 1-3)

Table 21-4. USB Endpoint Register Banks

NAME	BASE ADDRESS	DESCRIPTION
INDEXED	FF 0810h	Indexed Register Bank
EP0NIND	FF 0900h	Endpoint 0 Register Bank
EP1NIND	FF 0910h	Endpoint 1 Register Bank
EP2NIND	FF 0920h	Endpoint 2 Register Bank
EP3NIND	FF 0930h	Endpoint 3 Register Bank

21.3.1 Function Address Register (FADDR)

The FADDR register is an 8-bit, read/write register that holds the function address. When the USB controller is in Host mode, this register should be loaded with the value sent in a SET_ADDRESS command during device enumeration as the address for the peripheral device. When the USB controller is being used in Peripheral mode, this register should be loaded with the address received through a SET_ADDRESS command, which will then be used for decoding the function address in subsequent token packets. The new address will not take effect immediately, because the host will still be using the old address for the Status stage of the device request. The USB controller will continue to use the old address for decoding packets until the device request has completed. At reset, this register is cleared to 00h.

7	6	0
Reserved	FUNCADDR	

FUNCADDR The Function Address field holds the 7-bit address of the peripheral part of the transaction.

21.3.2 Power Register (POWER)

The POWER register is an 8-bit, read/write register that is used for controlling Suspend and Resume signaling, and some basic operational aspects of the USB controller. At reset, this register is initialized to 20h.

7	6	5	4	3	2	1	0
ISOUPDATE	SOFTCONN	Reserved (Must be 0)	Reserved (Must be 0)	RESET	RESUME	SUSPENDMODE	ENABLESUSPENDM

ENABLESUSPENDM The Enable Suspend Mode bit is used to shut down the USB transceiver in Suspend mode to save power.
0 – USB transceiver always enabled.
1 – USB transceiver disabled in Suspend mode.

SUSPENDMODE The Suspend Mode bit is set by software to enter Suspend mode (Host mode) or by hardware when Suspend mode is entered (Peripheral mode). It will be cleared when the INTRUSB register is read (as a result of receiving a Suspend interrupt). It will also be cleared if Suspend mode is exited by setting the Resume bit to initiate a remote wake-up.
0 – Suspend mode has not been entered.
1 – Suspend mode has been entered (Peripheral mode), or software has requested entry into Suspend mode (Host mode).

RESUME The Resume bit is set by software to generate Resume signaling on the USB to perform remote wake-up from Suspend mode. Once set, it should be left set for approximately 10 ms (at least 1 ms and no more than 15 ms), then cleared. In Host mode, this bit is set by hardware when Resume signaling from the peripheral is detected while the Host is in Suspend mode.
0 – No Resume signaling.
1 – Generate Resume signaling or Resume signaling detected.

RESET The Reset bit can be used to determine when Reset signaling is present on the USB. It is set when Reset signaling is detected and remains set until the bus reverts to an idle state. This bit is read/write in Host mode, and read only in Peripheral mode.
0 – No Reset signaling detected.
1 – Reset signaling detected.

SOFTCONN	The Soft Connect/Disconnect bit can be used when the USB controller is acting as a peripheral to switch the USB transceiver between normal mode and non-driving mode. As a result, it can be used to connect/disconnect the device from its host without having the change the physical connections, when the device is acting as a peripheral. After a hardware reset, the SOFTCONN bit is clear. The USB controller will therefore appear disconnected until the software has set the SOFTCONN bit. Software can then choose when to set the transceiver into its normal mode. Systems with a lengthy initialization procedure may use this to ensure that initialization is complete and the system is ready to perform enumeration before connecting to the USB. After the SOFTCONN bit has been set, clearing this bit will put the transceiver into non-driving mode. The USB controller will then appear to have been disconnected to other devices on the USB bus. This bit is only valid in Peripheral mode. 0 – USB transceiver is in non-driving mode. 1 – USB transceiver is enabled to drive D+ and D-.
ISOUPDATE	The Isochronous Update bit only affects IN Isochronous endpoints (Endpoint 1). It is normally used as a method of ensuring a “clean” start-up of an IN Isochronous pipe. When set, the USB controller will wait for an SOF token from the time TXPKTRDY is set before sending the packet. If an IN token is received before an SOF token, then a zero-length data packet will be sent. This bit is only valid in Peripheral mode. 0 – Normal mode. 1 – Isochronous update mode.

21.3.3 Interrupt Register for Endpoint 0 and Transmit Endpoints (INTRTX)

The INTRTX register is a 16-bit, read-only register that indicates which interrupts are currently asserted for Endpoint 0 and the Transmit Endpoints 1, 2, and 3. Bits for endpoints that have not been configured will always return 0. All asserted interrupts are cleared when this register is read. After reset, this register is cleared to 0000h.

15	4	3	2	1	0
Reserved		EP3TX	EP2TX	EP1TX	EP0

EP0	The Endpoint 0 bit indicates that an interrupt was asserted for Endpoint 0. 0 – No interrupt asserted. 1 – Interrupt asserted.
EPnTX	The Endpoint n Transmit bit indicates that an interrupt was asserted for Transmit Endpoint n (n = 1 to 3). 0 – No interrupt asserted. 1 – Interrupt asserted.

21.3.4 Interrupt Register for Receive Endpoints (INTRRX)

The INTRRX register is a 16-bit, read-only register that indicates which interrupts are currently asserted for Receive Endpoints 1, 2, and 3. Bits for endpoints that have not been configured will always return 0. All asserted interrupts are cleared when this register is read. After reset, this register is cleared to 0000h.

15	4	3	2	1	0
Reserved		EP3RX	EP2RX	EP1RX	Reserved

EPnRX The Endpoint n Receive bit indicates that an interrupt was asserted for Receive Endpoint n (n = 1 to 3).
0 – No interrupt asserted.
1 – Interrupt asserted.

21.3.5 Interrupt Enable Register for Endpoint 0 and Transmit Endpoints (INTRTXE)

The INTRTXE register is a 16-bit, read/write register that controls whether interrupts are enabled for Endpoint 0 and the Transmit Endpoints 1, 2, and 3. Bits for endpoints that have not been configured will always return 0. After reset, this register is initialized to 000Fh.

15 – 4	3	2	1	0
Reserved	EP3TX	EP2TX	EP1TX	EP0

EP0 The Endpoint 0 bit controls whether an interrupt is enabled for Endpoint 0.
0 – No interrupt enabled.
1 – Interrupt enabled.

EPnTX The Endpoint n Transmit bit controls whether an interrupt is enabled for Transmit Endpoint n (n = 1 to 3).
0 – No interrupt enabled.
1 – Interrupt enabled.

21.3.6 Interrupt Enable Register for Receive Endpoints (INTRRXE)

The INTRRXE register is a 16-bit, read/write register that controls whether interrupts are enabled for Receive End-points 1, 2, and 3. After reset, this register is initialized to 000Eh.

15 – 4	3	2	1	0
Reserved	EP3RX	EP2RX	EP1RX	Reserved

EPnRX The Endpoint n Receive bit controls whether an interrupt is enabled for Receive Endpoint n (n = 1 to 3).
0 – No interrupt enabled.
1 – Interrupt enabled.

21.3.7 Interrupt Register for USB Controller (INTRUSB)

The INTRUSB register is an 8-bit, read-only register that indicates which interrupts are currently asserted for the USB controller. All asserted interrupts are cleared when this register is read. After reset, this register is cleared to 00h.

7	6	5	4	3	2	1	0
VBUSERROR	SESSREQ	DISCON	CONN	SOF	RESETBABBLE	RESUME	SUSPEND

SUSPEND The Suspend bit indicates that an interrupt was asserted because Suspend signaling was detected on the bus. This bit is only valid in Peripheral mode.
0 – No interrupt asserted.
1 – Interrupt asserted.

RESUME The Resume bit indicates that an interrupt was asserted because Resume signaling was detected while the USB Controller was in Suspend mode.
0 – No interrupt asserted.
1 – Interrupt asserted.

RESETBABBLE The Reset/Babble bit indicates that an interrupt was asserted because babble was detected (unexpected bus activity) in Host mode or reset signaling was detected in Peripheral mode.
0 – No interrupt asserted.
1 – Interrupt asserted.

SOF	The Start of Frame bit indicates that an interrupt was asserted because a new frame started. 0 – No interrupt asserted. 1 – Interrupt asserted.
CONN	The Connection bit indicates that an interrupt was asserted because a device connection was detected. This bit is only valid in Host mode. 0 – No interrupt asserted. 1 – Interrupt asserted.
DISCON	The Disconnect bit indicates that an interrupt was asserted because a device disconnect was detected in Host mode or a session ended in Peripheral mode. 0 – No interrupt asserted. 1 – Interrupt asserted.
SESSREQ	The Session Request bit indicates that an interrupt was asserted because a Session Request signal was detected on the bus. This bit is only valid when the USB controller is an A device. 0 – No interrupt asserted. 1 – Interrupt asserted.
VBUSERROR	The VBUS Error bit indicates that an interrupt was asserted because VBUS dropped below the VBUS Valid threshold during a session. This bit is only valid when the USB controller is an A device. 0 – No interrupt asserted. 1 – Interrupt asserted.

21.3.8 Interrupt Enable Register for USB Controller (INTRUSBE)

The INTRUSBE register is an 8-bit, read/write register that controls which USB controller interrupts are enabled. After reset, this register is initialized to 06h.

7	6	5	4	3	2	1	0
VBUSERROR	SESSREQ	DISCON	CONN	SOF	RESETBABBLE	RESUME	SUSPEND

SUSPEND	The Suspend bit enables an interrupt when Suspend signaling is detected on the bus. This bit is only valid in Peripheral mode. 0 – No interrupt enabled. 1 – Interrupt enabled.
RESUME	The Resume bit enables an interrupt when Resume signaling is detected while the USB Controller is in Suspend mode. 0 – No interrupt enabled. 1 – Interrupt enabled.
RESETBABBLE	The Reset/Babble bit enables an interrupt when babble is detected (unexpected bus activity) in Host mode or reset signaling is detected in Peripheral mode. 0 – No interrupt enabled. 1 – Interrupt enabled.
SOF	The Start of Frame bit enables an interrupt when a new frame starts. 0 – No interrupt enabled. 1 – Interrupt enabled.
CONN	The Connection bit enables an interrupt when a device connection is detected. This bit is only valid in Host mode. 0 – No interrupt enabled. 1 – Interrupt enabled.

DISCON — The Disconnect bit enables an interrupt when a device disconnect is detected in Host mode or a session is ended in Peripheral mode.
0 – No interrupt enabled.
1 – Interrupt enabled.

SESSREQ — The Session Request bit enables an interrupt when a Session Request signal is detected on the bus. This bit is only valid when the USB controller is an A device.
0 – No interrupt enabled.
1 – Interrupt enabled.

VBUSERROR — The VBUS Error bit enables an interrupt when VBUS drops below the VBUS Valid threshold during a session. This bit is only valid when the USB controller is an A device.
0 – No interrupt enabled.
1 – Interrupt enabled.

21.3.9 Frame Register (FRAME)

The Frame register is a 16-bit, read-only register that holds the last received frame number. At reset, this register is cleared to 0000h.

15	11	10	0
Reserved		FRAMENUMBER	

FRAMENUMBER The Frame Number field holds the 11-bit frame number.

21.3.10 Index Register (INDEX)

The Index register is an 8-bit, read/write register that selects the endpoint-specific register bank which is accessible in the region FF 0810h to FF 081Fh. At reset, this register is cleared to 00h.

7	4	3	0
Reserved		INDEX	

INDEX The Index field holds the 4-bit endpoint index. Only 0000b, 0001b, 0010b, and 0011b are valid values.

21.3.11 Endpoint n FIFO Register (EPnFIFO)

The Endpoint n FIFO registers are 32-bit, read/write registers that access the transmit FIFOs when written and the receive FIFOs when read (n = 0 to 3).

31	0
FIFODATA	

21.3.12 Device Control Register (DEVCTL)

The DEVCTL register is an 8-bit, read/write register that is used to select whether the USB Controller is operating in Peripheral mode or in Host mode, and for controlling and monitoring the USB VBus line. After reset, this register is cleared to 00h.

7	6	5	4	3	2	1	0
BDEVICE	FSDEV	LSDEV	VBUS		HOSTMODE	HOSTREQ	SESSION

SESSION	The Session bit is set or cleared by software to start or end a session, when operating as an A device. As a B device, this bit is controlled by hardware to indicate when a session starts and ends. It is also set by software to initiate the Session Request Protocol, or cleared by software when in Suspend mode to perform a software disconnect. 0 – Session disabled. 1 – Session enabled.
HOSTREQ	The Host Request bit is set to initiate the Host Negotiation Protocol when Suspend mode is entered. It is cleared when Host Negotiation is completed. This bit is only valid as a B device. 0 – No host negotiation. 1 – Initiate host negotiation.
HOSTMODE	The Host Mode bit indicates whether the USB controller is in Host or Peripheral mode. This is a read-only bit. 0 – Peripheral mode. 1 – Host mode.
VBUS	The VBUS field indicates the current VBUS level. This is a read-only field. 00 – Below SessionEnd. 01 – Above SessionEnd, below AValid. 10 – Above AValid, below VBusValid. 11 – Above VBusValid.
LSDEV	The Low-Speed Device bit indicates when a low-speed device is detected. Low-speed mode is not supported. This is a read-only bit. 0 – Low-speed device not detected. 1 – Low-speed device detected.
FSDEV	The Full-Speed Device bit indicates when a full-speed device is detected. This bit is only valid in Host mode. This is a read-only bit. 0 – Full-speed device not detected. 1 – Full-speed device detected.
BDEVICE	The B Device bit indicates whether the USB controller is operating as an A device or a B device. This bit is only valid while a session is in progress. This is a read-only bit. 0 – A device. 1 – B device.

21.3.13 Endpoint 0 Control and Status Register (CSR0)

The CSR0 register is a 16-bit, read/write register that provides control and status bits for Endpoint 0. It has different formats in Peripheral and Host modes. After reset, this register is cleared to 0000h.

Peripheral Mode Format

15 – 9	8	7	6	5	4	3	2	1	0
Reserved	FLUSHFIFO	SERVICED SETUPEND	SERVICE DRXPKTRDY	SEND STALL	SETUP END	DATA END	SENT STALL	TXPKTRDY	RXPKTRDY

RXPKTRDY	The Receive Packet Ready bit is set when a data packet has been received. An interrupt is asserted (if enabled) when this bit is set. Software clears the RXPKTRDY bit by writing 1 to the SERVICEDRXPKTRDY bit. 0 – No data packet received. 1 – Data packet received.
TXPKTRDY	The Transmit Packet Ready bit is set by software after loading a data packet into the FIFO. The bit is cleared automatically when the data packet has been transmitted. An interrupt is asserted (if enabled) when this bit is cleared. 0 – No packet in transmit FIFO. 1 – Data packet in FIFO, ready to transmit.

SENTSTALL	The Sent STALL Handshake bit is set when a STALL handshake is transmitted. Software clears this bit to detect the next STALL handshake. 0 – No STALL handshake since the bit was last cleared. 1 – STALL handshake transmitted.
DATAEND	The Data End bit is set by software under three circumstances: when setting the TXPKTRDY bit for the last data packet, when clearing the RXPKTRDY bit after unloading the last data packet, or when setting the TXPKTRDY bit for a zero-length data packet. The bit is cleared automatically. 0 – Not the end of the data. 1 – Last or zero-length data packet.
SETUPEND	The Setup End bit is set when a control transaction ends before the DATAEND bit has been set. An interrupt will be asserted and the FIFO flushed at this time. The bit is cleared by writing a 1 to the SERVICEDSETUPEND bit. 0 – No unexpected end of control transaction. 1 – Unexpected end of control transaction.
SENDSTALL	The Send STALL Handshake bit is written with 1 to terminate the current transaction. The STALL handshake will be transmitted, and then this bit will be cleared automatically. 0 – Normal operation. 1 – Write 1 to send the STALL handshake.
SERVICEDRXPKTRDY	The Serviced Receive Packet Ready bit is written with 1 to clear the RXPKTRDY bit. The SERVICEDRXPKTRDY bit is cleared automatically. 0 – Normal state. 1 – Write 1 to clear the RXPKTRDY bit.
SERVICEDSETUPEND	The Serviced Setup End bit is written with 1 to clear the SETUPEND bit. The SERVICEDSETUPEND bit is cleared automatically. 0 – Normal state. 1 – Write 1 to clear the SETUPEND bit.
FLUSHFIFO	The Flush FIFO bit is written with 1 to flush the Endpoint 0 FIFO. The FIFO pointer is reset and the TXPKTRDY/RXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless TXPKTRDY/RXPKTRDY bit is set. The FLUSHFIFO bit is cleared automatically. 0 – Normal operation. 1 – Flush the Endpoint 0 FIFO and clear the TXPKTRDY/RXPKTRDY bit.

Host Mode Format

15 9	8	7	6	5	4	3	2	1	0
Reserved	FLUSHFIFO	NAKTIME OUT	STATUS PKT	REQPKT	ERROR	SETUP PKT	RXSTALL	TXPKTRDY	RXPKTRDY

RXPKTRDY	The Receive Packet Ready bit is set when a data packet has been received. An interrupt is asserted (if enabled) when this bit is set. Software clears this bit after unloading the packet from the FIFO to detect the next packet. 0 – No data packet received. 1 – Data packet received.
TXPKTRDY	The Transmit Packet Ready bit is set by software after loading a data packet into the FIFO. The bit is cleared automatically when the data packet has been transmitted. An interrupt is asserted (if enabled) when this bit is cleared. 0 – No packet in transmit FIFO. 1 – Data packet in FIFO, ready to transmit.

RXSTALL	The Receive STALL handshake bit is set when a STALL handshake is received. Software clears this bit to detect the next STALL handshake. 0 – No STALL handshake since the bit was last cleared. 1 – STALL handshake received.
SETUPPKT	The Setup Packet bit is set by software at the same time the TXPKTRDY bit is set, to select a SETUP token instead of an OUT token for the transaction. 0 – OUT token selected. 1 – SETUP token selected
ERROR	The Error bit is set when three attempts have been made to perform a transaction with no response from the peripheral. An interrupt is asserted when this bit is set. Software clears this bit to detect the next error event. 0 – No error event detected since the bit was last cleared. 1 – Error event detected.
REQPKT	The Request Packet bit is written with 1 to request an IN transaction. The REQPKT bit is cleared when the RXPKTRDY bit is set. 0 – No request asserted. 1 – Request for an IN transaction asserted.
STATUSPKT	The Status Packet bit is set by software at the same time as the TXPKTRDY or REQPKT bit is set, to perform a Status Stage transaction. Setting this bit ensures that the data toggle is set to 1 so that a DATA1 packet is used for the Status Stage transaction. 0 – Normal operation. 1 – Status Stage transaction selected.
NAKTIMEOUT	The NAK Timeout bit is set when Endpoint 0 is halted following the receipt of NAK responses for longer than the time set by the NAKLIMIT0 register. Software must clear this bit to allow the endpoint to continue. 0 – Normal operation. 1 – Endpoint 0 halted.
FLUSHFIFO	The FLUSHFIFO bit is written with 1 to flush the Endpoint 0 FIFO. The FIFO pointer is reset and the TXPKTRDY/RXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless TXPKTRDY/RXPKTRDY bit is set. 0 – Normal operation. 1 – Flush the Endpoint 0 FIFO and clear the TXPKTRDY/RXPKTRDY bit.

21.3.14 Endpoint 0 Count Register (COUNT0)

The COUNT0 register is an 8-bit, read-only register that indicates the current number of received data bytes in the Endpoint 0 FIFO. The value changes as the contents of the FIFO change, and it is only valid while the CSR0.RXPKTRDY bit is set. At reset, this register is cleared to 00h.

7	6 ... 0
Reserved	COUNT

COUNT	The Count field indicates how many valid bytes are in the Endpoint 0 receive FIFO.

21.3.15 Endpoint 0 NAK Limit Register (NAKLIMIT0)

The NAKLIMIT0 register is an 8-bit, read/write register that specifies the number of frames after which an Endpoint 0 timeout occurs in Host mode when receiving a stream of NAK responses. (Equivalent settings for other endpoints can be made through their TXINTERVAL and RXINTERVAL registers.)

The number of frames is 2^{m-1}, in which *m* is the value specified in the register. If the host receives NAK responses from the target for more frames than the number represented by the limit set in this register, the endpoint will be halted. At reset, this register is cleared to 00h.

7	5	4	0
Reserved		LIMIT	

LIMIT The Limit field specifies the Endpoint 0 timeout. A value of 0 or 1 disables the timeout function. Values from 2 to 16 (decimal) enable the timeout function. Values above 16 are reserved.

21.3.16 Transmit Maximum Packet Size Register (TXMAXP)

The TXMAXP register is a 16-bit, read/write register that specifies the maximum amount of data that can be transferred through the selected transmit endpoint in a single frame. This value must comply with the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transactions in full-speed operations. The value should match the wMaxPacketSize field of the Standard Endpoint Descriptor for the associated endpoint (see Universal Serial Bus Specification Revision 2.0, Chapter 9). A mismatch could cause unexpected results. The value written to this register must not exceed the transmit FIFO size. If the value written to this register is less than, or equal to, half the transmit FIFO size, two packets can be buffered. If this register is changed after packets have been sent from the endpoint, then the endpoint FIFO should be completely flushed (using the FLUSHFIFO bit in the TXCSR register) after writing the new value to the TXMAXP register. There is a TXMAXP register for each transmit endpoint (except Endpoint 0). At reset, this register is cleared to 0000h.

15	11	10	0
Reserved		MAXSIZE	

MAXSIZE The Maximum Packet Size field specifies (in bytes) the maximum payload transmitted in a single transaction. Valid values for this field are 8, 16, 32, and 64 (decimal).

21.3.17 Transmit Control and Status Register (TXCSR)

The TXCSR register is a 16-bit, read/write register that provides control and status bits for the associated endpoint (except Endpoint 0). It has different formats in Peripheral and Host modes. After reset, this register is initialized to 2000h.

Peripheral Mode Format

7	6	5	4	3	2	1	0
Reserved	CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	UNDERRUN	FIFONOTEMPTY	TXPKTRDY

15	14	13	12	11	10	9	8
AUTOSET	ISO	Reserved		FRCDATATOG	Reserved		

TXPKTRDY The Transmit Packet Ready bit is set by software after loading a data packet into the FIFO. The bit is cleared automatically when the data packet has been transmitted. An interrupt is asserted (if enabled) when this bit is cleared.
0 – No packet in transmit FIFO.
1 – Data packet in FIFO, ready to transmit.

FIFONOTEMPTY The FIFO Not Empty bit indicates that there is at least one packet in the transmit FIFO.
0 – Transmit FIFO is empty.
1 – Transmit FIFO is not empty.

UNDERRUN The Transmit Underrun bit is set when an IN token is received if the TXPKTRDY bit is clear. Software clears the UNDERRUN bit to detect the next underrun event.
0 – No underrun event detected since the bit was last cleared.
1 – Underrun event detected.

FLUSHFIFO — The FLUSHFIFO bit is written with 1 to flush the transmit FIFO. The FIFO pointer is reset and the TXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless TXPKTRDY bit is set. If double-buffering is enabled, the FLUSHFIFO bit may need to be written twice to completely clear the FIFO. The FLUSHFIFO bit is cleared automatically.
0 – Normal operation.
1 – Flush the transmit FIFO and clear the TXPKTRDY bit.

SENDSTALL — The Send STALL Handshake bit is set to send a STALL handshake in response to an IN token. Software must clear this bit to terminate the STALL condition. This bit is ignored when the endpoint is used for isochronous transfers.
0 – Normal operation.
1 – STALL condition.

SENTSTALL — The Sent STALL Handshake bit is set when a STALL handshake is transmitted. Also, the transmit FIFO is flushed, and the TXPKTRDY bit is cleared. Software clears this bit to detect the next STALL handshake.
0 – No STALL handshake since the bit was last cleared.
1 – STALL handshake transmitted.

CLRDATATOG — The Clear Data Toggle bit is written with 1 to reset the endpoint data toggle to 0.
0 – Normal operation.
1 – Write 1 to reset endpoint data toggle to 0.

FRCDATATOG — The Force Data Toggle bit is written with 1 to force the endpoint data toggle to switch and the data packet to be cleared from the FIFO, without regard to whether an ACK was received. This can be used by Interrupt transmit endpoints that are used to communicate rate feedback for Isochronous endpoints.
0 – Normal operation.
1 – Write 1 to force endpoint data to toggle.

ISO — The Isochronous Transfer bit specifies whether the Bulk/Interrupt or Isochronous transfer type is used.
0 – Bulk/Interrupt transfer type.
1 – Isochronous transfer type.

AUTOSET — The Automatic Set bit is set by software to enable a mode in which the TXPKTRDY bit is automatically set when data of the maximum packet size (value in TXMAXP register) is loaded in the transmit FIFO. If a packet of less than the maximum packet size is loaded, then the TXPKTRDY bit must be set explicitly by software. The TXPKTRDY bit is also automatically set when the first of two packets in the transmit FIFO has been sent and the second packet is the maximum packet size.
0 – Normal mode.
1 – Automatically set TXPKTRDY when maximum packet size packet is loaded into the transmit FIFO.

Host Mode Format

7	6	5	4	3	2	1	0
NAKTIMEOUT	CLRDATATOG	RXSTALL	Reserved	FLUSHFIFO	ERROR	FIFONOTEMPTY	TXPKTRDY

15	14	13	12	11	10	9	8
AUTOSET	Reserved			FRCDATATOG			Reserved

TXPKTRDY — The Transmit Packet Ready bit is set by software after loading a data packet into the FIFO. The bit is cleared automatically when the data packet has been transmitted. An interrupt is asserted (if enabled) when this bit is cleared.
0 – No packet in transmit FIFO.
1 – Data packet in FIFO, ready to transmit.

FIFONOTEMPTY	The FIFO Not Empty bit indicates that there is at least one packet in the transmit FIFO. 0 – Transmit FIFO is empty. 1 – Transmit FIFO is not empty.
ERROR	The Error bit is set when three attempts have been made to send a packet and no handshake packet has been received. An interrupt is asserted when this bit is set. Software clears the ERROR bit to detect the next error event. This bit is only valid for Bulk and Interrupt endpoints. 0 – No error event detected since the bit was last cleared. 1 – Error event detected.
FLUSHFIFO	The FLUSHFIFO bit is written with 1 to flush the transmit FIFO. The FIFO pointer is reset and the TXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless TXPKTRDY bit is set. If double-buffering is enabled, the FLUSHFIFO bit may need to be written twice to completely clear the FIFO. The FLUSHFIFO bit is cleared automatically. 0 – Normal operation. 1 – Flush the transmit FIFO and reset the TXPKTRDY bit.
RXSTALL	The Receive STALL Handshake bit is set when a STALL handshake is received. The FIFO pointer is reset and the TXPKTRDY bit is cleared. Software clears the RXSTALL bit to detect the next STALL handshake. 0 – No STALL event detected since the bit was last cleared. 1 – STALL event detected.
CLRDATATOG	The Clear Data Toggle bit is written with 1 to reset the endpoint data toggle to 0. 0 – Normal operation. 1 – Write 1 to reset endpoint data toggle to 0.
NAKTIMEOUT	The NAK Timeout bit is set when the transmit endpoint is halted following the receipt of NAK responses for longer than the time set as the NAK Limit in the TXINTERVAL register. Software clears the NAKTIMEOUT bit to allow the endpoint to continue. This bit is only valid for Bulk endpoints. 0 – No NAK timeout event detected since the bit was last cleared. 1 – NAK timeout event detected.
FRCDATATOG	The Force Data Toggle bit is written with 1 to force the endpoint data toggle to switch and the data packet to be cleared from the FIFO, without regard to whether an ACK was received. This can be used by Interrupt transmit endpoints that are used to communicate rate feedback for Isochronous endpoints. 0 – Normal operation. 1 – Write 1 to force endpoint data to toggle.
AUTOSET	The Automatic Set bit is set by software to enable a mode in which the TXPKTRDY bit is automatically set when data of the maximum packet size (value in TXMAXP register) is loaded in the transmit FIFO. If a packet of less than the maximum packet size is loaded, then the TXPKTRDY bit must be set explicitly by software. The TXPKTRDY bit is also automatically set when the first of two packets in the transmit FIFO have been sent and the second packet is the maximum packet size. 0 – Normal mode. 1 – Automatically set TXPKTRDY when maximum packet size packet is loaded into the transmit FIFO.

21.3.18 Receive Maximum Packet Size Register (RXMAXP)

The RXMAXP register is a 16-bit, read/write register that specifies the maximum amount of data that can be transferred through the selected receive endpoint in a single frame. The value is subject to the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt, and Isochronous transfers. The value should match the wMaxPacketSize field of the Standard Endpoint Descriptor for the associated endpoint (see *Universal Serial Bus Specification Revision 2.0*, Chapter 9). A mismatch could cause unexpected results. The value written to this register must not exceed the receive FIFO size. If the value written to this register is less than, or equal to, half the receive FIFO size, two packets can be buffered. There is a RXMAP register for each receive endpoint (except Endpoint 0). At reset, this register is cleared to 0000h.

15	11	10	0
Reserved		MAXSIZE	

MAXSIZE The Maximum Packet Size field specifies (in bytes) the maximum payload received in a single transaction.

21.3.19 Receive Control and Status Register (RXCSR)

The RXCSR register is a 16-bit, read/write register that provides control and status bits for the associated endpoint (except Endpoint 0). It has different formats in Peripheral and Host modes. After reset, this register is cleared to 0000h.

Peripheral Mode Format

7	6	5	4	3	2	1	0
CLRDATATOG	SENTSTALL	SENDSTALL	FLUSHFIFO	DATAERROR	OVERRUN	FIFOFULL	RXPKTRDY

15	14	13	12	11	10	8
AUTOCLEAR	ISO	Reserved				

RXPKTRDY The Receive Packet Ready bit is set when a data packet is received. Software should clear this bit when the packet is unloaded from the receive FIFO to detect the next packet. An interrupt is asserted (if enabled) when this bit is set.
0 – No packet in receive FIFO.
1 – Data packet in FIFO, ready to unload.

FIFOFULL The FIFO Full bit indicates that no more packets can be loaded into the receive FIFO.
0 – Receive FIFO is not full.
1 – Receive FIFO is full.

OVERRUN The Overrun bit is set when an OUT packet cannot be loaded into the receive FIFO. Software clears the OVERRUN bit to detect the next overrun event. This bit is only valid for Isochronous endpoints.
0 – No overrun event detected since the bit was last cleared.
1 – Overrun event detected.

DATAERROR The Data Error bit is set when the RXPKTRDY bit is set and the data packet has a CRC or bit-stuffing error. Software clears this bit to detect the next data error. This bit is only valid for Isochronous endpoints.
0 – No data error since the bit was last cleared.
1 – Data error detected.

FLUSHFIFO — The FLUSHFIFO bit is written with 1 to flush the receive FIFO. The FIFO pointer is reset and the RXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless RXPKTRDY bit is set. If double-buffering is enabled, the FLUSHFIFO bit may need to be written twice to completely clear the FIFO. The FLUSHFIFO bit is cleared automatically.
0 – Normal operation.
1 – Flush the transmit FIFO and reset the RXPKTRDY bit.

SENDSTALL — The Send STALL Handshake bit is set by software to issue a STALL handshake. Software must clear this bit to terminate the STALL condition. This bit is ignored for Isochronous end-points.
0 – Normal operation.
1 – STALL condition.

SENTSTALL — The Sent STALL Handshake bit is set when a STALL handshake is transmitted. Software clears this bit to detect the next STALL handshake.
0 – No STALL handshake since the bit was last cleared.
1 – STALL handshake transmitted.

CLRDATATOG — The Clear Data Toggle bit is written with 1 to reset the endpoint data toggle to 0.
0 – Normal operation.
1 – Write 1 to reset endpoint data toggle to 0.

ISO — The Isochronous Transfer bit specifies whether the Bulk/Interrupt or Isochronous transfer type is used.
0 – Bulk/Interrupt transfer type.
1 – Isochronous transfer type.

AUTOCLEAR — The Automatic Clear bit is set by software to enable a mode in which the RXPKTRDY bit is automatically cleared when data of the maximum packet size (value in RXMAXP register) has been unloaded from the receive FIFO. If a packet of less than the maximum packet size is unloaded, then the RXPKTRDY bit must be set explicitly by software.
0 – Normal mode.
1 – Automatically clear RXPKTRDY when maximum packet size packet is unloaded from the receive FIFO.

Host Mode Format

7	6	5	4	3	2	1	0
CLRDATATOG	RXSTALL	REQPKT	FLUSHFIFO	DATAERROR NAKTIMEOUT	ERROR	FIFOFULL	RXPKTRDY

15	14	13	12	11	10	8
AUTOCLEAR	AUTOREQ	Reserved				

RXPKTRDY — The Receive Packet Ready bit is set when a data packet has been received in the FIFO. Software should clear this bit when the FIFO is unloaded to detect the next packet. An interrupt is asserted (if enabled) when this bit is set.
0 – No packet in receive FIFO.
1 – Data packet in FIFO, ready to unload.

FIFOFULL — The FIFO Full bit is set when no more packets can be loaded into the receive FIFO.
0 – Receive FIFO is empty.
1 – Receive FIFO is full.

ERROR	The Error bit is set when three attempts have been made to receive a packet and no packet has been received. Software clears the ERROR bit to detect the next error event. An interrupt is asserted when the bit is set. This bit is only valid for Bulk and Interrupt endpoints. 0 – No error event detected since the bit was last cleared. 1 – Error event detected.
DATAERROR	The DATAERROR bit (Isochronous endpoint only) is set when the RXPKTRDY bit is set and the data packet has a CRC or bit-stuffing error. It is clear when the RXPKTRDY bit is clear. 0 – No data error since the bit was last cleared. 1 – Data error detected.
NAKTIMEOUT	The NAKTIMEOUT bit (Bulk endpoint only) is set when the receive endpoint is halted following receipt of NAK responses for longer than the time set as the NAK limit in the RXINTERVAL register. Software should clear this bit to allow the endpoint to continue. 0 – No timeout. 1 – NAK timeout occurred.
FLUSHFIFO	The FLUSHFIFO bit is written with 1 to flush the receive FIFO. The FIFO pointer is reset and the RXPKTRDY bit is cleared. The FLUSHFIFO bit has no effect unless RXPKTRDY bit is set. If double-buffering is enabled, the FLUSHFIFO bit may need to be written twice to completely clear the FIFO. 0 – Normal operation. 1 – Flush the receive FIFO and clear the RXPKTRDY bit.
REQPKT	The Request Packet bit is written with 1 to request an IN transaction. The REQPKT bit is cleared when the RXPKTRDY bit is set. 0 – No request asserted. 1 – Request for an IN transaction asserted.
RXSTALL	The Receive STALL Handshake bit is set when a STALL handshake is received. Software clears the RXSTALL bit to detect the next STALL handshake. An interrupt is asserted (if enabled) when this bit is set. 0 – No STALL handshake detected since the bit was last cleared. 1 – STALL handshake detected.
CLRDATATOG	The Clear Data Toggle bit is written with 1 to reset the endpoint data toggle to 0. 0 – Normal operation. 1 – Write 1 to reset endpoint data toggle to 0.
AUTOREQ	The Automatic Request bit specifies whether the REQPKT bit is automatically set when the RXPKTRDY bit is cleared. 0 – Automatic request is disabled. 1 – Automatic request is enabled.
AUTOCLEAR	The Automatic Clear bit is set by software to enable a mode in which the RXPKTRDY bit is automatically cleared when data of the maximum packet size (value in RXMAXP register) is unloaded from the receive FIFO. If a packet of less than the maximum packet size is unloaded, then the RXPKTRDY bit must be set explicitly by software. 0 – Normal mode. 1 – Automatically clear RXPKTRDY when maximum packet size packet is unloaded from the receive FIFO.

21.3.20 Receive Count Register (RXCOUNT)

The RXCOUNT register is a 16-bit, read-only register that indicates the current number of received data bytes in the receive FIFO. The value changes as the contents of the FIFO change, and it is only valid while the RXPKTRDY bit is set. At reset, this register is cleared to 0000h.

15 – 13	12 – 0
Reserved	COUNT

COUNT The Count field indicates how many valid bytes are in the receive FIFO.

21.3.21 Transmit Transfer Type Register (TXTYPE)

The TXTYPE register is an 8-bit, read/write register that must be written with the endpoint number to be targeted by the endpoint and the transaction protocol to use for the currently selected transmit endpoint. There is a TXTYPE register for each endpoint (except Endpoint 0). This register is only used in Host mode. At reset, this register is cleared to 00h.

7 – 6	5 – 4	3 – 0
Reserved	PROTOCOL	ENDPOINT

ENDPOINT The Endpoint field specifies the target endpoint. Software must set this value to the endpoint number contained in the transmit endpoint descriptor returned during device enumeration.

PROTOCOL The Protocol field specifies the protocol for the transmit endpoint.
00 – Reserved.
01 – Isochronous.
10 – Bulk.
11 – Interrupt.

21.3.22 Transmit Interval Register (TXINTERVAL)

The TXINTERVAL register is an 8-bit, read/write register that specifies the polling interval for the currently selected transmit endpoint, for Interrupt and Isochronous endpoints. For Bulk endpoints, this register specifies the number of frames after which the endpoint should timeout on receiving a stream of NAK responses. There is a TXINTERVAL register for each endpoint (except Endpoint 0). This register is only used in Host mode. At reset, this register is cleared to 00h.

7 – 0
INTERVAL/TIMEOUT

INTERVAL/TIMEOUT The INTERVAL/TIMEOUT field specifies a number of frames. The value is interpreted differently depending on the transfer type.

TRANSFER TYPE	VALID VALUES (n)	NUMBER OF FRAMES
Interrupt	1-255	n
Iso.	1-16	2^{n-1}
Bulk	2-16 (0 or 1 disables NAK function)	2^{n-1}

21.3.23 Receive Transfer Type Register (RXTYPE)

The RXTYPE register is an 8-bit, read/write register that must be written with the endpoint number to be targeted by the endpoint and the transaction protocol to use for the currently selected receive endpoint. There is a RXTYPE register for each endpoint (except Endpoint 0). This register is only valid in Host mode. At reset, this register is cleared to 00h.

15 6	5 4	3 0
Reserved	PROTOCOL	ENDPOINT

ENDPOINT — The Endpoint field specifies the target endpoint. Software must set this value to the endpoint number contained in the receive endpoint descriptor returned during device enumeration.

PROTOCOL — The Protocol field specifies the protocol for the receive endpoint.
00 – Reserved.
01 – Isochronous.
10 – Bulk.
11 – Interrupt.

21.3.24 Receive Interval Register (RXINTERVAL)

The RXINTERVAL register is an 8-bit, read/write register that specifies the polling interval for the currently selected receive endpoint, for Interrupt and Isochronous endpoints. For Bulk endpoints, this register specifies the number of frames after which the endpoint should timeout on receiving a stream of NAK responses. There is a RXINTERVAL register for each endpoint (except Endpoint 0). This register is only valid in Host mode. At reset, this register is cleared to 00h.

7 0
INTERVAL/TIMEOUT

INTERVAL/TIMEOUT — INTERVAL/TIMEOUT field specifies a number of frames. The value is interpreted differently depending on the transfer type.

TRANSFER TYPE	VALID VALUES (n)	NUMBER OF FRAMES
Interrupt	1-255	n
Iso.	1-16	2^{n-1}
Bulk	2-16 (0 or 1 disables NAK function)	2^{n-1}

21.3.25 USB Transceiver Control Register (VCTRL)

The VCTRL register is a 16-bit, read/write register used to provide the address for accessing the USB transceiver control registers. These registers are only accessible through this register-based interface. The VCTRL register also provides data for writes to those registers. At reset, this register is cleared to 0000h.

15	14 8	7 0
DWEN	ADDRESS	DATA

DATA — The Data field specifies the data loaded into the addressed register when writes are performed.

ADDRESS — The Address field specifies the register address. The only valid address is:
01h – PTCI_SUSPCTRL.

DWEN — The Data Write Enable bit controls whether the addressed register is written with the data.
0 – Write disabled.
1 – Write enabled.

21.3.26 USB Transceiver Status Register (VSTATUS)

The VSTATUS register is an 8-bit, read-only register that returns the data in the addressed USB transceiver control register. At reset, this register is cleared to 00h.

7	0
VSTATUS	

VSTATUS The VSTATUS field returns the data in the register addressed by the VCTRL register.

21.3.27 USB Suspend Control Register (PTCI_SUSPCTRL)

The PTCI_SUSPCTRL register is an 8-bit, write-only register which provides an interface for forcing the USB transceiver and OTG signals into Suspend mode. At reset, this register is cleared to 00h.

7 4	3	2	1	0
Reserved	SUSPOTG_CTRL	SUSP_CTRL	FORCE_SUSP	FORCE_SUSPOTG

FORCE_SUSPOTG The Force Suspend OTG bit forces the USB OTG signals into Suspend mode, when selected by the SUSPOTG_CTRL bit.
0 – Normal operation.
1 – Suspend mode.

FORCE_SUSP The Force Suspend bit forces the USB transceiver into Suspend mode, when selected by the SUSP_CTRL bit.
0 – Normal operation.
1 – Suspend mode

SUSP_CTRL The Suspend Control bit selects whether the USB logic or the FORCE_SUSP bit controls Suspend mode for the USB transceiver.
0 – USB logic.
1 – FORCE_SUSP bit.

SUSPOTG_CTRL The Suspend OTG bit selects whether the USB logic or the FORCE_SUSPOTG bit controls Suspend mode for the USB OTG signals.
0 – USB logic.
1 – FORCE_SUSPOTG bit.

22 Dual CAN Interfaces

Each CAN interface contains a Full CAN class, CAN (Controller Area Network) serial bus interface for low/high speed applications. It supports reception and transmission of extended frames with a 29-bit identifier, standard frames with an 11-bit identifier, applications that require high speed (up to 1 MBit/s), and a low-speed CAN interface with CAN master capability. Data transfer between the CAN bus and the CPU is handled by 15 message buffers per interface, which can be individually configured as receive or transmit buffers. Every message buffer includes a status/control register which provides information about its current status and capabilities to configure the buffer. All message buffers are able to assert an interrupt on the reception of a valid frame or the successful transmission of a frame. In addition, an interrupt can be generated on bus errors.

An incoming message is only accepted if the message identifier passes one of two acceptance filtering masks. The filtering mask can be configured to receive a single message ID for each buffer or a group of IDs for each receive buffer. One of the buffers uses a separate message filtering procedure. This provides the capability to establish a BASIC-CAN path. Remote transmission requests can be processed automatically by automatic reconfiguration to a receiver after transmission or by automated transmit scheduling upon reception. A priority decoder allows any buffer to have one of 16 transmit priorities including the highest or lowest absolute priority, for a total of 240 different transmit priorities.

A decided bit time counter (16-bit wide) is provided to support real time applications. The contents of this counter are captured into the message buffer RAM on reception or transmission. The counter can be synchronized through the CAN network. This synchronization feature allows a reset of the counter after the reception or transmission of a message in buffer 0.

The CAN modules are fast APB bus peripherals which allow single-cycle read/write access. The CPU controls the CAN modules by programming the registers in the CAN register blocks. This includes initialization of the CAN baud rate, logic level of the CAN pins, and enable/disable of the CAN modules. A set of diagnostic features, such as loopback, listen only, and error identification, support development with the CAN modules and provide a sophisticated error management tool.

The CAN modules implement the following features:

- CAN specification 2.0B
  - Standard data and remote frames
  - Extended data and remote frames
  - 0 to 8 bytes data length
  - Programmable bit rate up to 1 Mbit/s
- 15 message buffers, each configurable as receive or transmit buffers
  - Message buffers are 16-bit wide RAM mapped to the lower words of a 32-bit wide memory space
  - One buffer may be used as a BASIC-CAN path
- Remote Frame support
  - Automatic transmission after reception of a Remote Transmission Request (RTR)
  - Auto receive after transmission of a RTR
- Acceptance filtering
  - Two filtering capabilities: global acceptance mask and individual buffer identifiers
  - One of the buffers uses an independent acceptance filtering procedure
- Programmable transmit priority
- Interrupt capability
  - One interrupt vector for all message buffers in each module (receive/transmit/error)
  - Each interrupt source can be enabled/disabled
- 16-bit counter with time stamp capability on successful reception or transmission of a message
- Power Save capabilities with programmable Wake-Up over the CAN bus (alternate source for the Multi-Input Wake-Up module)
- Push-pull capability of the input/output pins
- Diagnostic functions
  - Error identification
  - Loopback and listen-only features for test and initialization purposes

22.1 Functional Description

As shown in Figure 22-1, the CAN modules consists of three blocks: the CAN core, interface management, and a RAM containing the message buffers.

There are two dedicated device pins for each CAN interface, CANnTX for the transmit output and CANnRX for the receive input (n = 0 or 1).

The CAN core implements the basic CAN protocol features such as bit-stuffing, CRC calculation/checking, and error management. It controls the transceiver logic and creates error signals according to the bus rules. In addition, it converts the data stream from the CPU (parallel data) to the serial CAN bus data.

The interface management block is divided into the register block and the interface management processor. The register block provides the CAN interface with control information from the CPU and provides the CPU with status information from the CAN module. Additionally, it generates the interrupt to the CPU.

The interface management processor is a state machine executing the CPU's transmission and reception commands and controlling the data transfer between several message buffers and the RX/TX shift registers.

Fifteen (15) message buffers are memory mapped into RAM to transmit and receive data through the CAN bus. Eight 16-bit registers belong to each buffer. One of the registers contains control and status information about the message buffer configuration and the current state of the buffer. The other registers are used for the message identifier, a maximum of up to eight data bytes, and the time stamp information. During the receive process, the incoming message will be stored in a hidden receive buffer until the message is valid. Then, the buffer contents will be copied into the first message buffer which accepts the ID of the received message.

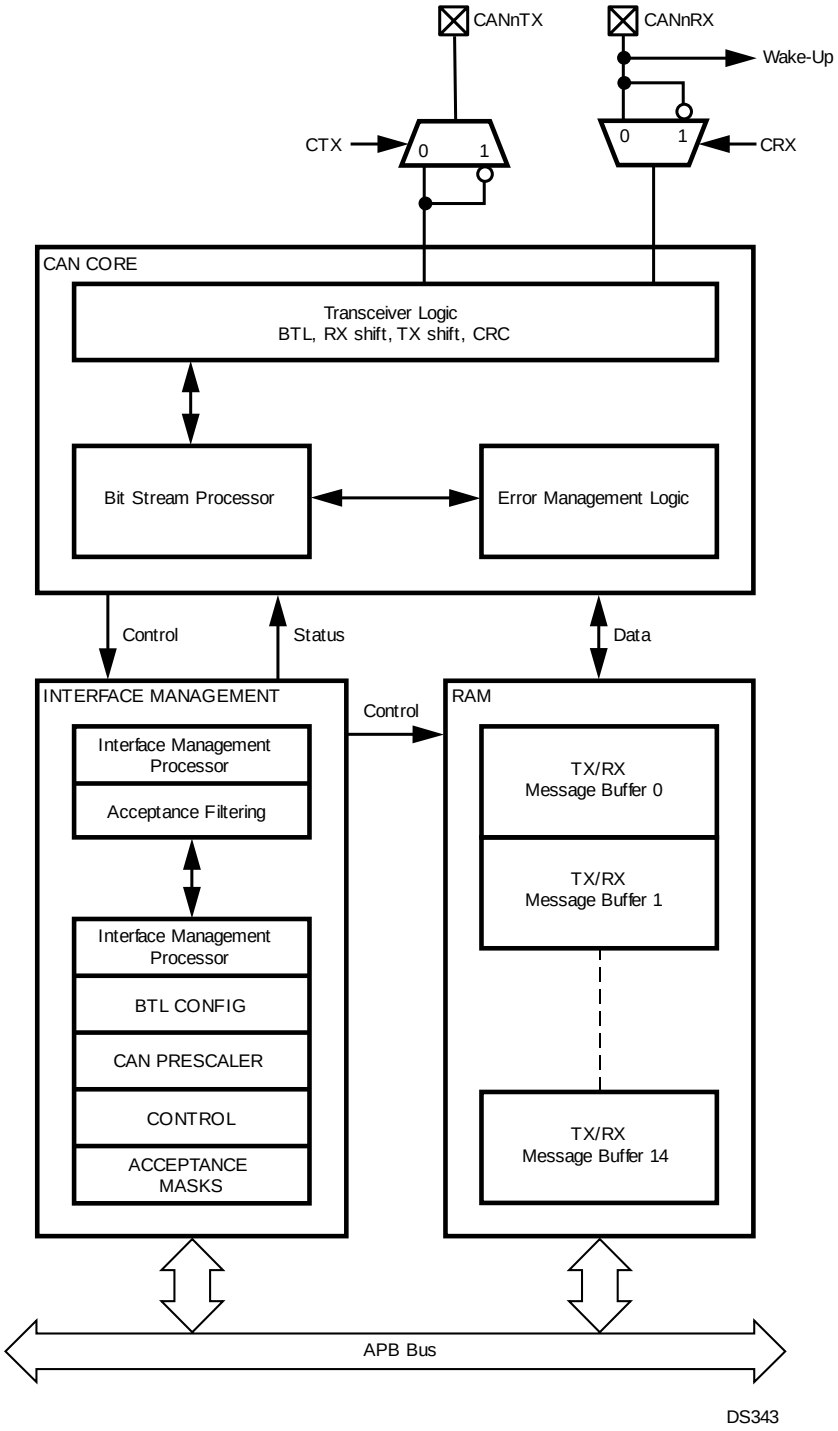


Figure 22-1. CAN Block Diagram

22.2 Basic CAN Concepts

This section provides a generic overview of the basic concepts of the Controller Area Network (CAN).

The CAN protocol is a message-based protocol that allows a total of 2032 (2^{11} –16) different messages in the standard format and 512 million (2^{29} – 16) different messages in the extended frame format.

Every CAN Frame is broadcast on the common bus. Each module receives every frame and filters out the frames which are not required for the module's task. For example, if a dashboard sends a request to switch on headlights, the CAN module responsible for brake lights must not process this message.

A CAN master module has the ability to set a specific bit called the “remote data request bit” (RTR) in a frame. Such a message is also called a “Remote Frame”. It causes another module, either another master or a slave which accepts this remote frame, to transmit a data frame after the remote frame has been completed.

Additional modules can be added to an existing network without a configuration change. These modules can either perform completely new functions requiring new data, or process existing data to perform a new functionality.

As the CAN network is message oriented, a message can be used as a variable which is automatically updated by the controlling processor. If any module cannot process information, it can send an overload frame.

The CAN protocol allows several transmitting modules to start a transmission at the same time as soon as they detect the bus is idle. During the start of transmission, every node monitors the bus line to detect whether its message is over-written by a message with a higher priority. As soon as a transmitting module detects another module with a higher priority accessing the bus, it stops transmitting its own frame and switches to receive mode, as shown in Figure 22-2.

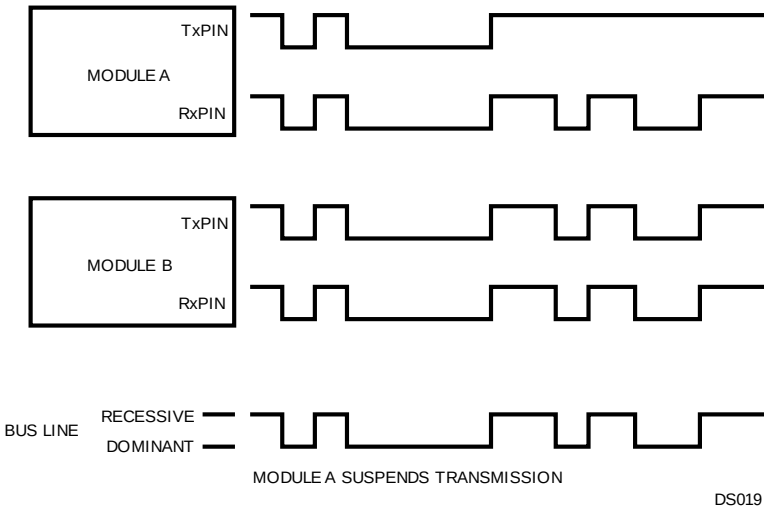


Figure 22-2. CAN Message Arbitration

If a data or remote frame loses arbitration on the bus due to a higher-prioritized data or remote frame, or if it is destroyed by an error frame, the transmitting module will automatically retransmit it until the transmission is successful or software has canceled the transmit request.

If a transmitted message loses arbitration, the CAN module will restart transmission at the next possible time with the message which has the highest internal transmit priority.

22.2.1 CAN Frame Types

Communication via the CAN bus is basically established by means of four different frame types:

- Data Frame
- Remote Frame
- Error Frame
- Overload Frame

Data and remote frames can be used in both standard and extended frame format. If no message is being transmitted, i.e., the bus is idle, the bus is kept at the “recessive” level.

Remote and data frames are non-return to zero (NRZ) coded with bit-stuffing in every bit field, which holds computable information for the interface, i.e., start of frame, arbitration field, control field, data field (if present), and CRC field.

Error and overload frames are also NRZ coded, but without bit-stuffing.

After five consecutive bits of the same value (including inserted stuff bits), a stuff bit of the inverted value is inserted into the bit stream by the transmitter and deleted by the receiver. The following shows the stuffed and destuffed bit stream for consecutive ones and zeros.

Original or unstuffed bit stream	10000011111 . . .	01111100000 . . .
Stuffed bit stream (stuff bits in bold)	1000001111101 . . .	0111110000010 . . .

22.2.2 CAN Frame Fields

Data and remote frames consist of the following bit fields:

- Start of Frame (SOF)
- Arbitration Field
- Control Field
- Data Field
- CRC Field
- ACK Field
- EOF Field

22.2.2.1 Start of Frame (SOF)

The Start of Frame (SOF) indicates the beginning of data and remote frames. It consists of a single “dominant” bit. A node is only allowed to start transmission when the bus is idle. All nodes have to synchronize to the leading edge (first edge after the bus was idle) caused by the SOF of the node which starts transmission first.

22.2.2.2 Arbitration Field

The Arbitration field consists of the identifier field and the RTR (Remote Transmission Request) bit. For extended frames there is also a SRR (Substitute Remote Request) and a IDE (ID Extension) bit inserted between ID18 and ID17 of the identifier field. The value of the RTR bit is “dominant” in a data frame and “recessive” in a remote frame.

22.2.2.3 Control Field

The Control field consists of six bits. For standard frames it starts with the ID Extension bit (IDE) and a reserved bit (RB0). For extended frames, the control field starts with two reserved bits (RB1, RB0). These bits are followed by the 4 bit Data Length Code (DLC).

The CAN receiver accepts all possible combinations of the reserved bits (RB1, RB0). The transmitter must be configured to send only zeros.

22.2.2.4 Data Length Code (DLC)

The DLC field indicates the number of bytes in the data field. It consists of four bits. The data field can be of length zero. The admissible number of data bytes for a data frame ranges from 0 to 8.

22.2.2.5 Data Field

The Data field consists of the data to be transferred within a data frame. It can contain 0 to 8 bytes. A remote frame has no data field.

22.2.2.6 Cyclic Redundancy Check (CRC)

The CRC field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived by the transmitter from the modulo 2 division of the preceding bit fields, starting with the SOF up to the end of the data field, excluding stuff-bits, by the generator polynomial:

$$x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1$$

The remainder of this division is the CRC sequence transmitted over the bus. On the receiver side, the module divides all bit fields up to the CRC delimiter excluding stuff bits, and checks if the result is zero. This will then be interpreted as a valid CRC. After the CRC sequence a single “recessive” bit is transmitted as the CRC delimiter.

22.2.2.7 ACK Field

The ACK field is two bits long and contains the ACK slot and the ACK delimiter. The ACK slot is filled with a “recessive” bit by the transmitter. This bit is overwritten with a “dominant” bit by every receiver that has received a correct CRC sequence. The second bit of the ACK field is a “recessive” bit called the acknowledge delimiter.

The End of Frame field closes a data and a remote frame. It consists of seven “recessive” bits.

22.2.3 CAN Frame Formats

22.2.3.1 Data Frame

The structure of a standard data frame is shown in Figure 22-3. The structure of an extended data frame is shown in Figure 22-4.

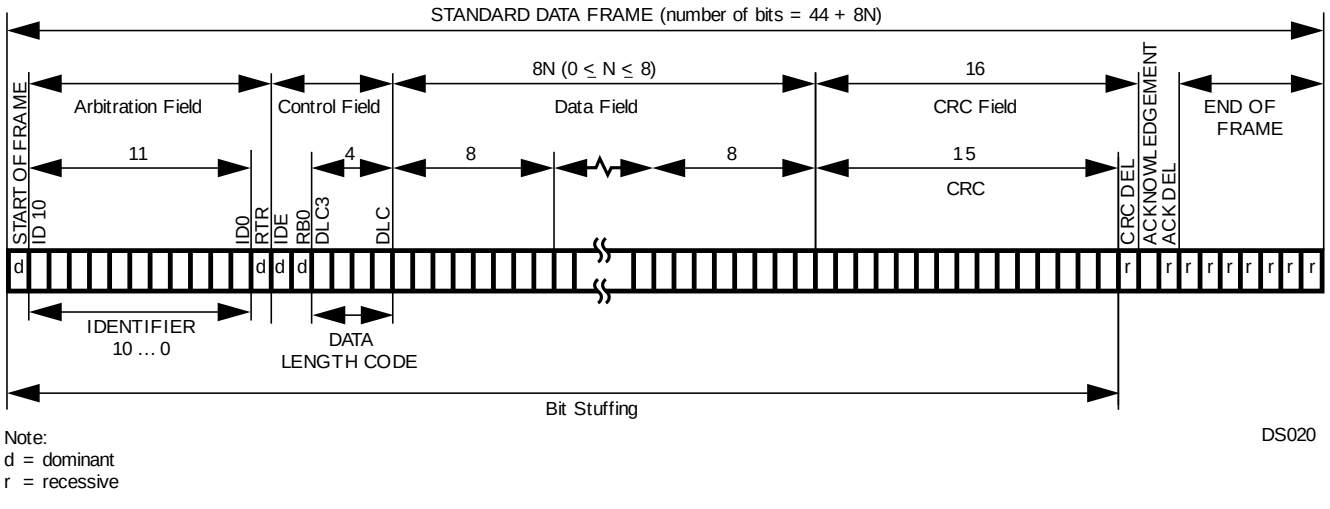


Figure 22-3. Standard Data Frame

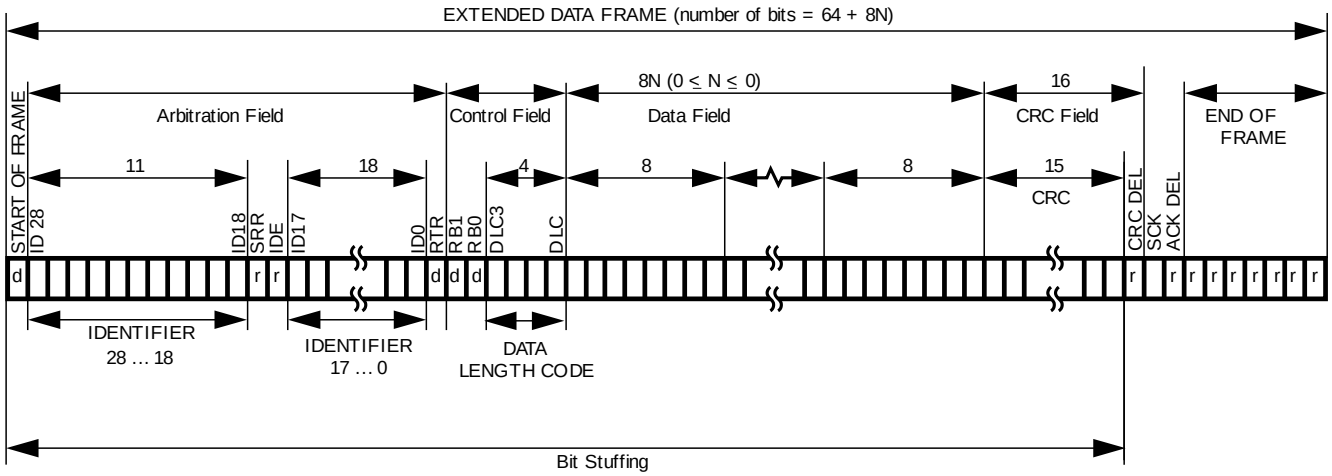


Figure 22-4. Extended Data Frame

A CAN data frame consists of the following fields:

- Start of Frame (SOF)
- Arbitration Field + Extended Arbitration
- Control Field
- Data Field
- Cyclic Redundancy Check Field (CRC)
- Acknowledgment Field (ACK)
- End of Frame (EOF)

22.2.3.2 Remote Frame

Figure 22-5 shows the structure of a standard remote frame. Figure 22-6 shows the structure of an extended remote frame.

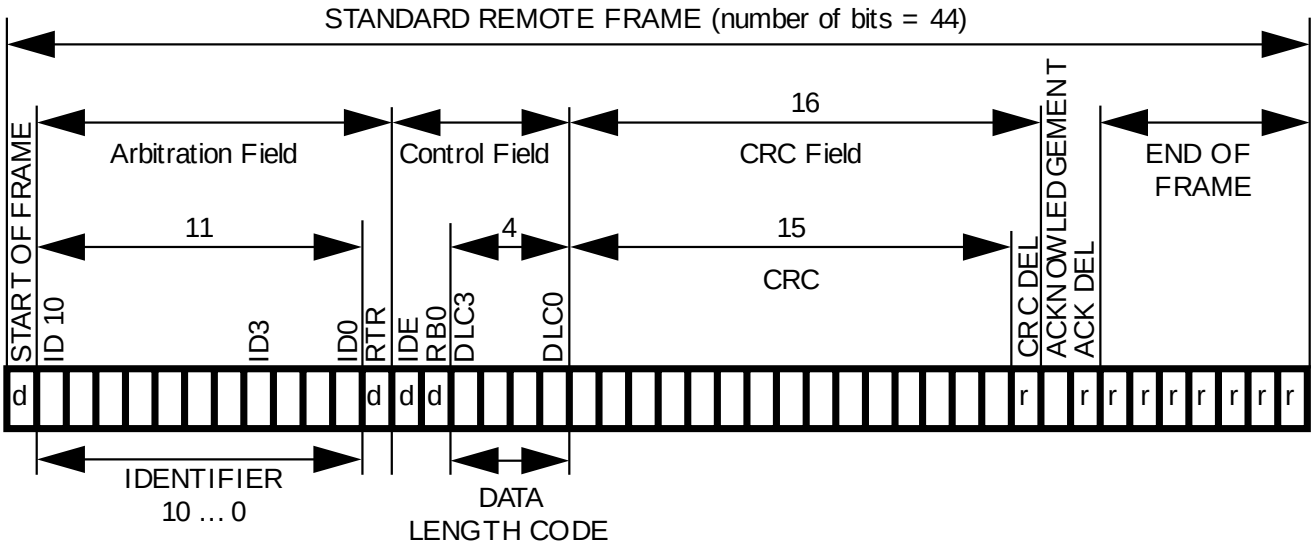


Figure 22-5. Standard Remote Frame

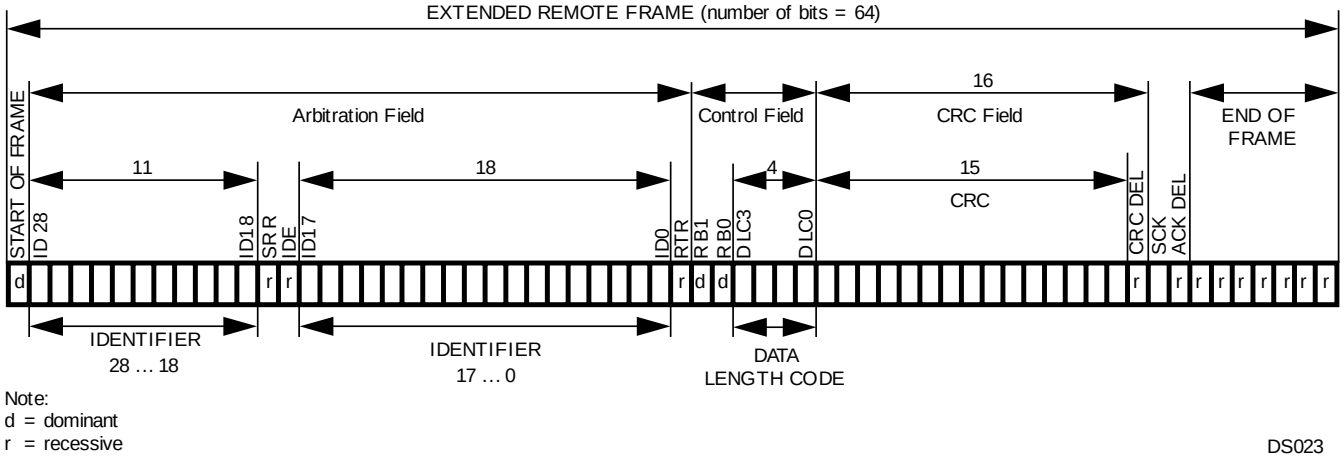


Figure 22-6. Extended Remote Frame

A remote frame is comprised of the following fields, which is the same as a data frame (see Section 22.2.2) except for the data field, which is not present.

- Start of Frame (SOF
- Arbitration Field + Extended Arbitration
- Control Field
- Cyclic Redundancy Check Field (CRC)
- Acknowledgment field (ACK)
- End of Frame (EOF)

Note that the DLC must have the same value as the corresponding data frame to prevent contention on the bus. The RTR bit is “recessive”.

22.2.3.3 Error Frame

As shown in Figure 22-7, the Error Frame consists of the error flag and the error delimiter bit fields. The error flag field is built up from the various error flags of the different nodes. Therefore, its length may vary from a minimum of six bits up to a maximum of twelve bits depending on when a module has detected the error. Whenever a bit error, stuff error, form error, or acknowledgment error is detected by a node, the node starts transmission of an error flag at the next bit. If a CRC error is detected, transmission of the error flag starts at the bit following the acknowledge delimiter, unless an error flag for a previous error condition has already been started.

If a device is in the error active state, it can send a “dominant” error flag, while a error passive device is only allowed to transmit “recessive” error flags. This is done to prevent the CAN bus from getting stuck due to a local defect. For the various CAN device states, refer to Section 22.2.4.

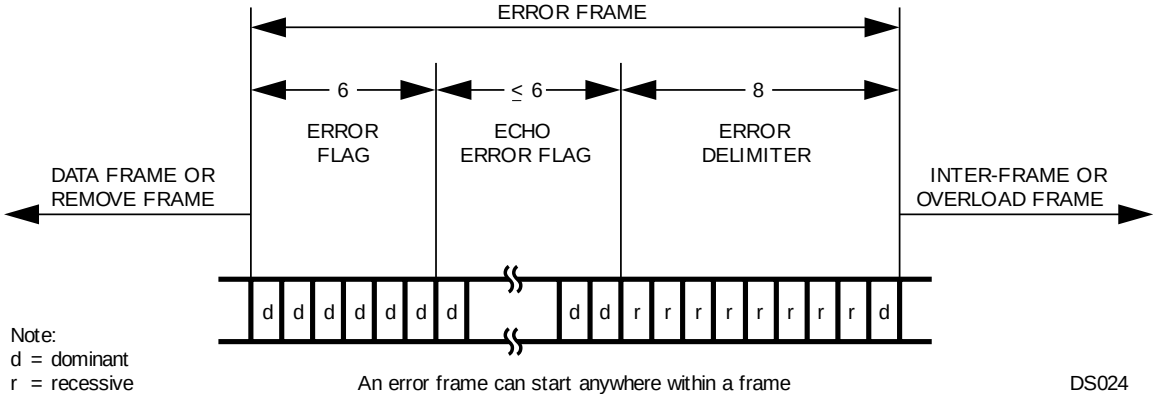


Figure 22-7. Error Frame

22.2.3.4 Overload Frame

As shown in Figure 22-8, an overload frame consists of the overload flag and the overload delimiter bit fields. The bit fields have the same length as the error frame field: six bits for the overload flag and eight bits for the delimiter. The overload frame can only be sent after the end of frame (EOF) field and in this way destroys the fixed form of the intermission field. As a result, all other nodes also detect an overload condition and start the transmission of an overload flag. After an overload flag has been transmitted, the overload frame is closed by the overload delimiter.

NOTE

A CAN module never initiates an overload frame due to its inability to process an incoming message. However, it is able to recognize and respond to overload frames initiated by other devices.

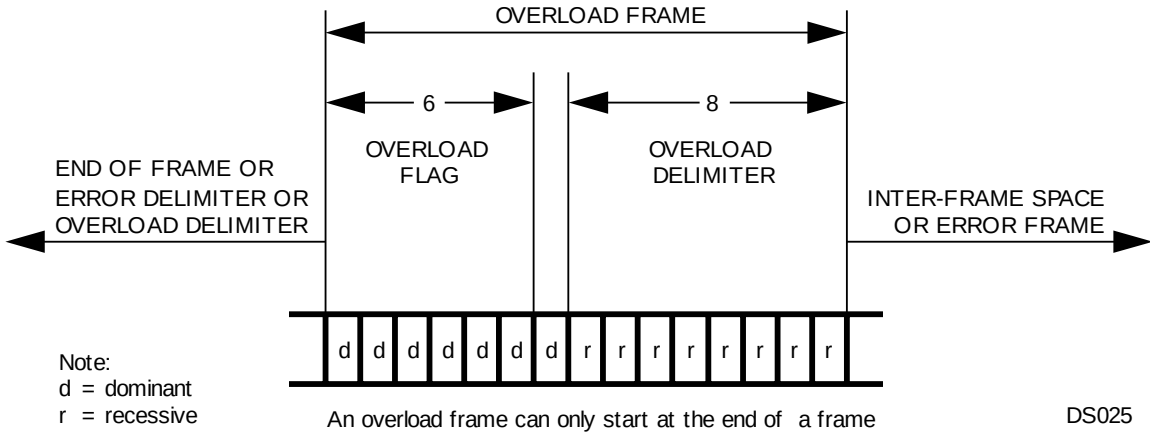


Figure 22-8. Overload Frame

22.2.3.5 Interframe Space

Data and remote frames are separated from every preceding frame (data, remote, error and overload frames) by the interframe space (see Figure 22-9). Error and overload frames are not preceded by an interframe space; they can be transmitted as soon as the condition occurs. The interframe space consists of a minimum of three bit fields depending on the error state of the node.

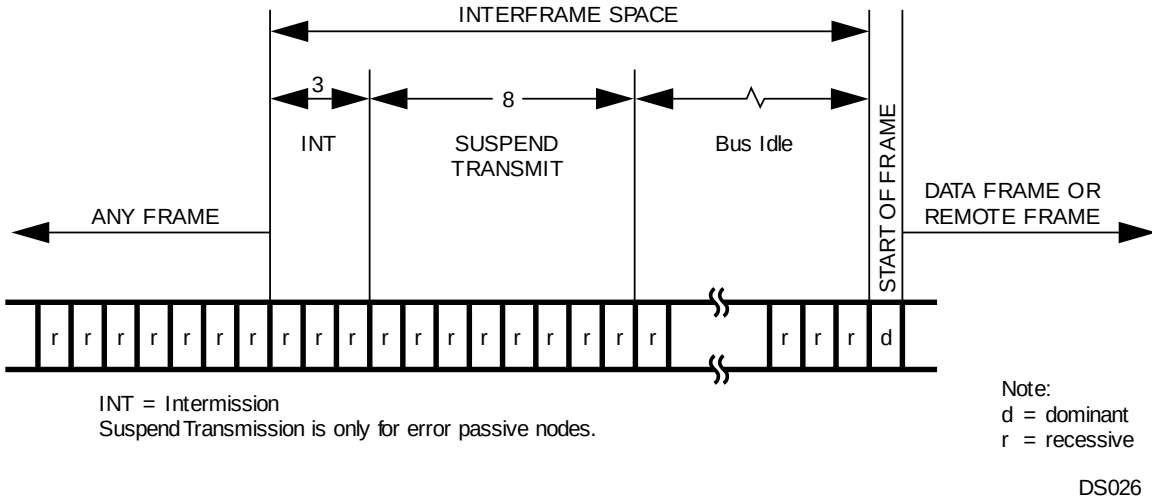


Figure 22-9. Interframe Space

22.2.4 Error Types

22.2.4.1 Bit Error

A CAN device which is currently transmitting also monitors the bus. If the monitored bit value is different from the transmitted bit value, a bit error is detected. However, the reception of a “dominant” bit instead of a “recessive” bit during the transmission of a passive error flag, during the stuffed bit stream of the arbitration field, or during the acknowledge slot is not interpreted as a bit error.

22.2.4.2 Stuff Error

A stuff error is detected if 6 consecutive bits occur without a state change in a message field encoded with bit stuffing.

22.2.4.3 Form Error

A form error is detected, if a fixed frame bit (e.g., CRC delimiter, ACK delimiter) does not have the specified value. For a receiver, a “dominant” bit during the last bit of End of Frame does not constitute a frame error.

22.2.4.4 Bit CRC Error

A CRC error is detected if the remainder from the CRC calculation of a received CRC polynomial is non-zero.

22.2.4.5 Acknowledgment Error

An acknowledgment error is detected whenever a transmitting node does not get an acknowledgment from any other node (i.e., when the transmitter does not receive a “dominant” bit during the ACK frame).

22.2.4.6 Error States

The device can be in one of five states with respect to error handling (see Figure 22-10).

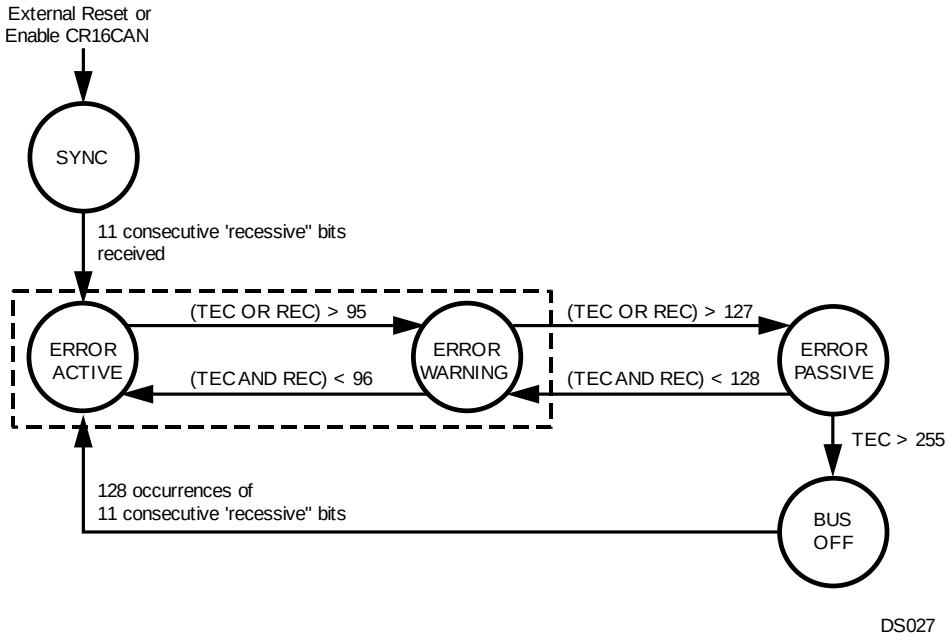


Figure 22-10. Bus States

22.2.4.7 Synchronize

Once a CAN module is enabled, it waits for 11 consecutive recessive bits to synchronize with the bus. After that, a CAN module becomes error active and can participate in the bus communication. This state must also be entered after waking-up the device using the Multi-Input Wake-Up feature. See Section 22.11.

22.2.4.8 Error Active

An error active unit can participate in bus communication and may send an active (“dominant”) error flag.

22.2.4.9 Error Warning

The Error Warning state is a sub-state of Error Active to indicate a heavily disturbed bus. A CAN module behaves as in Error Active mode. The device is reset into the Error Active mode if the value of both counters is less than 96.

22.2.4.10 Error Passive

An error passive unit can participate in bus communication. However, if the unit detects an error it is not allowed to send an active error flag. The unit sends only a passive (“recessive”) error flag. A device is error passive when the transmit error counter or the receive error counter is greater than 127. A device becoming error passive will send an active error flag. An error passive device becomes error active again when both transmit and receive error counter are less than 128.

22.2.4.11 Bus Off

A unit that is bus off has the output drivers disabled, i.e., it does not participate in any bus activity. A device is bus off when the transmit error counter is greater than 255. A bus off device will become error active again after monitoring 128 × 11 “recessive” bits (including bus idle) on the bus. When the device goes from “bus off“ to “error active“, both error counters will have a value of 0.

22.2.5 Error Counters

There are multiple mechanisms in the CAN protocol to detect errors and inhibit erroneous modules from disabling all bus activities. Each CAN module includes two error counters to perform error management. The receive error counter (REC) and the transmit error counter (TEC) are 8-bits wide, located in the 16-bit wide CANEC register. The counters are modified by the CAN module according to the rules listed in Table 22-1. This table provides an overview of the CAN error conditions and the behavior of the CAN module; for a detailed description of the error management and fault confinement rules, refer to the CAN Specification 2.0B.

If the MSB (bit 7) of the REC is set, the node is error passive and the REC will not increment any further.

The Error counters can be read by application software as described under Section 22.10.15.

Table 22-1. Error Counter Handling

CONDITION	ACTION
Receive Error Counter Conditions	
A receiver detects a bit error during sending an active error flag.	Increment by 8
A receiver detects a “dominant“ bit as the first bit after sending an error flag	Increment by 8
After detecting the 14th consecutive “dominant“ bit following an active error flag or overload flag, or after detecting the 8th consecutive “dominant“ bit following a passive error flag. After each sequence of additional 8 consecutive “dominant” bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK)	Increment by 1
A valid reception or transmission	Decrement by 1 unless counter is already 0
Transmit Error Counter Conditions	
A transmitter detects a bit error while sending an active error flag	Increment by 8
After detecting the 14th consecutive “dominant“ bit following an active error flag or overload flag or after detecting the 8th consecutive “dominant“ bit following a passive error flag. After each sequence of additional 8 consecutive ‘dominant’ bits.	Increment by 8
Any other error condition (stuff, frame, CRC, ACK)	Increment by 8
A valid reception or transmission	Decrement by 1 unless counter is already 0

Special error handling for the TEC counter is performed in the following situations:

- A stuff error occurs during arbitration, when a transmitted “recessive” stuff bit is received as a “dominant” bit. This does not lead to an increment of the TEC.
- An ACK-error occurs in an error passive device and no “dominant” bits are detected while sending the passive error flag. This does not lead to an increment of the TEC.
- If only one device is on the bus and this device transmits a message, it will get no acknowledgment. This will be detected as an error and the message will be repeated. When the device goes “error passive” and detects an acknowledge error, the TEC counter is not incremented. Therefore the device will not go from ”error passive” to the “bus off” state due to such a condition.

22.2.6 Bit Time Logic

In the Bit Time Logic (BTL), the CAN bus speed and the Synchronization Jump Width can be configured by software. The CAN module divides a nominal bit time into three time segments: synchronization segment, time segment 1 (TSEG1), and time segment 2 (TSEG2). Figure 22-11 shows the various elements of a CAN bit time.

22.2.6.1 CAN Bit Time

The number of time quanta in a CAN bit (CAN Bit Time) ranges between 4 and 25. The sample point is positioned between TSEG1 and TSEG2 and the transmission point is positioned at the end of TSEG2.

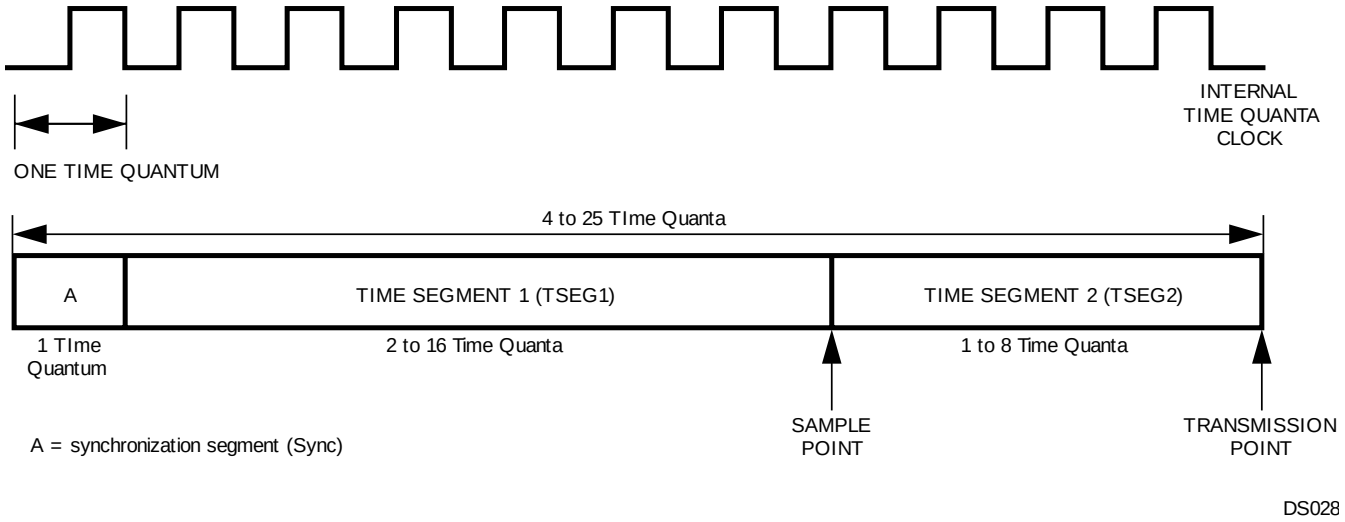


Figure 22-11. Bit Timing

TSEG1 includes the propagation segment and the phase segment 1 as specified in the CAN specification 2.0B. The length of the time segment 1 in time quanta (tq) is defined by the TSEG1[3:0] bits.

TSEG2 represents the phase segment 2 as specified in the CAN specification 2.0B. The length of time segment 2 in time quanta (tq) is defined by the TSEG2[3:0] bits.

The Synchronization Jump Width (SJW) defines the maximum number of time quanta (tq) by which a received CAN bit can be shortened or lengthened in order to achieve resynchronization on “recessive” to “dominant” data transitions on the bus. In the CAN implementation, the SJW must be configured less or equal to TSEG1 or TSEG2, whichever is smaller.

22.2.6.2 Synchronization

The Synchronization Jump Width (SJW) defines the maximum number of time quanta (tq) by which a received CAN bit can be shortened or lengthened in order to achieve re-synchronization on “recessive” to “dominant” data trans tions on the bus. In the CAN implementation, the SJW must be configured less or equal to TSEG1 or TSEG2, whichever is smaller.

However, two CAN nodes never operate at exactly the same clock rate, and the bus signal may deviate from the ideal waveform due to the physical conditions of the network (bus length and load). To compensate for the various delays within a network, the sample point can be positioned by programming the length of TSEG1 and TSEG2 (see Figure 22-11).

In addition, two types of synchronization are supported. The BTL logic compares the incoming edge of a CAN bit with the internal bit timing. The internal bit timing can be adapted by either hard or soft synchronization (re-synchronization).

Hard synchronization is performed at the beginning of a new frame with the falling edge on the bus while the bus is idle. This is interpreted as the SOF. It restarts the internal logic.

Soft synchronization is performed during the reception of a bit stream to lengthen or shorten the internal bit time. Depending on the phase error (e), TSEG1 may be increased or TSEG2 may be decreased by a specific value, the re-synchronization jump width (SJW).

The phase error is given by the deviation of the edge to the SYNC segment, measured in CAN clocks. The value of the phase error is defined as:

e = 0, if the edge occurs within the SYNC segment
e > 0, if the edge occurs within TSEG1
e < 0, if the edge occurs within TSEG2 of previous bit

Re-synchronization is performed according to the following rules:

- If the magnitude of e is less then or equal to the programmed value of SJW, resynchronization will have the same effect as hard synchronization.
- If e > SJW, TSEG1 will be lengthened by the value of the SJW (see Figure 22-12).
- If e < –SJW, TSEG2 will be shortened by the value SJW (see Figure 22-13).

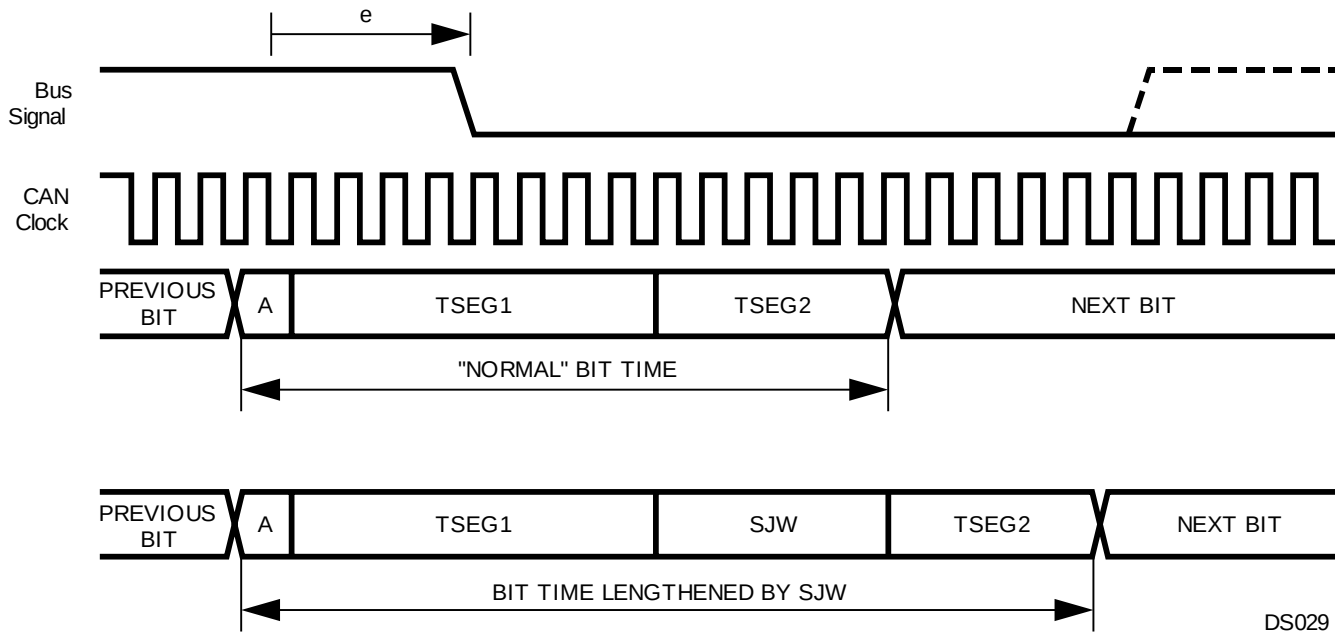


Figure 22-12. Resynchronization (e > SJW)

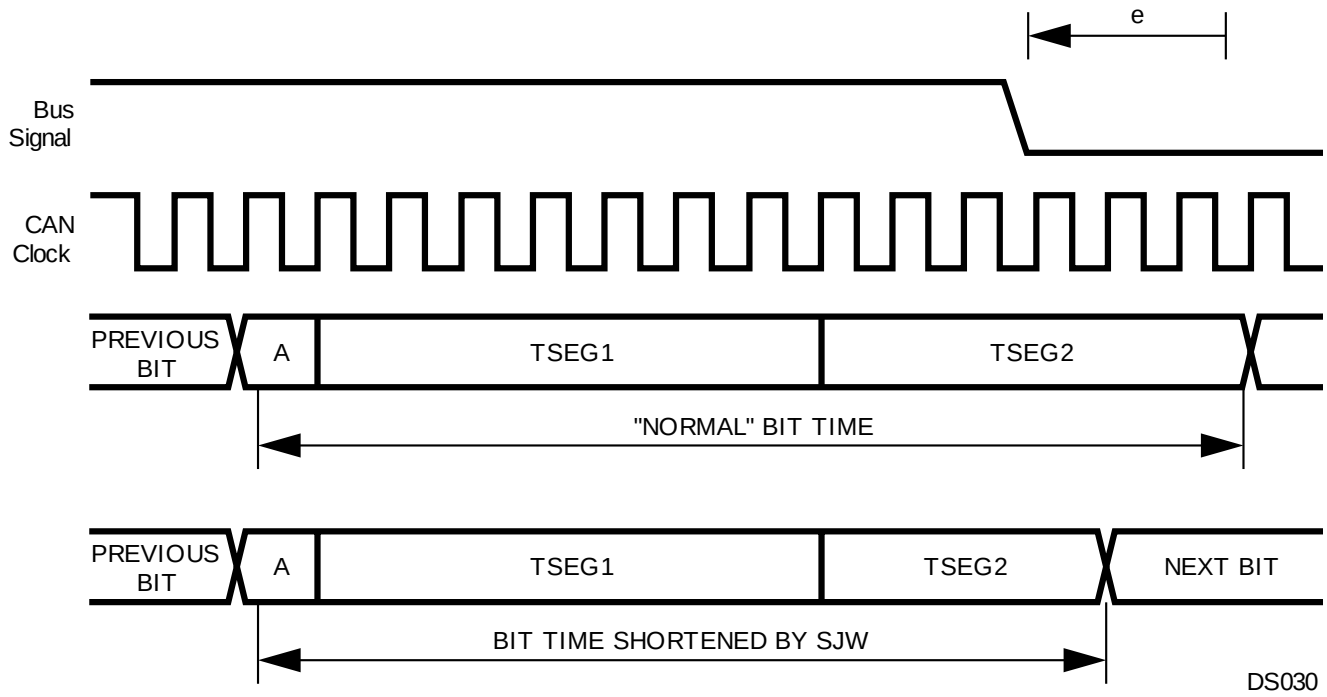


Figure 22-13. Resynchronization (e < –SJW)

22.2.7 Clock Generator

The CAN prescaler (PSC) is shown is Figure 22-14. It divides the CKI input clock by the value defined in the CTIM register. The resulting clock is called time quanta clock and defines the length of one time quantum (tq).

Refer to Section 22.10.7 for a detailed description of the CTIM register.

NOTE

PSC is the value of the clock prescaler. TSEG1 and TSEG2 are the length of time segment 1 and 2 in time quanta.

The resulting bus clock can be calculated by Equation 6:

$$\text{busclock} = \frac{\text{CKI}}{(\text{PSC}) \times (1 + \text{TSEG1} + \text{TSEG2})} \quad (6)$$

The values of PSC, TSEG1, and TSEG2 are specified by the contents of the registers PSC, TSEG1, and TSEG2 as follows:

PSC = PSC[5:0] + 2
TSEG1 = TSEG1[3:0] + 1
TSEG2 = TSEG2[2:0] + 1

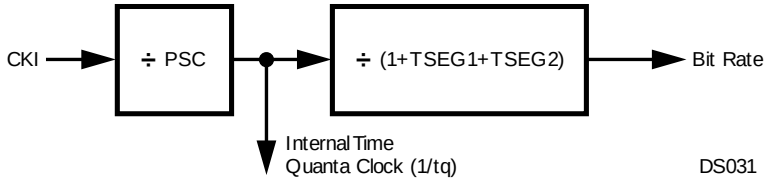


Figure 22-14. CAN Prescaler

22.3 Message Transfer

Each CAN module has access to 15 independent message buffers, which are memory mapped in RAM. Each message buffer consists of 8 different 16-bit RAM locations and can be individually configured as a receive message buffer or as a transmit message buffer.

A dedicated acceptance filtering procedure enables software to configure each buffer to receive only a single message ID or a group of messages. One buffer uses an independent filtering procedure, which provides the possibility to establish a BASIC-CAN path.

For reception of data frame or remote frames, a CAN module follows a “receive on first match” rule which means that a given message is only received by one buffer: the first one which matches the received message ID.

The transmission of a frame can be initiated by software writing to the transmit status and priority register. An alternate way to schedule a transmission is the automatic answer to remote frames. In the latter case, the CAN module will schedule every buffer for transmission to respond to remote frames with a given identifier if the acceptance mask matches. This implies that a single remote frame is able to poll multiple matching buffers configured to respond to the triggering remote transmission request.

22.4 Acceptance Filtering

Two 32-bit masks are used to filter unwanted messages from the CAN bus: GMASK and BMASK. Figure 22-15 shows the mask and the buffers controlled by the masks.

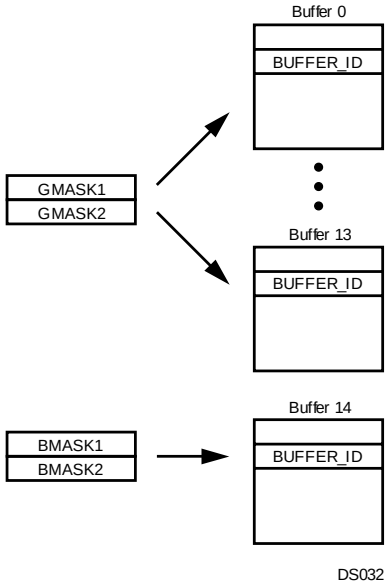


Figure 22-15. Acceptance Filtering

Acceptance filtering of the incoming messages for the buffers 0...13 is performed by means of a global filtering mask (GMASK) and by the buffer ID of each buffer. Acceptance filtering of incoming messages for buffer 14 is performed by a separate filtering mask (BMASK) and by the buffer ID of that buffer.

Once a received object is waiting in the hidden buffer to be copied into a buffer, the CAN module scans all buffers configured as receive buffers for a matching filtering mask. The buffers 0 to 13 are checked in ascending order beginning with buffer 0. The contents of the hidden buffer are copied into the first buffer with a matching filtering mask.

Bits holding a 1 in the global filtering mask (GMASK) can be represented as a “don’t care” of the associated bit of each buffer identifier, regardless of whether the buffer identifier bit is 1 or 0.

This provides the capability to accept only a single ID for each buffer or to accept a group of IDs. The following two examples illustrate the difference.

22.4.1 Example 1: Acceptance of a Single Identifier

If the global mask is loaded with 00h, the acceptance filtering of an incoming message is only determined by the individual buffer ID. This means that only one message ID is accepted for each buffer.

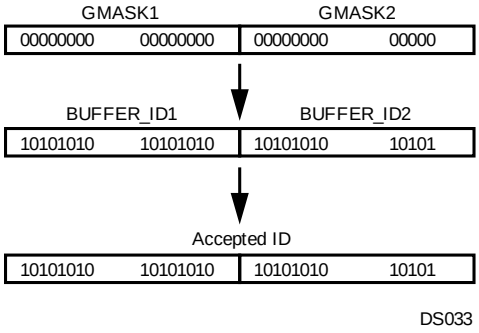


Figure 22-16. Acceptance of a Single Identifier

22.4.2 Example 2: Reception of an Identifier Group

Set bits in the global mask register change the corresponding bit status within the buffer ID to “don’t care” (X). Messages which match the non-“don’t care” bits (the bits corresponding to clear bits in the global mask register) are accepted.

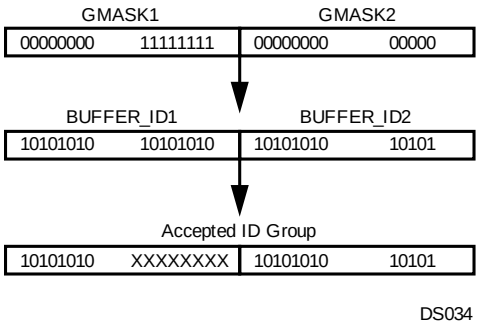


Figure 22-17. Acceptance of a Group of Identifiers

A separate filtering path is used for buffer 14. For this buffer, acceptance filtering is established by the buffer ID in conjunction with the basic filtering mask. This basic mask uses the same method as the global mask (set bits correspond to “don’t care” bits in the buffer ID).

Therefore, the basic mask allows a large number of infrequent messages to be received by this buffer.

NOTE

If the BMASK register is equal to the GMASK register, the buffer 14 can be used the same way as the buffers 0 to 13.

The buffers 0 to 13 are scanned prior to buffer 14. Subsequently, the buffer 14 will not be checked for a matching ID when one of the buffers 0 to 13 has already received an object.

By setting the BUFFLOCK bit in the configuration register, the receiving buffer is automatically locked after reception of one valid frame. The buffer will be unlocked again after the CPU has read the data and has written RX_READY in the buffer status field. With this lock function, software has the capability to save several messages with the same identifier or same identifier group into more than one buffer. For example, a buffer with the second highest priority will receive a message if the buffer with the highest priority has already received a message and is now locked (provided that both buffers use the same acceptance filtering mask).

As shown in Figure 22-18, several messages with the same ID are received while BUFFLOCK is enabled. The filtering mask of the buffers 0, 1, 13, and 14 is set to accept this message. The first incoming frame will be received by buffer 0. Because buffer 0 is now locked, the next frame will be received by buffer 1, and so on. If all matching receive buffers are full and locked, a further incoming message will not be received by any buffer.

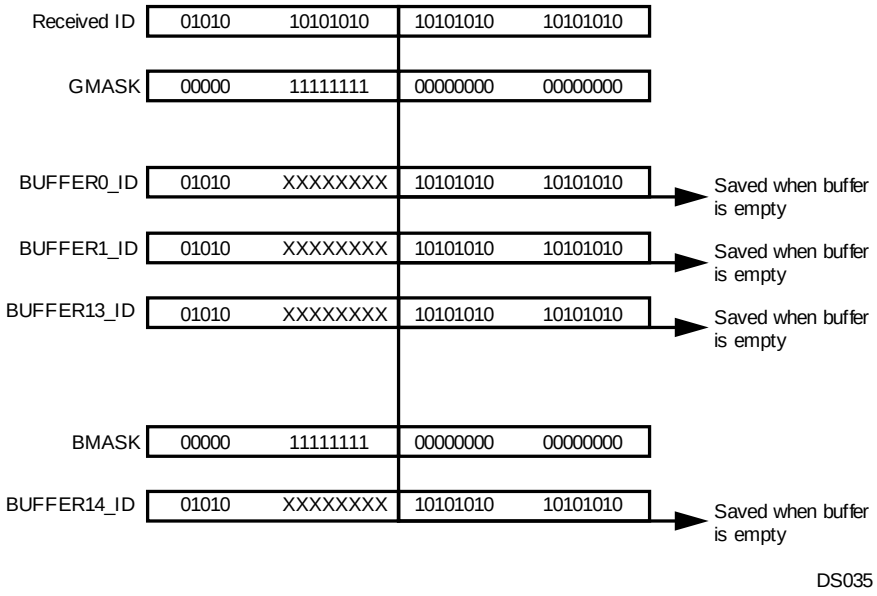


Figure 22-18. Message Storage with BUFFLOCK Enabled

22.5 Receive Structure

All received frames are initially buffered in a hidden receive buffer until the frame is valid. (The validation point for a received message is the next-to-last bit of the EOF.) The received identifier is then compared to every buffer ID together with the respective mask and the status. As soon as the validation point is reached, the whole contents of the hidden buffer are copied into the matching message buffer as shown in Figure 22-19.

NOTE

The hidden receive buffer must not be accessed by the CPU.

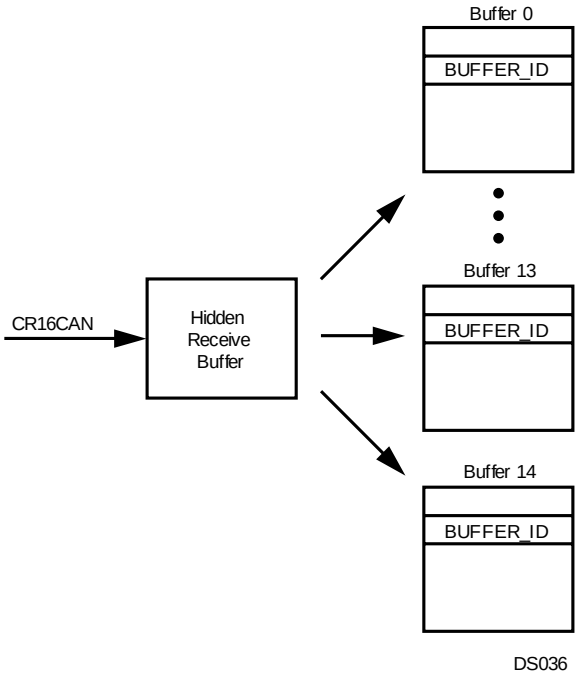


Figure 22-19. Receive Buffer

The following section gives an overview of the reception of the different types of frames.

The received data frame is stored in the first matching receive buffer beginning with buffer 0. For example, if the message is accepted by buffer 5, then at the time the message will be copied, the RX request is cleared and the CAN module will not try to match the frame to any subsequent buffer.

All contents of the hidden receive buffer are always copied into the respective receive buffer. This includes the received message ID as well as the received Data Length Code (DLC); therefore when some mask bits are set to don't care, the ID field will get the received message ID which could be different from the previous ID. The DLC of the receiving buffer will be updated by the DLC of the received frame. The DLC of the received message is not compared with the DLC already present in the CNSTAT register of the message buffer. This implies that the DLC code of the CNSTAT register indicates how may data bytes actually belong to the latest received message.

The remote frames are handled by the CAN interface in two different ways. In the first method, remote frames can be received like data frames by configuring the buffer to be RX_READY and setting the ID bits including the RTR bit. In that case, the same procedure applies as described for Data Frames. In the second method, a remote frame can trigger one or more message buffer to transmit a data frame upon reception. This procedure is described under section Section 22.6.

22.5.1 Receive Timing

As soon as a CAN module receives a "dominant" bit on the CAN bus, the receive process is started. The received ID and data will be stored in the hidden receive buffer if the global or basic acceptance filtering matches. After the reception of the data, CAN module tries to match the buffer ID of buffer 0...14. The data will be copied into the buffer after the reception of the 6th EOF bit as a message is valid at this time. The copy process of every frame, regardless of the length, takes at least 17 CKI cycles (see also Section 22.9.1). Figure 22-20 shows the receive timing.

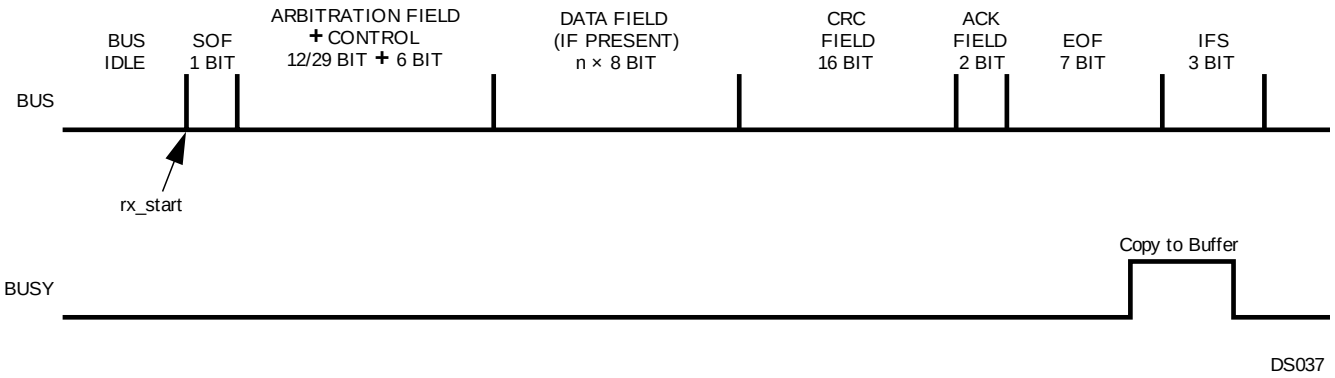


Figure 22-20. Receive Timing

To indicate that a frame is waiting in the hidden buffer, the BUSY bit (ST[0]) of the selected buffer is set during the copy procedure. The BUSY bit will be cleared by the CAN module immediately after the data bytes are copied into the buffer. After the copy process is finished, the CAN module changes the status field to RX_FULL. In turn, the CPU should change the status field to RX_READY when the data is processed. When a new object has been received by the same buffer, before the CPU changed the status to RX_READY, the CAN module will change the status to RX_OVERRUN to indicate that at least one frame has been overwritten by a new one. Table 22-2 summarizes the current status and the resulting update from the CAN module.

Table 22-2. Writing to Buffer Status Code During RX_BUSY

CURRENT STATUS	RESULTING STATUS
RX_READY	RX_FULL
RX_NOT_ACTIVE	RX_NOT_ACTIVE
RX_FULL	RX_OVERRUN

During the assertion of the BUSY bit, all writes to the receiving buffer are disabled with the exception of the status field. If the status is changed while the BUSY bit is asserted, the status is updated by the CAN module as shown in Table 22-2.

The buffer states are indicated and controlled by the ST[3:0] bits in the CNSTAT register (see Section 22.10.1). The various receive buffer states are explained in Section 22.5.3.

22.5.2 Receive Procedure

Software executes the following procedure to initialize a message buffer for the reception of a CAN message.

1. Configure the receive masks (GMASK or BMASK).
2. Configure the buffer ID.
3. Configure the message buffer status as RX_READY.

To read the out of a received message, the CPU must execute the following steps (see Figure 22-21):

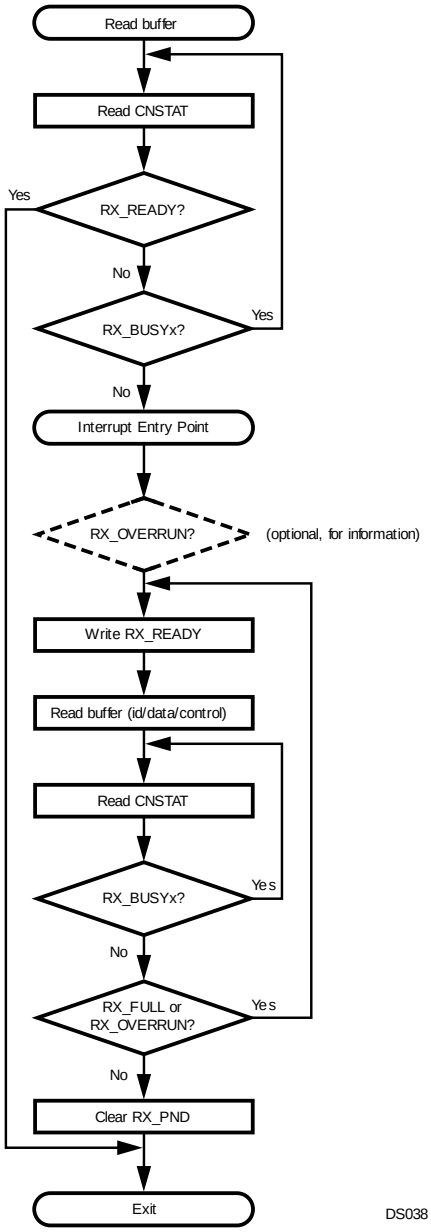


Figure 22-21. Buffer Read Routine (BUFFLOCK Disabled)

The first step is only applicable if polling is used to get the status of the receive buffer. It can be deleted for an interrupt driven receive routine.

1. Read the status (CNSTAT) of the receive buffer. If the status is RX_READY, no was the message received, so exit. If the status is RX_BUSY, the copy process from hidden receive buffer is not completed yet, so read CNSTAT again.

   If a buffer is configured to RX_READY and its interrupt is enabled, it will assert an interrupt as soon as the buffer has received a message and entered the RX_FULLstate (see also Section 22.7). In that case the procedure described below must be followed.
2. Read the status to determine if a new message has overwritten the one originally received which triggered the interrupt.
3. Write RX_READY into CNSTAT.
4. Read the ID/data and object control (DLC/RTR) from the message buffer.

5. Read the buffer status again and check it is not RX_BUSYx. If it is, repeat this step until RX_BUSYx has gone away.
6. If the buffer status is RX_FULL or RX_OVERRUN, one or more messages were copied. In that case, start over with step 2.
7. If status is still RX_READY (as set by the CPU at step 2), clear interrupt pending bit and exit.

When the BUFFLOCK function is enabled (see BUFFLOCK in section Section 22.4), it is not necessary to check for new messages received during the read process from the buffer, as this buffer is locked after the reception of the first valid frame. A read from a locked receive buffer can be performed as shown in Figure 22-22.

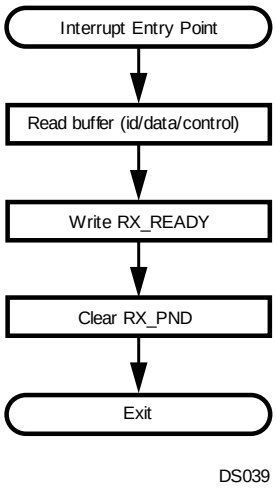


Figure 22-22. Buffer Read Routine (BUFFLOCK Enabled)

For simplicity only the applicable interrupt routine is shown:

1. Read the ID/data and object control (DLC/RTR) from the message buffer.
2. Write RX_READY into CNSTAT.
3. Clear interrupt pending bit and exit.

22.5.3 Rx Buffer States

As shown in Figure 22-22, a receive procedure starts as soon as software has set the buffer from the RX_NOT_ACTIVE state into the RX_READY state. The status section of CNSTAT register is set from 0000b to 0010b. When a message is received, the buffer will be RX_BUSYx during the copy process from the hidden receive buffer into the message buffer. Afterwards this buffer is RX_FULL. The CPU can then read the buffer data and either reset the buffer status to RX_READY or receive a new frame before the CPU reads the buffer. In the second case, the buffer state will automatically change to RX_OVERRUN to indicate that at least one message was lost. During the copy process the buffer will again be RX_BUSYx for a short time, but in this case the CNSTAT status section will be 0101b, as the buffer was RX_FULL (0100b) before. After finally reading the last received message, the CPU can reset the buffer to RX_READY.

22.6 Transmit Structure

To transmit a CAN message, software must configure the message buffer by changing the buffer status to TX_NOT_ACTIVE. The buffer is configured for transmission if the ST[3] bit of the buffer status code (CNSTAT) is set. In TX_NOT_ACTIVE status, the buffer is ready to receive data from the CPU. After receiving all transmission data (ID, data bytes, DLC, and PRI), the CPU can start the transmission by writing TX_ONCE into the buffer status register. During the transmission, the status of the buffer is TX_BUSYx. After successful transmission, the CAN module will reset the buffer status to TX_NOT_ACTIVE. If the transmission process fails, the buffer condition will remain TX_BUSYx for retransmission until the frame was successfully transmitted or the CPU has canceled the transmission request.

To Send a Remote Frame (Remote Transmission Request) to other CAN nodes, software sets the RTR bit of the message identifier (see Section 22.10.5) and changes the status of the message buffer to TX_ONCE. After this remote frame has been transmitted successfully, this message buffer will automatically enter the RX_READY state and is ready to receive the appropriate answer. Note that the mask bits RTR/XRTR need to be set to receive a data frame (RTR = 0) in a buffer which was configured to transmit a remote frame (RTR = 1).

To answer Remote Frames, the CPU writes TX_RTR in the buffer status register, which causes the buffer to wait for a remote frame. When a remote frame passes the acceptance filtering mask of one or more buffers, the buffer status will change to TX_ONCE_RTR, the contents of the buffer will be transmitted, and afterwards the CAN module will write TX_RTR in the status code register again.

If the CPU writes TX_ONCE_RTR into the buffer status, the contents of the buffer will be transmitted, and the successful transmission the buffer goes into the “wait for Remote Frame” condition TX_RTR.

22.6.1 Transmit Scheduling

After writing TX_ONCE into the buffer status, the transmission process begins and the BUSY bit is set. As soon as a buffer gets the TX_BUSY status, the buffer is no longer accessible by the CPU except for the ST[3:1] bits of the CNSTAT register. Starting with the beginning of the CRC field of the current frame, the CAN module looks for another buffer transmit request and selects the buffer with the highest priority for the next transmission by changing the buffer state from TX_ONCE to TX_BUSY. This transmit request can be canceled by the CPU or can be overwritten by another transmit request of a buffer with a higher priority as long as the transmission of the next frame has not yet started. This means that between the beginning of the CRC field of the current frame and the transmission start of the next frame, two buffers, the current buffer and the buffer scheduled for the next transmission, are in the BUSY status. To cancel the transmit request of the next frame, the CPU must change the buffer state to TX_NOT_ACTIVE. When the transmit request has been overwritten by another request of a higher priority buffer, the CAN module changes the buffer state from TX_BUSY to TX_ONCE. Therefore, the transmit request remains pending. Figure 22-23 further illustrates the transmit timing.

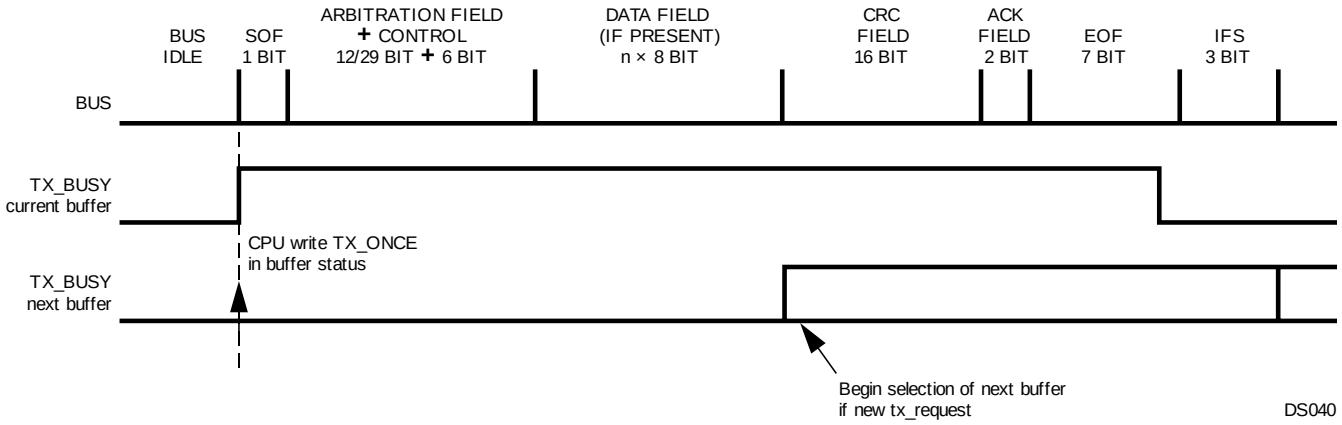


Figure 22-23. Data Transmission

If the transmit process fails or the arbitration is lost, the transmission process will be stopped and will continue after the interrupting reception or the error signaling has finished (see Figure 22-23). In that case, a new buffer select follows and the TX process is executed again.

NOTE

The canceled message can be delayed by a TX request of a buffer with a higher priority. While TX_BUSY is high, software cannot change the contents of the message buffer object. In all cases, writing to the BUSY bit will be ignored.

22.6.2 Transmit Priority

The CAN module is able to generate a stream of scheduled messages without releasing the bus between two messages so that an optimized performance can be achieved. It will arbitrate for the bus immediately after sending the previous message and will only release the bus due to a lost arbitration.

If more than one buffer is scheduled for transmission, the priority is built by the message buffer number and the priority code in the CNSTAT register. The 8-bit value of the priority is combined by the 4-bit TXPRI value and the 4-bit buffer number (0...14) as shown below. The lowest resulting number results in the highest transmit priority.

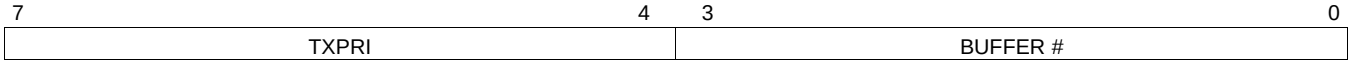


Table 22-3 shows the transmit priority configuration if the priority is TXPRI = 0 for all transmit buffers.

Table 22-3. Transmit Priority (TXPRI = 0)

TXPRI	BUFFER NUMBER	PRI	TX PRIORITY
0	0	0	Highest
0	1	1	
:	:	:	:
:	:	:	:
0	14	14	Lowest

Table 22-4 shows the transmit priority configuration if TXPRI is different from the buffer number.

Table 22-4. Transmit Priority (TXPRI not 0)

TXPRI	BUFFER NUMBER	PRI(1)	TX PRIORITY
14	0	224	Lowest
13	1	209	
12	2	194	
11	3	179	
10	4	164	
9	5	149	
8	6	134	
7	7	119	
6	8	104	
5	9	89	
4	10	74	
3	11	59	

(1) If two buffers have the same priority (PRI), the buffer with the lower buffer number will have the higher priority.

Table 22-4. Transmit Priority (TXPRI not 0) (continued)

TXPRI	BUFFER NUMBER	PRI(1)	TX PRIORITY
2	12	44	
1	13	29	
0	14	14	Highest

22.6.3 Transmit Procedure

The transmission of a CAN message must be executed as follows (see Figure 22-24)

1. Configure the CNSTAT status field as TX_NOT_ACTIVE. If the status is TX_BUSY, a previous transmit request is still pending and software has no access to the data contents of the buffer. In that case, software may choose to wait until the buffer becomes available again as shown. Other options are to exit from the update routine until the buffer has been transmitted with an interrupt generated, or the transmission is aborted by an error.
2. Load buffer identifier and data registers. (For remote frames the RTR bit of the identifier needs to be set and loading data bytes can be omitted.)
3. Configure the CNSTAT status field to the desired value:
   – TX_ONCE to trigger the transmission process of a single frame.
   – TX_ONCE_RTR to trigger the transmission of a single data frame and then wait for a received remote frame to trigger consecutive data frames.
   – TX_RTR waits for a remote frame to trigger the transmission of a data frame.

Writing TX_ONCE or TX_ONCE_RTR in the CNSTAT status field will set the internal transmit request for the CAN module.

If a buffer is configured as TX_RTR and a remote frame is received, the data contents of the addressed buffer will be transmitted automatically without further CPU activity.

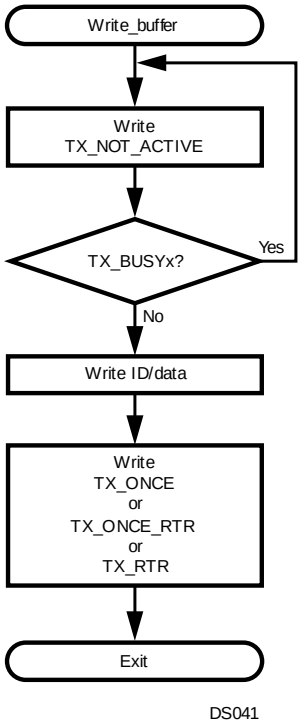


Figure 22-24. Buffer Write Routine

22.6.4 TX Buffer States

The transmission process can be started after software has loaded the buffer registers (data, ID, DLC, PRI) and set the buffer status from TX_NOT_ACTIVE to TX_ONCE, TX_RTR, or TX_ONCE_RTR.

When the CPU writes TX_ONCE, the buffer will be TX_BUSY as soon as the CAN module has scheduled this buffer for the next transmission. After the frame could be successfully transmitted, the buffer status will be automatically reset to TX_NOT_ACTIVE when a data frame was transmitted or to RX_READY when a remote frame was transmitted.

If the CPU configures the message buffer to TX_ONCE_RTR, it will transmit its data contents. During the transmission, the buffer state is 1111b as the CPU wrote 1110b into the status section of the CNSTAT register. After the successful transmission, the buffer enters the TX_RTR state and waits for a remote frame. When it receives a remote frame, it will go back into the TX_ONCE_RTR state, transmit its data bytes, and return to TX_RTR. If the CPU writes 1010b into the buffer status section, it will only enter the TX_RTR state, but it will not send its data bytes before it waits for a remote frame. Figure 22-25 illustrates the possible transmit buffer states.

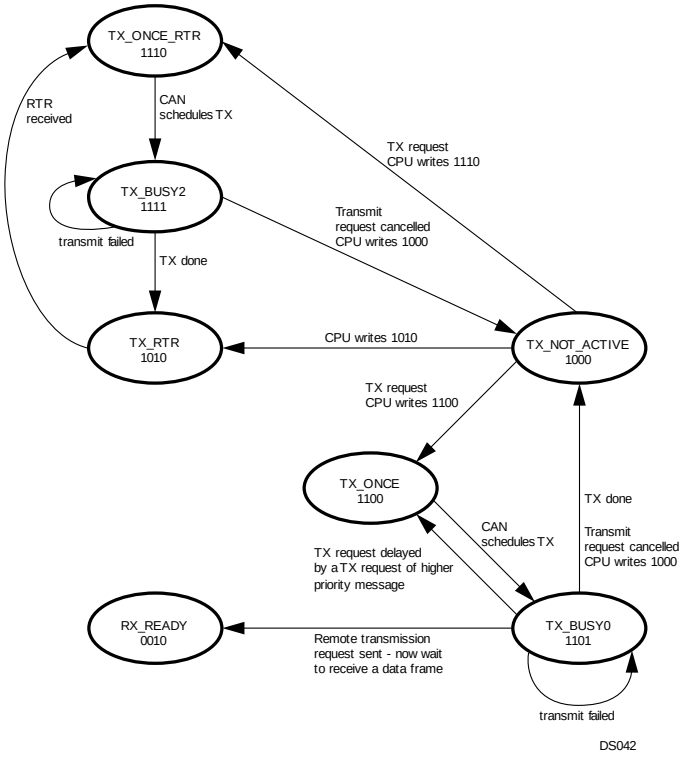


Figure 22-25. Transmit Buffer States

22.7 Interrupts

Each CAN module has one dedicated ICU interrupt vector for all interrupt conditions. In addition, the data frame receive event is an input to the MIWU (see Section 17). The interrupt process can be initiated from the following sources.

- CAN data transfer
  - Reception of a valid data frame in the buffer. (Buffer state changes from RX_READY to RX_FULL or RX_OVERRUN.)
  - Successful transmission of a data frame. (Buffer state changes from TX_ONCE to TX_NOT_ACTIVE or RX_READY.)
  - Successful response to a remote frame. (Buffer state changes from TX_ONCE_RTR to TX_RTR.)
  - Transmit scheduling. (Buffer state changes from TX_RTR to TX_ONCE_RTR.)

- CAN error conditions
  - Detection of an CAN error. (The CEIPND bit in the CIPND register will be set as well as the corresponding bits in the error diagnostic register CEDIAG.)

The receive/transmit interrupt access to every message buffer can be individually enabled/disabled in the CIEN register. The pending flags of the message buffer are located in the CIPND register (read only) and can be cleared by resetting the flags in the CICLR registers.

22.7.1 Highest Priority Interrupt Code

To reduce the decoding time for the CIPND register, the buffer interrupt request with the highest priority is placed as interrupt status code into the IST[3:0] section of the CSTPND register.

Each of the buffer interrupts as well as the error interrupt can be individually enabled or disabled in the CAN Interrupt Enable register (CIEN). As soon as an interrupt condition occurs, every interrupt request is indicated by a flag in the CAN Interrupt Pending register (CIPND). When the interrupt code logic for the present highest priority interrupt request is enabled, this interrupt will be translated into the IST3:0 bits of the CAN Status Pending register (CSTPND). An interrupt request can be cleared by setting the corresponding bit in the CAN Interrupt Clear register (CICLR).

Figure 22-26 shows the CAN interrupt management.

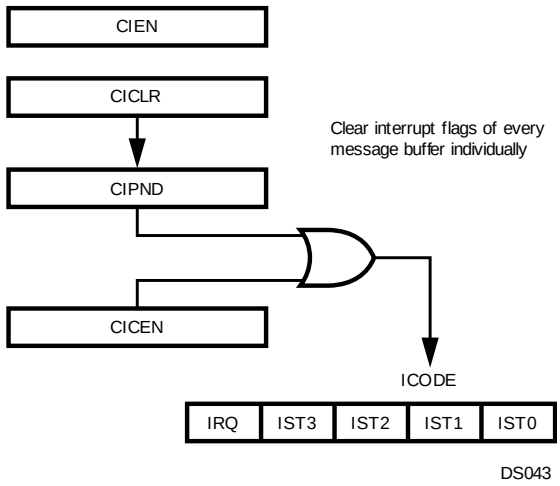


Figure 22-26. Interrupt Management

The highest priority interrupt source is translated into the bits IRQ and IST3:0 as shown in Table 22-5.

Table 22-5. Highest Priority Interrupt Code (ICEN=FFFF)

CAN INTERRUPT REQUEST	IRQ	IST3	IST2	IST1	IST0
No Request	0	0	0	0	0
Error Interrupt	1	0	0	0	0
Buffer 0	1	0	0	0	1
Buffer 1	1	0	0	1	0
Buffer 2	1	0	0	1	1
Buffer 3	1	0	1	0	0
Buffer 4	1	0	1	0	1
Buffer 5	1	0	1	1	0
Buffer 6	1	0	1	1	1
Buffer 7	1	1	0	0	0
Buffer 8	1	1	0	0	1

Table 22-5. Highest Priority Interrupt Code (ICEN=FFFF) (continued)

CAN INTERRUPT REQUEST	IRQ	IST3	IST2	IST1	IST0
Buffer 9	1	1	0	1	0
Buffer 10	1	1	0	1	1
Buffer 11	1	1	1	0	0
Buffer 12	1	1	1	0	1
Buffer 13	1	1	1	1	0
Buffer 14	1	1	1	1	1

22.7.2 Usage Notes

The interrupt code IST3:0 can be used within the interrupt handler as a displacement to jump to the relevant subroutine.

The CAN Interrupt Code Enable (CICEN) register is used in the CAN interrupt handler if software is servicing all receive buffer interrupts first, followed by all transmit buffer interrupts. In this case, software can first enable only receive buffer interrupts to be coded, then scan and service all pending interrupt requests in the order of their priority. After processing all the receive interrupts, software changes the CICEN register to disable all receive buffers and enable all transmit buffers, then services all pending transmit buffer interrupt requests according to their priorities.

22.8 Time Stamp Counter

Each CAN module features a free running 16-bit timer (CTMR) incrementing every bit time recognized on the CAN bus. The value of this timer during the ACK slot is captured into the TSTP register of a message buffer after a successful transmission or reception of a message. Figure 22-27 shows a simplified block diagram of the Time Stamp counter.

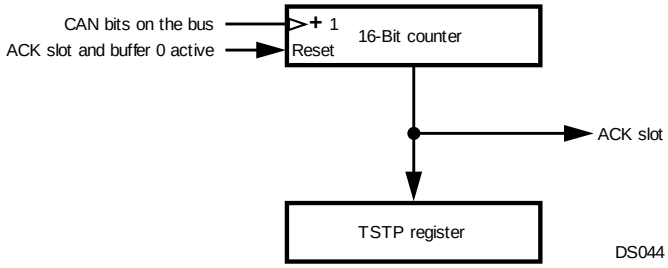


Figure 22-27. Time Stamp Counter

The timer can be synchronized over the CAN network by receiving or transmitting a message to or from buffer 0. In this case, the TSTP register of buffer 0 captures the current CTMR value during the ACK slot of a message (as above), and then the CTMR is reset to 0000b. Synchronization can be enabled or disabled using the CGCR.TSTPEN bit.

22.9 Memory Organization

Each CAN module occupies 288 words in the memory address space. This space is organized as 15 banks of 16 words per bank (plus one 16-word reserved bank) for the message buffers and 28 words (plus 4 reserved words) for control and status.

22.9.1 CPU Access to CAN Registers/Memory

All memory locations occupied by the message buffers are shared by the CPU and CAN modules, with CPU access having priority over CAN access. The CAN modules and the CPU normally have single-cycle access to this memory. However, if an access contention occurs, the access to the memory is blocked every cycle until the contention is resolved. This access arbitration is transparent to software.

If a buffer is busy during the reception of an object (copy process from the hidden receive buffer) or is scheduled for transmission, the CPU has no write access to the data contents of the buffer. Write to the status/control byte and read access to the whole buffer is always enabled.

All configuration and status registers can either be accessed by the CAN module or the CPU. These registers provide single-cycle access without any potential wait state.

All register descriptions within the next sections have the following layout:

15	0
Bit/Field Names	
Reset Value	
CPU Access (R = read only, W = write only, R/W = read/write)	

The CAN peripheral registers and shared memory are implemented as 16-bit words which occupy the lower half of 32-bit doubleword locations in the address space. The upper half is reserved.

22.9.2 Message Buffer Organization

The message buffers are the communication interfaces between CAN and the CPU for the transmission and the reception of CAN frames. There are 15 message buffers located at fixed addresses in RAM. As shown in Table 22-6, each buffer consists of two words reserved for the identifiers, 4 words reserved for up to eight CAN data bytes, one word reserved for the time stamp, and one word for data length code, transmit priority code, and the buffer status codes.

Table 22-6. Message Buffer Map

Address	Buffer Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE + 00h	CNSTAT	DLC					Reserved				PRI			ST			
BASE + 04h	TSTP	TSTP[15:0]															
BASE + 08h	DATA3	Data7[7:0]									Data8[7:0]						
BASE + 0Ch	DATA2	Data5[7:0]									Data6[7:0]						
BASE + 10h	DATA1	Data3[7:0]									Data4[7:0]						
BASE + 14h	DATA0	Data1[7:0]									Data2[7:0]						
BASE + 18h	ID0	XI[14:0]															RTR
BASE + 1Ch	ID1	XI[28:18]/ID[10:0]											SRR/RTR	IDE	XI[17:15]		

22.10 CAN Controller Registers

Table 22-7 lists the registers for the two CAN modules (CAN0 and CAN1).

Table 22-7. CAN Controller Registers

NAME	ADDRESS	DESCRIPTION
CNSTAT	See Table 22-6	Message Buffer Status/Control Register
CGCR0	FF BA00h	CAN0 Global Configuration Register
CGCR1	FF BE00h	CAN1 Global Configuration Register
CTIM0	FF BA04h	CAN0 Timing Register
CTIM1	FF BE04h	CAN1 Timing Register
GMSKX0	FF BA08h	CAN0 Global Mask Register
GMSKX1	FF BE08h	CAN1 Global Mask Register
GMSKB0	FF BA0Ch	CAN0 Global Mask Register
GMSKB1	FF BE0Ch	CAN1 Global Mask Register
BMSKX0	FF BA10h	CAN0 Basic Mask Register

Table 22-7. CAN Controller Registers (continued)

NAME	ADDRESS	DESCRIPTION
BMSKX1	FF BE10h	CAN1 Basic Mask Register
BMSKB0	FF BA14h	CAN0 Basic Mask Register
BMSKB1	FF BE14h	CAN1 Basic Mask Register
CIEN0	FF BA18h	CAN0 Interrupt Enable Register
CIEN1	FF BE18h	CAN1 Interrupt Enable Register
CIPND0	FF BA1Ch	CAN0 Interrupt Pending Register
CIPND1	FF BE1Ch	CAN1 Interrupt Pending Register
CICLR0	FF BA20h	CAN0 Interrupt Clear Register
CICLR1	FF BE20h	CAN1 Interrupt Clear Register
CICEN0	FF BA24h	CAN0 Interrupt Code Enable Register
CICEN1	FF BE24h	CAN1 Interrupt Code Enable Register
CSTPND0	FF BA28h	CAN0 Status Pending Register
CSTPND1	FF BE28h	CAN1 Status Pending Register
CANEC0	FF BA2Ch	CAN0 Error Counter Register
CANEC1	FF BE2Ch	CAN1 Error Counter Register
CEDIAG0	FF BA30h	CAN0 Error Diagnostic Register
CEDIAG1	FF BE30h	CAN1 Error Diagnostic Register
CTMR0	FF BA34h	CAN0 Timer Register
CTMR1	FF BE34h	CAN1 Timer Register

22.10.1 Message Buffer Status/Control Register (CNSTAT)

The buffer status (ST), the buffer priority (PRI), and the data length code (DLC) are controlled by manipulating the contents of the Buffer Status/Control Register (CNSTAT). The CPU and CAN module have access to this register.

15 – 12	11 – 8	7 – 4	3 – 0
DLC	Reserved	PRI	ST
0			
R/W			

ST The Buffer Status field contains the status information of the buffer as shown in Table 22-8. This field can be modified by the CAN module. The ST0 bits acts as a buffer busy indication. When the BUSY bit is set, any write access to the buffer is disabled with the exception of the lower byte of the CNSTAT register. The CAN module sets this bit if the buffer data is currently copied from the hidden buffer or if a message is scheduled for transmission or is currently transmitting. The CAN module always clears this bit on a status update.

Table 22-8. Buffer Status Section of the CNSTAT Register

ST3 (DIR)	ST2	ST1	ST0 (BUSY)	BUFFER STATUS
0	0	0	0	RX_NOT_ACTIVE
0	0	0	1	Reserved for RX_BUSY. (This condition indicates that software wrote RX_NOT_ACTIVE to a buffer when the data copy process is still active.)
0	0	1	0	RX_READY
0	0	1	1	RX_BUSY0 (Indicates data is being copied for the first time RX_READY → RX_BUSY0.)
0	1	0	0	RX_FULL
0	1	0	1	RX_BUSY1 (Indicates data is being copied for the second time RX_FULL → RX_BUSY1.)
0	1	1	0	RX_OVERRUN
0	1	1	1	RX_BUSY2 (Indicates data is being copied for the third or subsequent times RX_OVERRUN → RX_BUSY2.)

Table 22-8. Buffer Status Section of the CNSTAT Register (continued)

ST3 (DIR)	ST2	ST1	ST0 (BUSY)	BUFFER STATUS
1	0	0	0	TX_NOT_ACTIVE
1	0	0	1	Reserved for TX_BUSY. (This state indicates that software wrote TX_NOT_ACTIVE to a transmit buffer which is scheduled for transmission or is currently transmitting.)
1	1	0	0	TX_ONCE
1	1	0	1	TX_BUSY0 (Indicates that a buffer is scheduled for transmission or is actively transmitting; it can be due to one of two cases: a message is pending for transmission or is currently transmitting, or an automated answer is pending for transmission or is currently transmitting.)
1	0	1	0	TX_RTR (Automatic response to a remote frame.)
1	0	1	1	Reserved for TX_BUSY1. (This condition does not occur.)
1	1	1	0	TX_ONCE_RTR (Changes to TX_RTR after transmission.)
1	1	1	1	TX_BUSY2 (Indicates that a buffer is scheduled for transmission or is actively transmitting; it can be due to one of two cases: a message is pending for transmission or is currently transmitting, or an automated answer is pending for transmission or is currently transmitting.)

PRI The Transmit Priority Code field holds the software-defined transmit priority code for the message buffer.

DLC The Data Length Code field determines the number of data bytes within a received/transmitted frame. For transmission, these bits need to be set according to the number of data bytes to be transmitted. For reception, these bits indicate the number of valid received data bytes available in the message buffer. Table 22-9 shows the possible bit combinations for DLC3:0 for data lengths from 0 to 8 bytes.

Table 22-9. Data Length Coding

DLC	Number of Data Bytes(1)
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8

(1) The maximum number of data bytes received/transmitted is 8, even if the DLC field is set to a value greater than 8. Therefore, if the data length code is greater or equal to eight bytes, the DLC field is ignored.

22.10.2 Storage of Standard Messages

During the processing of standard frames, the Extended-Identifier (IDE) bit is clear. The ID1[3:0] and ID0[15:0] bits are “don’t care” bits. A standard frame with eight data bytes is shown in Table 22-10.

IDE The Identifier Extension bit determines whether the message is a standard frame or an extended frame.
0 – Message is a standard frame using 11 identifier bits.
1 – Message is an extended frame.

RTR The Remote Transmission Request bit indicates whether the message is a data frame or a remote frame.
0 – Message is a data frame.
1 – Message is a remote frame.

ID The ID field is used for the 11 standard frame identifier bits.

Table 22-10. Standard Frame with 8 Data Bytes

Address	Buffer Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE + 00h	CNSTAT	DLC				Reserved				PRI				ST			
BASE + 04h	TSTP	TSTP[15:0]															
BASE + 08h	DATA3	Data7[7:0]								Data8[7:0]							
BASE + 0Ch	DATA2	Data5[7:0]								Data6[7:0]							
BASE + 10h	DATA1	Data3[7:0]								Data4[7:0]							
BASE + 14h	DATA0	Data1[7:0]								Data2[7:0]							
BASE + 18h	ID0	Don't Care															
BASE + 1Ch	ID1	ID[10:0]											RTR	IDE	Don't Care		

22.10.3 Storage of Messages with Less Than 8 Data Bytes

The data bytes that are not used for data transfer are “don’t cares”. If the object is transmitted, the data within these bytes will be ignored. If the object is received, the data within these bytes will be overwritten with invalid data.

22.10.4 Storage of Extended Messages

If the IDE bit is set, the buffer handles extended frames. The storage of the extended ID follows the descriptions in Table 22-11. The SRR bit is at the bit position of the RTR bit for standard frame and needs to be transmitted as 1.

Table 22-11. Extended Messages with 8 Data Bytes

Address	Buffer Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE + 00h	CNSTAT	DLC				Reserved				PRI				ST			
BASE + 04h	TSTP	TSTP															
BASE + 08h	DATA3	Don't Care															
BASE + 0Ch	DATA2																
BASE + 10h	DATA1																
BASE + 14h	DATA0																
BASE + 18h	ID0	ID[14:0]															RTR
BASE + 1Ch	ID1	ID[28:18]											SRR	IDE	ID17:15]		

SRR The Substitute Remote Request bit replaces the RTR bit used in standard frames at this bit position. The SRR bit needs to be set by software.

IDE The Identifier Extension bit determines whether the message is a standard frame or an extended frame.
0 – Message is a standard frame using 11 identifier bits.
1 – Message is an extended frame.

RTR The Remote Transmission Request bit indicates whether the message is a data frame or a remote frame.
0 – Message is a data frame.
1 – Message is a remote frame.

ID The ID field is used to build the 29-bit identifier of an extended frame. The ID[28:18] field is used for the 11 standard frame identifier bits.

22.10.5 Storage of Remote Messages

During remote frame transfer, the buffer registers DATA0–DATA3 are “don’t cares”. If a remote frame is transmitted, the contents of these registers are ignored. If a remote frame is received, the contents of these registers will be overwritten with invalid data. The structure of a message buffer set up for a remote frame with extended identifier is shown in Table 22-12.

Table 22-12. Extended Remote Frame

Address	Buffer Register	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE + 00h	CNSTAT	DLC				Reserved				PRI				ST			
BASE + 04h	TSTP	TSTP[15:0]															
BASE + 08h	DATA3	Data7[7:0]								Data8[7:0]							
BASE + 0Ch	DATA2	Data5[7:0]								Data6[7:0]							
BASE + 10h	DATA1	Data3[7:0]								Data4[7:0]							
BASE + 14h	DATA0	Data1[7:0]								Data2[7:0]							
BASE + 18h	ID0	ID[14:0]															RTR
BASE + 1Ch	ID1	ID[28:18]											SRR	IDE	ID17:15]		

SRR The Substitute Remote Request bit replaces the RTR bit used in standard frames at this bit position. The SRR bit needs to be set by software if the buffer is configured to transmit a message with an extended identifier. It will be received as monitored on the CAN bus.

IDE The Identifier Extension bit determines whether the message is a standard frame or an extended frame.
0 – Message is a standard frame using 11 identifier bits.
1 – Message is an extended frame.

RTR The Remote Transmission Request bit indicates whether the message is a data frame or a remote frame.
0 – Message is a data frame.
1 – Message is a remote frame.

ID The ID field is used to build the 29-bit identifier of an extended frame.

22.10.6 CAN Global Configuration Register n (CGCRn)

The CGCRn register is a 16-bit, read/write register used to:

- Enable/disable the CAN module.
- Configure the BUFFLOCK function for the message buffer 0..14.
- Enable/disable the time stamp synchronization.
- Set the logic levels of the CAN Input/Output pins, CANnRX and CANnTX.
- Choose the data storage direction (DDIR).
- Select the error interrupt type (EIT).
- Enable/disable diagnostic functions.

15 12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	EIT	DIAGEN	INTERNAL	LOOPBACK	IGNACK	LO	DDIR	TSTPEN	BUFFLOCK	CRX	CTX	CANEN
0					0							
R/W					R/W							

CANEN — The CAN Enable bit enables/disables the CAN module. When the CAN module is disabled, all internal states and the TEC and REC counter registers are cleared. In addition the CAN module clock is disabled. All CAN module control registers and the contents of the object memory are left unchanged. Software must make sure that no message is pending for transmission before the CAN module is disabled.
0 – CAN module is disabled.
1 – CAN module is enabled.

CTX — The Control Transmit bit configures the logic level of the CAN transmit pin CANnTX.
0 – Dominant state is 0; recessive state is 1.
1 – Dominant state is 1; recessive state is 0.

CRX — The Control Receive bit configures the logic level of the CAN receive pin CANnRX.
0 – Dominant state is 0; recessive state is 1.
1 – Dominant state is 1; recessive state is 0.

BUFFLOCK — The Buffer Lock bit configures the buffer lock function. If this feature is enabled, a buffer will be locked upon a successful frame reception. The buffer will be unlocked again by writing RX_READY in the buffer status register, i.e., after reading data.
0 – Lock function is disabled for all buffers.
1 – Lock function is enabled for all buffers.

TSTPEN — The Time Stamp Enable bit enables or disables the time stamp synchronization function of the CAN module.
0 – Time synchronization disabled. The Time Stamp counter value is not reset upon reception or transmission of a message to/ from buffer 0.
1 – Time synchronization enabled. The Time Stamp counter value is reset upon reception or transmission of a message to/from buffer 0.

DDIR — The Data Direction bit selects the direction the data bytes are transmitted and received. The CAN module transmits and receives the CAN Data1 byte first and the Data8 byte last (Data1, Data2,...,Data7, Data8). If the DDIR bit is clear, the data contents of a received message is stored with the first byte at the highest data address and the last data at the lowest data address (see Figure 69). The same applies for transmitted data.
0 – First byte at the highest address, subsequent bytes at lower addresses.
1 – First byte at the lowest address, subsequent bytes at higher addresses.

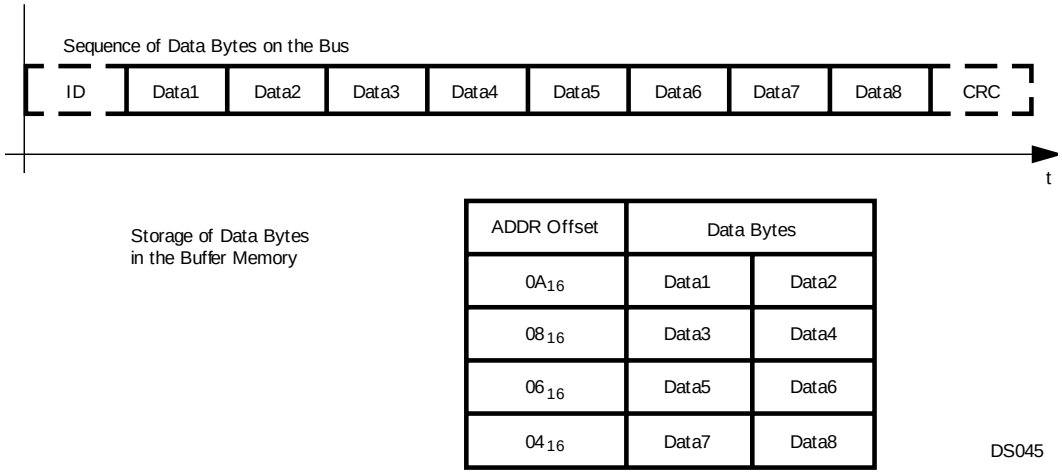


Figure 22-28. Data Direction Bit Clear

Setting the DDIR bit will cause the direction of the data storage to be reversed — the last byte received is stored at the highest address and the first byte is stored at the lowest address, as shown in Figure 22-29.

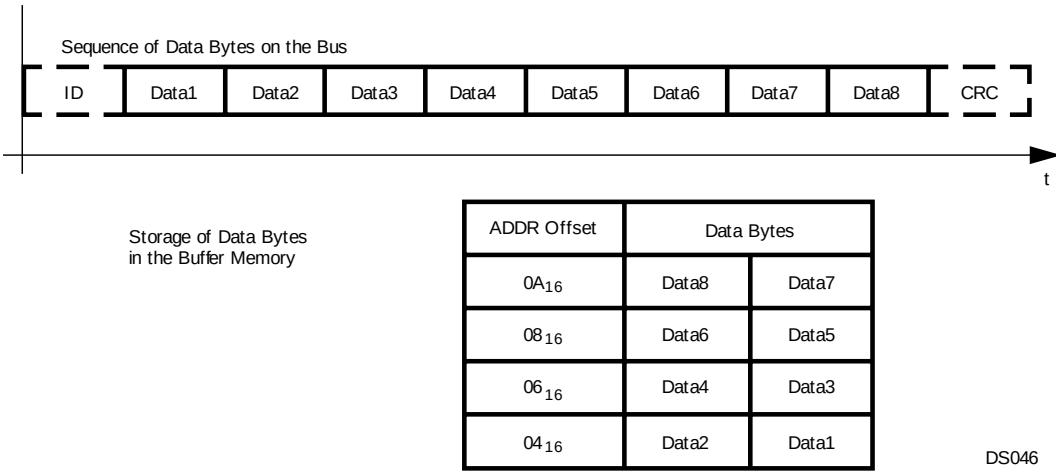


Figure 22-29. Data Direction Bit Set

LO — The Listen Only bit can be used to configure the CAN interface to behave only as a receiver. This means:

- Cannot transmit any message.
- Cannot send a dominant ACK bit
- When errors are detected on the bus, the CAN module will behave as in the error passive mode.

Using this listen only function, the CAN interface can be adjusted for connecting to an operating network with unknown bus speed.
0 – Transmit/receive mode.
1 – Listen-only mode.

When the Ignore Acknowledge bit is set, the CAN module does not expect to receive a dominant ACK bit to indicate the validity of a transmitted message. It will not send an error frame when the transmitted frame is not acknowledged by any other CAN node. This feature can be used in conjunction with the LOOPBACK bit for stand-alone tests outside of a CAN network.
0 – Normal mode.
1 – The CAN module does not expect to receive a dominant ACK bit to indicate the validity of a transmitted message.

LOOPBACK — When the Loopback bit is set, all messages sent by the CAN module can also be received by a CAN module buffer with a matching buffer ID. However, the CAN module does not acknowledge a message sent by itself. Therefore, the CAN module will send an error frame when no other device connected to the bus has acknowledged the message.
0 – No loopback.
1 – Loopback enabled.

INTERNAL — If the Internal function is enabled, the CANnTX and CANnRX pins of the CAN module are internally connected to each other. This feature can be used in conjunction with the LOOPBACK mode. This means that the CAN module can receive its own sent messages without connecting an external transceiver chip to the CANnTX and CANnRX pins; it allows software to run real stand-alone tests without any peripheral devices.
0 – Normal mode.
1 – Internal mode.

DIAGEN — The Diagnostic Enable bit globally enables or disables the special diagnostic features of the CAN module. This includes the following functions:

- LO (Listen Only)

- IGNACK (Ignore Acknowledge).
- LOOPBACK (Loopback)
- INTERNAL (Internal Loopback).
- Write access to hidden receive buffer.

0 – Normal mode.
1 – Diagnostic features enabled.

EIT The Error Interrupt Type bit controls the behavior of the Error Interrupt Pending bit (CIPNDn.EIPND) when an error Interrupt is asserted, if enabled by the Error Interrupt Enable bit (CIENn.EIEN).

0 – The EIPND bit is set on every error on the CAN bus.

1 – The EIPND bit is set only if the error state (CSTPNDn.NS) changes as a result of incrementing either the receive or transmit error counter.

22.10.7 CAN Timing Register n (CTIMn)

The CTIMn register defines the configuration of the Bit Time Logic (BTL).

15 9	8 7	6 3	2 0
PSC	SJW	TSEG1	TSEG2
0			
R/W			

PSC The Prescaler Configuration field specifies the CAN prescaler. The settings are shown in Table 22-13.

Table 22-13. CAN Prescaler Settings

PSC6:0	Prescaler
000000	2
000001	3
000010	4
000011	5
000100	6
:	:
1111101	127
1111110	128
1111111	128

SJW The Synchronization Jump Width field specifies the Synchronization Jump Width, which can be programmed between 1 and 4 time quanta (see Table 22-14).

Table 22-14. SJW Setting

SJW(1)	Synchronization Jump Width (SJW)
00	1 time quantum
01	2 time quanta
10	3 time quanta
11	4 time quanta

(1) The settings of SJW must be configured to be smaller or equal to TSEG1 and TSEG2

TSEG1 The Time Segment 1 field configures the length of the Time Segment 1 (TSEG1). It is not recommended to configure the time segment 1 to be smaller than 2 time quanta. (see Table 22-15).

Table 22-15. Time Segment 1 Settings

TSEG1[3:0]	Length of Time (TSEG1)
0000	Not recommended
0001	2 time quanta
0010	3 time quanta
0011	4 time quanta
0100	5 time quanta
0101	6 time quanta
0110	7 time quanta
0111	8 time quanta
1000	9 time quanta
1001	10 time quanta
1010	11 time quanta
1011	12 time quanta
1100	13 time quanta
1101	14 time quanta
1110	15 time quanta
1111	16 time quanta

TSEG2 The Time Segment 2 field specifies the number of time quanta (tq) for phase segment 2 (see Table 22-16).

Table 22-16. Time Segment 2 Settings

TSEG2	Length of TSEG2
000	1 time quantum
001	2 time quanta
010	3 time quanta
011	4 time quanta
100	5 time quanta
101	6 time quanta
110	7 time quanta
111	8 time quanta

22.10.8 CAN Global Mask Register n (GMSKBn/GMSKXn)

The GMSKBn and GMSKXn registers allow software to globally mask, or “don’t care” the incoming extended/standard identifier bits, RTR/XRTR and IDE. Throughout this document, the GMSKBn and GMSKXn 16-bit registers are referenced as a 32-bit register GMSK.

The following are the bits for the GMSKBn register.

15	5	4	3	2	0
GM[28:18]		RTR	IDE	GM[17:15]	
0					
R/W					

The following are the bits for the GMSKXn register.

15	1	0
GM[14:0]		XRTR
0		
R/W		

For all GMSKBn and GMSKXn register bits, the following applies:

0 – The incoming identifier bit must match the corresponding bit in the message buffer identifier register.

1 – Accept 1 or 0 (“don’t care”) in the incoming ID bit independent from the corresponding bit in the message buffer ID registers. The corresponding ID bit in the message buffer will be overwritten by the incoming identifier bits.

When an extended frame is received from the CAN bus, all GMSKn bits GM28:0, IDE, RTR, and XRTR are used to mask the incoming message. In this case, the RTR bit in the GMSKn register corresponds to the SRR bit in the message. The XRTR bit in the GMSKn register corresponds to the RTR bit in the message.

During the reception of standard frames only the GMSKn bits GM28:18, RTR, and IDE are used. In this case, the GM28:18 bits in the GMSKn register correspond to the ID10:0 bits in the message.

Global Mask	GM28:18	RTR	IDE	GM17:0	XRTR
Standard Frame	ID10:0	RTR	IDE	Unused	
Extended Frame	ID28:18	SRR	IDE	ID17:0	RTR

22.10.9 CAN Basic Mask Register n (BMSKBn/BMSKXn)

The BMSKBn and BMSKXn registers allow masking the buffer 14, or “don’t care” the incoming extended/standard identifier bits, RTR/XRTR, and IDE. Throughout this document, the two 16-bit registers BMSKBn and BMSKXn are referenced as a 32-bit register BMSKn.

The following are the bits for the BMSKXn register.

15	5	4	3	2	0
bM[28:18]		RTR	IDE	BM[17:15]	
0					
R/W					

The following are the bits for the BMSKXn register.

15	1	0
BM[14:0]		XRTR
0		
R/W		

For all BMSKBn and BMSKXn register bits the following applies:

0 – The incoming identifier bit must match the corresponding bit in the message buffer identifier register.

1 – Accept 1 or 0 (“don’t care”) in the incoming ID bit independent from the corresponding bit in the message buffer ID registers. The corresponding ID bit in the message buffer will be overwritten by the incoming identifier bits.

When an extended frame is received from the CAN bus, all BMSKn bits BM28:0, IDE, RTR, and XRTR are used to mask the incoming message. In this case, the RTR bit in the BMSKn register corresponds to the SRR bit in the message. The XRTR bit in the BMSKn register corresponds to the RTR bit in the message.

During the reception of standard frames, only the BMSKn bits BM28:18, RTR, and IDE are used. In this case, the BM28:18 bits in the BMSKn register correspond to the ID10:0 bits in the message.

Basic Mask	BM28:18	RTR	IDE	BM17:0	XRTR
Standard Frame	ID10:0	RTR	IDE	Unused	
Extended Frame	ID28:18	SRR	IDE	ID17:0	RTR

22.10.10 CAN Interrupt Enable Register n (CIENn)

The CIENn register enables the transmit/receive interrupts of the message buffers 0 through 14 as well as the CAN Error Interrupt.

15	14 ... 0
EIEN	IEN
0	
R/W	

EIEN The Error Interrupt Enable bit allows the CAN module to interrupt the CPU if any kind of CAN receive/transmit errors are detected. This causes any error status change in the error counter registers REC/TEC is able to generate an error interrupt.
0 – The error interrupt is disabled and no error interrupt will be generated.
1 – The error interrupt is enabled and a change in REC/TEC will cause an interrupt to be generated.

IEN The Buffer Interrupt Enable bits allow software to enable/disable the interrupt source for the corresponding message buffer. For example, IEN14 controls interrupts from buffer14, and IEN0 controls interrupts from buffer0.
0 – Buffer as interrupt source disabled.
1 – Buffer as interrupt source enabled.

22.10.11 CAN Interrupt Pending Register n (CIPNDn)

The CIPNDn register indicates any CAN Receive/Transmit Interrupt Requests caused by the message buffers 0..14 and CAN error occurrences.

15	14 ... 0
EIPND	IPND
0	
R	

EIPND The Error Interrupt Pending field indicates the status change of TEC/REC and will execute an error interrupt if the EIEN bit is set. Software has the responsibility to clear the EIPND bit using the CICLRn register.
0 – CAN status is not changed.
1 – CAN status is changed.

IPND The Buffer Interrupt Pending bits are set by the CAN module following a successful transmission or reception of a message to or from the corresponding message buffer. For example, IPND14 corresponds to buffer14, and IPND0 corresponds to buffer0.
0 – No interrupt pending for the corresponding message buffer.
1 – Message buffer has generated an interrupt.

22.10.12 CAN Interrupt Clear Register n (CICLRn)

The CICLRn register bits individually clear CAN interrupt pending flags caused by the message buffers and from the Error Management Logic. Do not modify this register with instructions that access the register as a read-modify-write operand, such as the bit manipulation instructions.

15	14	0
EICLR	ICLR	
0		
W		

EICLR The Error Interrupt Clear bit is used to clear the EIPND bit.
0 – The EIPND bit is unaffected by writing 0.
1 – The EIPND bit is cleared by writing 1.

ICLR The Buffer Interrupt Clear bits are used to clear the IPND bits.
0 – The corresponding IPND bit is unaffected by writing 0.
0 – The corresponding IPND bit is cleared by writing 1.

22.10.13 CAN Interrupt Code Enable Register n (CICENn)

The CICENn register controls whether the interrupt pending flag in the CIPND register is translated into the Interrupt Code field of the CSTPND register. All interrupt requests, CAN error, and message buffer interrupts can be enabled/ disabled separately for the interrupt code indication field.

15	14	0
EICEN	ICEN	
0		
R/W		

EICEN The Error Interrupt Code Enable bit controls encoding for error interrupts.
0 – Error interrupt pending is not indicated in the interrupt code.
1 – Error interrupt pending is indicated in the interrupt code.

ICEN The Buffer Interrupt Code Enable bits control encoding for message buffer interrupts.
0 – Message buffer interrupt pending is not indicated in the interrupt code.
1 – Message buffer interrupt pending is indicated in the interrupt code.

22.10.14 CAN Status Pending Register n (CSTPNDn)

The CSTPNDn register holds the status of the CAN Node and the Interrupt Code.

15	8	7	5	4	3	0
Reserved		NS		IRQ	IST	
0						
R						

NS The CAN Node Status field indicates the status of the CAN node as shown in Table 22-17.

Table 22-17. CAN Node Status

NS	Node Status
000	Not Active
010	Error Active
011	Error Warning Level
10x	Error Passive
11x	Bus Off

IRQ/IST The IRQ bit and IST field indicate the interrupt source of the highest priority interrupt currently pending and enabled in the CICENn register. Table 22-18 shows the several interrupt codes when the encoding for all interrupt sources is enabled (CICEN = FFFFh).

Table 22-18. Highest Priority Interrupt Code

IRQ	IST3:0	CAN Interrupt Request
0	0000	No interrupt request
1	0000	Error interrupt
1	0001	Buffer 0
1	0010	Buffer 1
1	0011	Buffer 2
1	0100	Buffer 3
1	0101	Buffer 4
1	0110	Buffer 5
1	0111	Buffer 6
1	1000	Buffer 7
1	1001	Buffer 8
1	1010	Buffer 9
1	1011	Buffer 10
1	1100	Buffer 11
1	1101	Buffer 12
1	1110	Buffer 13
1	1111	Buffer 14

22.10.15 CAN Error Counter Register n (CANECn)

The CANECn register reports the values of the CAN Receive Error Counter and the CAN Transmit Error Counter.

15	8	7	0
REC		TEC	
0			
R			

REC The CAN Receive Error Counter field reports the value of the receive error counter.

TEC The CAN Transmit Error Counter field reports the value of the transmit error counter.

22.10.16 CAN Error Diagnostic Register n (CEDIAGn)

The CEDIAGn register reports information about the last detected error. The CAN module identifies the field within the CAN frame format in which the error occurred, and it identifies the bit number of the erroneous bit within the frame field. The APB bus master has read-only access to this register, and all bits are cleared on reset.

15	14	13	12	11	10	9	4	3	0
Reserved	DRIVE	MON	CRC	STUFF	TXE	EBID		EFID	
0									
R									

EFID The Error Field Identifier field identifies the frame field in which the last error occurred. The encoding of the frame fields is shown in Table 22-19.

Table 22-19. Error Field Identifier

EFID3:0	Field
0000	ERROR
0001	ERROR DEL
0010	ERROR ECHO
0011	BUS IDLE
0100	ACK
0101	EOF
0110	INTERMISSION
0111	SUSPEND TRANSMISSION
1000	SOF
1001	ARBITRATION
1010	IDE
1011	EXTENDED ARBITRATION
1100	R1/R0
1101	DLC
1110	DATA
1111	CRC

EBID The Error Bit Identifier field reports the bit position of the incorrect bit within the erroneous frame field. The bit number starts with the value equal to the respective frame field length minus one at the beginning of each field and is decremented with each CAN bit. Figure 22-30 shows an example on how the EBID is calculated.

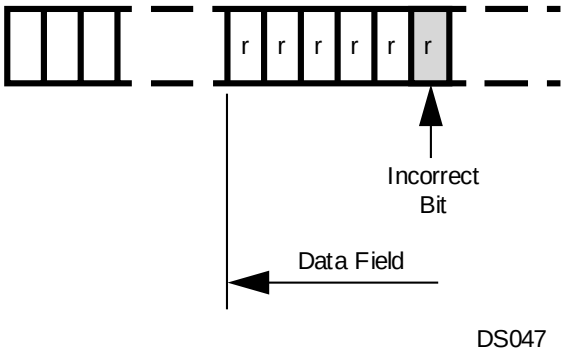


Figure 22-30. EBID Example

For example, assume the EFID field shows 1110b and the EBID field shows 111001b. This means the faulty field was the data field. To calculate the bit position of the error, the DLC of the message needs to be known. For example, for a DLC of 8 data bytes, the bit counter starts with the value: (8 × 8) – 1 = 63; so when EBID5:0 = 111001b = 57, then the bit number was 63 – 57 = 6.

TXE The Transmit Error bit indicates whether the CAN module was an active transmitter at the time the error occurred.
0 – The CAN module was a receiver at the time the error occurred.
1 – The CAN module was an active transmitter at the time the error occurred.

STUFF The Stuff Error bit indicates whether the bit stuffing rule was violated at the time the error occurred. Note that certain bit fields do not use bit stuffing and therefore this bit may be ignored for those fields.
0 – No bit stuffing error.
1 – The bit stuffing rule was violated at the time the error occurred.

CRC The CRC Error bit indicates whether the CRC is invalid. This bit should only be checked if the EFID field shows the code of the ACK field.
0 – No CRC error occurred.
1 – CRC error occurred.

MON The Monitor bit shows the bus value on the CANnRX pin as sampled by the CAN module at the time of the error.

DRIVE The Drive bit shows the output value on the CANnTX pin at the time of the error. Note that a receiver will not drive the bus except during ACK and during an active error flag.

22.10.17 CAN Timer Register n (CTMRn)

The CTMR register reports the current value of the Time Stamp Counter as described in Section 22.8.

15	0
CTMR15:0	
0	
R	

The CTMRn register is a free running 16-bit counter. It contains the number of CAN bits recognized by the CAN module since the register has been cleared. The counter starts to increment from the value 0000h after a hardware reset. If the Timer Stamp Enable bit (TSTPEN) in the CAN global configuration register (CGCRn) is set, the counter will also be cleared on a message transfer of the message buffer 0

The contents of CTMRn are captured into the Time Stamp register of the message buffer after successfully sending or receiving a frame, as described in Section 22.8.

22.11 System Start-Up and Multi-Input Wake-Up

After system start-up, all CAN-related registers are in their reset state. The CAN module can be enabled after all configuration registers are set to their desired value. The following initial settings must be made:

- Configure the CAN Timing register (CTIMn). See Section 22.2.6.
- Configure every buffer to its function as receive/transmit. See Section 22.10.1.
- Set the acceptance filtering masks. See Section 22.4.
- Enable the CAN interface. See Section 22.10.6.

Before disabling the CAN module, software must make sure that no transmission is still pending.

NOTE

Activity on the CAN bus can wake up the device from a low-power mode by selecting the CANnRX pin as an input to the Multi-Input Wake-Up module. In this case, the CAN module must not be disabled before entering the low-power mode. Disabling the CAN module also disables the CANnRX pin. As an alternative, the CANnRX pin can be connected to any other input pin of the Multi-Input Wake-Up module. This input channel must then be configured to trigger a wake-up event on a falling edge (if a dominant bit is represented by a low level). In this case, the CAN module can be disabled before entering the low-power mode. After waking up, software must enable the CAN module again. All configuration and buffer registers still contain the same data they held before the low-power mode was entered.

22.11.1 External Connection

The CAN module uses the CANnTX and CANnRX pins to connect to the physical layer of the CAN interface. They provide the functionality described in Table 22-20.

Table 22-20. External CAN Pins

Signal Name	Type	Description
CANnTX	Output	Transmit data to the CAN bus
CANnRX	Input	Receive data from the CAN bus

The logic levels are configurable by the CTX and CRX bits of the Global Configuration Register CGCRn (see Section 22.10.6).

22.11.2 Transceiver Connection

An external transceiver chip must be connected between the CAN block and the bus. It establishes a bus connection in differential mode and provides the driver and protection requirements. Figure 22-31 shows a possible ISO-High-Speed configuration.

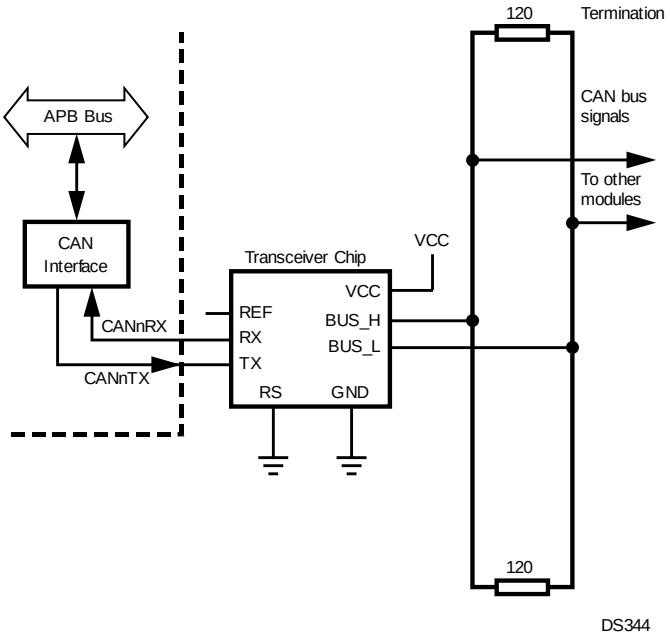


Figure 22-31. External Transceiver

22.11.3 Timing Requirements

Processing messages and updating message buffers require a certain number of clock cycles, as shown in Table 22-21. These requirements may lead to some restrictions regarding the Bit Time Logic settings and the overall CAN performance which are described below in more detail. Wait cycles need to be added to the cycle count for CPU access to the object memory as described in Section 22.9.1. The number of occurrences per frame is dependent on the number of matching identifiers.

Table 22-21. CAN Module Internal Timing

Task	Cycle Count	Occurrence/Frame
Copy hidden buffer to receive message buffer	17	0–1
Update status from TX_RTR to TX_ONCE_RTR	3	0–15

Table 22-21. CAN Module Internal Timing (continued)

Task	Cycle Count	Occurrence/Frame
Schedule a message for transmission	2	0–1

The critical path derives from receiving a remote frame, which triggers the transmission of one or more data frames. There are a minimum of four bit times in-between two consecutive frames. These bit times start at the validation point of received frame (reception of 6th EOF bit) and end at the earliest possible transmission start of the next frame, which is after the third intermission bit at 100% burst bus load.

These four bit times have to be set in perspective with the timing requirements of the CAN module.

The minimum duration of the four CAN bit times is determined by the following Bit Time Logic settings:

PSC = PSCmin = 2
TSEG1 = TSEG1min = 2
TSEG2 = TSEG2min = 1
Bit time = Sync + Time Segment 1 + Time Segment 2
= (1 + 2 + 1) tq = 4 tq
= (4 tq × PSC) clock cycles
= (4 tq × 2) clock cycles = 8 clock cycles

For these minimum BTL settings, four CAN bit times take 32 clock cycles.

The following is an example that assumes typical case:

- Minimum BTL settings
- Reception and copy of a remote frame
- Update of one buffer from TX_RTR
- Schedule of one buffer from transmit

As outlined in Table 22-21, the copy process, update, and scheduling the next transmission gives a total of 17 + 3 + 2 = 22 clock cycles. Therefore under these conditions there is no timing restriction.

The following example assumes the worst case:

- Minimum BTL settings
- Reception and copy of a remote fram
- Update of the 14 remaining buffers from TX_RTR
- Schedule of one buffer for transmit

All these actions in total require 17 + (14 × 3) + 2 = 61 clock cycles to be executed by the CAN module. This leads to the limitation of the Bit Time Logic of 61 / 4 = 15.25 clock cycles per CAN bit as a minimum, resulting in the minimum clock frequencies listed below. (The frequency depends on the desired baud rate and assumes the worst case scenario can occur in the application.)

Table 22-22 gives examples for the minimum clock frequency in order to ensure proper functionality at various CAN bus speeds.

Table 22-22. Minimum Clock Frequency Requirements

Baud Rate	Minimum Clock Frequency
1 Mbit/sec	15.25 MHz
500 kbit/sec	7.625 MHz
250 kbit/sec	3.81 MHz

22.11.4 Bit Time Logic Calculation Examples

The calculation of the CAN bus clocks using CKI = 16 MHz is shown in the following examples. The desired baud rate for both examples is 1 Mbit/s.

Example 1
PSC = PSC[5:0] + 2 = 0 + 2 = 2
TSEG1 = TSEG1[3:0] + 1 = 3 + 1 = 4
TSEG2 = TSEG2[2:0] + 1 = 2 + 1 = 3
SJW = TSEG2 = 3

- Sample point positioned at 62.5% of bit time
- Bit time = 125 ns × (1 + 4 + 3 ± 3) = (1 ± 0.375) µs
- Bus Clock = 16 MHz / (2 × (1 + 4 + 3)) = 1 Mbit/s (nominal)

Example 2
PSC = PSC[5:0] + 1 = 2 + 2 = 4
TSEG1 = TSEG1[3:0] + 1 = 1 + 1 = 2
TSEG2 = TSEG2[2:0] + 1 = 0 + 1 = 1
SJW = TSEG2 = 1

- Sample point positioned at 75% of bit time
- Bit time = 250 ns × (1 + 2 + 1 ± 1) = (1 ± 0.25) µs
- Bus Clock = 16 MHz / (2 × (1 + 4 + 3)) = 1Mbit/s (nominal)

22.11.5 Acceptance Filter Considerations

The CAN module provides two acceptance filter masks GMSKn and BMSKn, as described in Section 22.4, Section 22.10.8, and Section 22.10.9. These masks allow filtering of up to 32 bits of the message object, which includes the standard identifier, the extended identifier, and the frame control bits RTR, SRR, and IDE.

22.11.6 Remote Frames

Remote frames can be automatically processed by the CAN module. However, to fully enable this feature, the RTR/ XRTR bits (for both standard and extended frames) within the BMSKn and/or GMSKn register need to be set to “don’t care”. This is because a remote frame with the RTR bit set should trigger the transmission of a data frame with the RTR bit clear and therefore the ID bits of the received message need to pass through the acceptance filter. The same applies to transmitting remote frames and switching to receive the corresponding data frames.

22.12 Usage Notes

Under certain conditions, the CAN module receives a frame sent by itself, even though the loopback feature is disabled. Two conditions must be true to cause this malfunction:

- A transmit buffer and at least one receive buffer are configured with the same identifier. Assume this identifier is called ID_RX_TX. With regard to the receive buffer, this means that the buffer identifier and the corresponding fi ter masks are set up in a way that the buffer is able to receive frames with the identifier ID_RX_TX.
- The following sequence of events occurs:
  1. A message with the identifier ID_RX_TX from another CAN node is received into the receive buffer.
  2. A message with the identifier ID_RX_TX is sent by the CAN module immediately after the reception took place.

When these conditions occur, the frame sent by the CAN module will be copied into the next receive buffer available for the identifier ID_RX_TX.

If a frame with an identifier different to ID_RX_TX is sent or received in between events 1 and 2, the problem does not occur.

23 Analog-to-Digital Converter

The ADC provides the following features:

- 10-input analog multiplexer
- 10 single-ended channels or 5 differential channels
- External filtering capability
- 12-bit resolution with 10-bit accuracy
- Sign bit
- 10-microsecond conversion time
- Internal or external start trigger

- Programmable start delay after start trigger
- Poll or interrupt on conversion complete

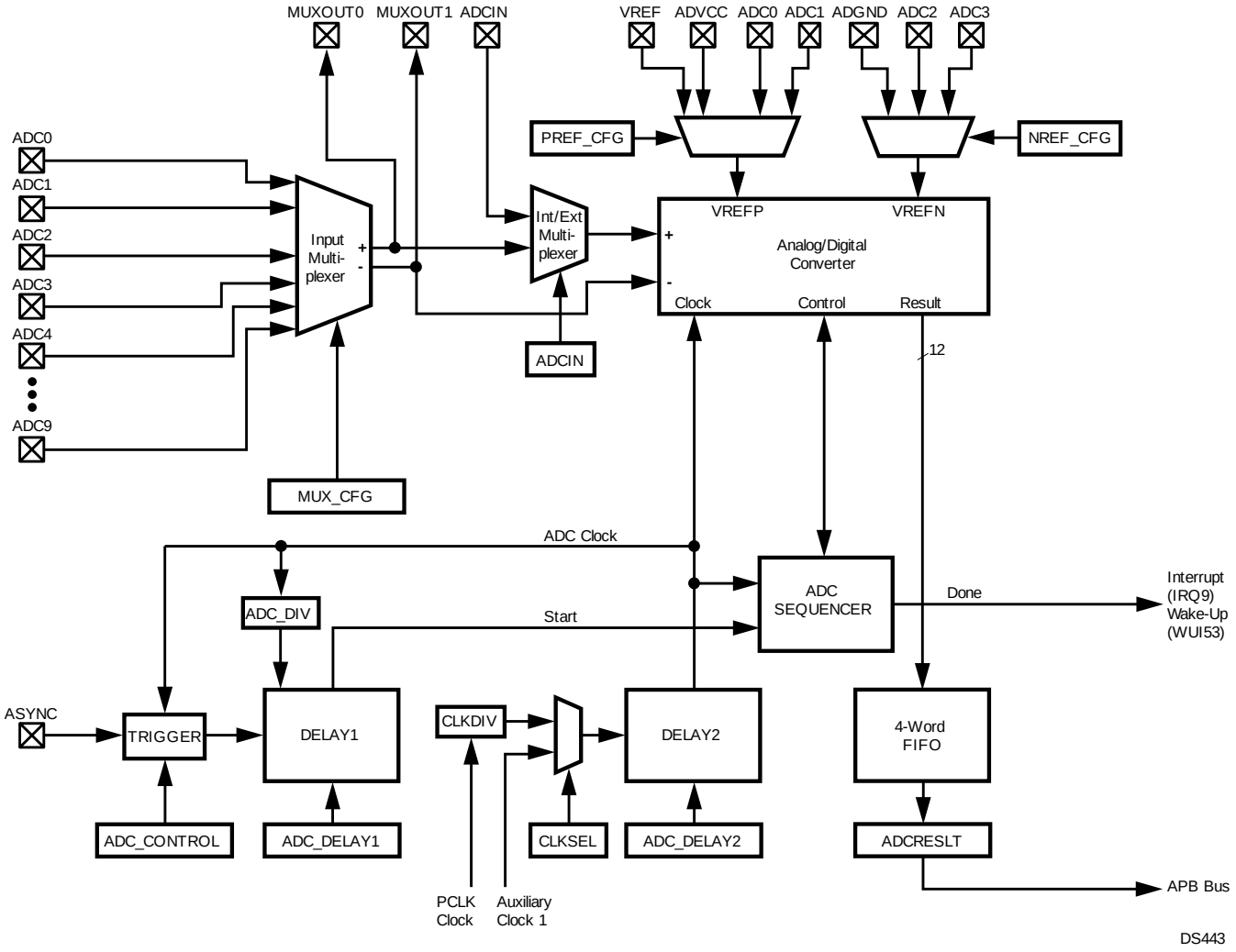


Figure 23-1. Analog-to-Digital Converter Block Diagram

23.1 Funcational Description

The ADC module consists of an analog/digital converter and associated state machine, together with analog multiplexers to set up signal paths for sampling and voltage references, logic to control triggering of the converter, and a bus interface.

23.1.1 Data Path

Up to 10 pins on the FBGA-224 package may be configured as 10 single-ended analog inputs or 5 differential pairs. (Two pins are available as single-ended inputs or one differential pair on the FBGA-144 package). Analog/digital data passes through four main blocks in the ADC module between the input pins and the APB bus:

- **Input Multiplexer**—an analog multiplexer that selects among the input channels.
- **Internal/External Multiplexer**—an analog multiplexer that selects between the output of the Input Multiplexer and the ADCIN external analog input.
- **Analog/Digital Converter**—receives the output of the Internal/External Multiplexer and performs the analog to digital conversion.

- **ADCRESLT Register**—makes conversion results from the ADC available to the on-chip bus. The ADCRESLT register includes the software-visible end of a 4-word FIFO used to queue conversion results.

The configuration of the analog signal paths is controlled by fields in the ADCGCR register. The Input Multiplexer is controlled by the MUX_CFG field. The Internal/External Multiplexer is controlled by the ADCIN bit. The analog multiplexers for selecting the voltage references used by the ADC are controlled by the PREF_CFG and NREF_CFG fields.

The output of the Input Multiplexer is available externally as the MUXOUT0 and MUXOUT1 signals. In single-ended mode, only MUXOUT0 is used. In differential mode, MUXOUT0 is the positive side and MUXOUT1 is the negative side. The MUXOUT0 and MUXOUT1 outputs and the ADCIN external analog input are provided so that external signal conditioning circuits (such as filters) may be applied to the analog signals before conversion. The MUXOUT0, MUXOUT1, and ADCIN signals are alternate functions of the ADC inputs, so the number of available ADC inputs is reduced when these signals are used.

23.1.2 Operation

The TRIGGER block may be configured to initiate a conversion from either of these sources:

- **External ASYNC Input**—an edge on the ASYNC input triggers a conversion. This input may be configured to be sensitive to rising or falling edges, as controlled by the POL bit in the ADCCNTRL register.
- **ADCSTART Register**—writing any value to the ADCSTART register triggers a conversion.

The TRIGGER block incorporates a glitch filter to suppress transient spikes on the ASYNC input. The TRIGGER block will recognize ASYNC pulse widths of 10 ns or greater. Once a trigger event has been recognized, no further triggering is recognized until the conversion is completed.

When the ASYNC input is selected as the trigger source, it may be configured for automatic or non-automatic mode, as controlled by the AUTO bit in the ADCCNTRL register:

- **Automatic Mode**—a conversion is triggered by any qualified edge on the ASYNC input (unless a conversion is already in progress).
- **Non-Automatic Mode**—before a conversion may be triggered from the ASYNC input, software must “prime” the TRIGGER block by writing the ADCSTART register. Once the TRIGGER block is primed, a conversion is triggered by any qualified edge on the ASYNC input. After the conversion is completed, no additional trigger events will be recognized until software once again primes the TRIGGER block by writing the ADCSTART register.

Once a trigger event is recognized, the DELAY1 block waits for a programmable delay specified in the ADC_DELAY1 field of the ADCSCDLY register. Then, it asserts the Start signal to the ADC SEQUENCER block.

When the Start signal is received, the ADC SEQUENCER block initiates the conversion in the ADC. After the conversion is complete, the result is loaded into the FIFO, and the Done signal is asserted.

The ADCRESLT register includes the software-visible end of a 4-word FIFO, which allows up to 4 conversion results to be queued for reading. Reading the ADCRESLT register unloads the FIFO. If the FIFO overflows, a bit is set in the ADCRESLT register, and the last conversion result is lost.

The Done signal is visible to software as the ADC_DONE bit in the ADCRESLT register. The Done signal is also an input to the interrupt controller (IRQ9). The interrupt will be asserted whenever the FIFO is not empty (but will deassert for one PCLK Clock period after the ADCRESLT register is read). Total conversion time is 10 microseconds.

The Done signal is also an input to the Multi-Input Wake-Up unit (WUI53). The MIWU input is asserted whenever the FIFO is not empty (but will deassert for one clock period after the ADCRESLT register is read). The wake-up output is provided so that the ADC module can bring the system out of a low-power mode when a conversion operation is completed. It asserts earlier than the interrupt output.

23.1.3 ADC Clock Generation

The DELAY2 block generates ADC Clock, which is the clock used internally by the ADC module. ADC Clock is derived from either:

- **PCLK Clock**—a programmable divider is available to generate the 12 MHz clock required by the ADC from the PCLK Clock.
- **Auxiliary Clock 1**—may be used to perform conversions when the PCLK Clock is slowed down or suspended in low-power modes.

The DELAY2 block receives the clock source selected by the CLKSEL bit of the ADCACR register and adds a number of asynchronous incremental delay units specified in the ADC_DELAY2 field of the ADCSCDLY register. This delayed clock (ADC Clock) then drives the TRIGGER, ADC, and ADC SEQUENCER blocks. ADC Clock also drives the ADC_DIV clock divider, which generates the clock which drives the DELAY1 block.

Because the ADCRESLT FIFO is driven by PCLK Clock (not ADC Clock), a conversion result will not propagate to the output of the FIFO when PCLK Clock is suspended.

23.1.4 ADC Voltage References

The ADC block has positive and negative voltage reference inputs, VREFP and VREFN. In single-ended mode, only VREFP is used. An analog multiplexer allows selecting an external VREF pin, the analog supply voltage ADVCC, or the analog inputs ADC0 or ADC1 as the positive voltage reference, as controlled by the PREF_CFG field of the ADCGCR register. Another analog multiplexer allows selecting the analog ground ADGND or the analog inputs ADC2 or ADC3 as the negative voltage reference, as controlled by the NREF_CFG field of the ADCGCR register.

23.2 Operation in Low-Power Modes

To reduce the level of switching noise in the environment of the ADC, it is possible to operate the CP3SP33 in low-power modes, in which the PCLK Clock is slowed or switched off. Under these conditions, Auxiliary Clock 1 can be selected as the clock source for the ADC module, however conversion results cannot be read by the system while the PCLK Clock is suspended. To operate in a low-power mode:

1. ADC is configured and a conversion is primed or triggered.
2. A low-power mode is entered.
3. ADC conversion completes and a wake-up signal is asserted to the MIWU unit.
4. Device wakes up and processes the conversion result.

To conserve power, the ADC should be disabled before entering a low-power mode if its function is not required.

23.3 Freeze Mode

When Freeze mode is entered, the ADC will exhibit the following specific behavior:

- The automatic clear-on-read function of the result register (ADCRESLT) is disabled.
- The FIFO is updated as usual, and an interrupt for a completed conversion can be asserted.

23.4 ADC Register Set

Table 23-1 lists the ADC registers.

Table 23-1. ADC Registers

Name	Address	Description
ADCGCR	FF AC00h	ADC Global Configuration Register
ADCACR	FF AC04h	ADC Auxiliary Configuration Register
ADCCNTRL	FF AC08h	ADC Conversion Control Register
ADCSTART	FF AC0Ch	ADC Start Conversion Register
ADCSCDLY	FF AC10h	ADC Start Conversion Delay Register
ADCRESLT	FF AC14h	ADC Result Register

23.4.1 ADC Global Configuration Register (ADCGCR)

The ADCGCR register is a 16-bit, read/write register that controls the basic operation of the interface. The APB bus master has read/write access to the ADCGCR register. After reset this register is cleared.

15	14	13 12	11 10	9 7	6 3	2	1	0
MUXOUTEN	INTEN	NREF_CFG	PREF_CFG	Reserved	MUX_CFG	DIFF	ADCIN	CLKEN

CLKEN The Clock Enable bit controls whether the ADC module is running. When this bit is clear, all ADC clocks are disabled, the ADC analog circuits are in a low-power state, and ADC registers (other than the ADCGCR and AGCACR registers) are not writeable. Clearing this bit reinitializes the ADC state machine and cancels any pending trigger event. Setting this bit enables the ADC clocks and powers up the ADC analog circuits. The converter is operational within 0.25 µs of being enabled.
0 – ADC disabled.
1 – ADC enabled.

ADCIN The ADCIN bit selects the source of the ADC input. When the bit is clear, the source is the 10-channel Input Multiplexer. When the bit is set, the source is the ADCIN pin.
0 – ADC input is from 10-channel multiplexer.
1 – ADC input is from ADCIN pin.

DIFF The Differential Operation Mode bit and the MUX_CFG field configure the analog circuits of the ADC module. When this bit is clear, the ADC module operates in single-ended mode. When this bit is set, the ADC operates in differential mode.
0 – Single-ended mode.
1 – Differential mode.

MUX_CFG The Multiplexer Configuration field and the DIFF bit configure the analog circuits of the ADC module, as shown below.

MUX_CFG	Channel (DIFF = 0)	Channel (DIFF = 1)	
		+	-
0000	0	0	1
0001	1	1	0
0010	2	2	3
0011	3	3	2
0100	4	4	5
0101	5	5	4
0110	6	6	7
0111	7	7	6

MUX_CFG	Channel (DIFF = 0)	Channel (DIFF = 1)	
		+	-
1000	8	8	9
1001	9	9	8
1010 to 1111	Reserved		

PREF_CFG The Positive Voltage Reference Configuration field specifies the source of the ADC positive voltage reference, as shown below:

PREF_CFG	PREF Source
00	Internal (ADVCC)
01	VREF
10	ADC0
11	ADC1

NREF_CFG The Negative Voltage Reference Configuration field specifies the source of the ADC negative voltage reference, as shown below:

NREF_CFG	NREF source
00	Internal (ADGND)
01	Reserved
10	ADC2
11	ADC3

MUXOUTEN The MUXOUT Enable bit controls whether the output of the Input Multiplexer is available externally. In single-ended mode, the MUXOUT0 pin is active and the MUXOUT1 pin is disabled (TRI-STATE). In differential mode, both MUXOUT0 and MUXOUT1 are active.
0 – MUXOUT0 and MUXOUT1 disabled.
1 – MUXOUT0 and MUXOUT1 enabled.

INTEN The Interrupt Enable bit controls whether the ADC interrupt (IRQ9) is enabled. When enabled, the interrupt request is asserted when valid data is available in the ADCRESLT register. This bit has no effect on the wake-up signal to the MIWU unit (WUI53).
0 – IRQ9 disabled.
1 – IRQ9 enabled.

23.4.2 ADC Auxiliary Configuration Register (ADCACR)

The ADCACR register is a 16-bit, read/write register used to control the clock configuration and report the status of the ADC module. The APB bus master has read/write access to the ADCACR register. After reset, this register is clear.

15	14	13	12 … 3	2 … 1	0
CNVT	TRG	PRM	Reserved	CLKDIV	CLKSEL

CLKSET The Clock Select bit selects the clock source used by the DELAY2 block to generate the ADC clock.
0 – ADC clock derived from PCLK Clock.
1 – ADC clock derived from Auxiliary Clock 1.

CLKDIV The Clock Divisor field specifies the divisor applied to PCLK Clock to generate the 12 MHz clock required by the ADC module. Only the PCLK Clock is affected by this divisor. The divisor is not used when Auxiliary Clock 1 is selected as the clock source.

CLKDIV	Clock Divisor
00	1
01	2
10	4
11	Reserved

PRM The ADC Primed bit is a read-only bit that indicates the ADC has been primed to perform a conversion by writing to the ADCSTART register. The bit is cleared after the conversion is completed.
0 – ADC has not been primed.
1 – ADC has been primed.

TRG The ADC Triggered bit is a read-only bit that indicates the ADC has been triggered. The bit is set during any pre-conversion delay. The bit is cleared after the conversion is completed. Once triggered, no new trigger events will be recognized until after the conversion has completed, as indicated by the ADC_DONE bit in the ADCRESLT register.
0 – ADC has not been triggered.
1 – ADC has been triggered.

CNVT The ADC Conversion bit is a read-only bit that indicates the ADC has been primed to perform a conversion, a valid internal or external trigger event has occurred, any preconversion delay has expired, and the ADC conversion is in progress. The bit is cleared after the conversion is completed.
0 – ADC is not performing a conversion.
1 – ADC conversion is in progress.

23.4.3 ADC Conversion Control Register (ADCCNTRL)

The ADCCNTRL register is a 16-bit, read/write register that specifies the trigger conditions for an ADC conversion. After reset, the register is cleared.

15 … 3	2	1	0
Reserved	AUTO	EXT	POL

POL The ASYNC Polarity bit specifies the polarity of edges which trigger ADC conversions.
0 – ASYNC input is sensitive to rising edges.
1 – ASYNC input is sensitive to falling edges.

EXT The External Trigger bit selects whether conversions are triggered by writing the ADCSTART register or activity on the ASYNC input.
0 – ADC conversions triggered by writing to the ADCSTART register.
1 – ADC conversions triggered by qualified edges on ASYNC input.

AUTO The Automatic bit controls whether automatic mode is enabled, in which any qualified edge on the ASYNC input is recognized as a trigger event. When automatic mode is disabled, the ADC module must be “primed” before a qualified edge on the ASYNC input can trigger a conversion. To prime the ADC module, software must write the ADCSTART register with any value before an edge on the ASYNC input is recognized as a trigger event. After the conversion is completed, the ASYNC input will be ignored until software again writes the ADCSTART register. The AUTO bit is ignored when the EXT bit is 0.
0 – Automatic mode disabled.
1 – Automatic mode enabled.

23.4.4 ADC Start Conversion Register (ADCSTART)

The ADCSTART register is a write-only register used by software to initiate an ADC conversion. Writing any value to this register will cause the ADC to initiate a conversion or prime the ADC to initiate a conversion, as controlled by the ADCCNTRL register.

23.4.5 ADC Start Conversion Delay Register (ADCSCDLY)

The ADCSCDLY register is a 16-bit, read/write register that controls critical timing parameters for the operation of the ADC module. After reset, the register is cleared.

15 14	13 5	4 0
ADC_DIV	ADC_DELAY1	ADC_DELAY2

ADC_DELAY2 The ADC Delay 2 field specifies the delay between the ADC module clock source (either PCLK Clock after a programmable divider or Auxiliary Clock 1) and the ADC clock. The range of effective values for this field is 0 to 20. Values above 20 produce the same delay as 20, which is about 42 ns.

ADC_DELAY1 The ADC Delay 1 field specifies the number of clock periods by which the trigger event will be delayed before initiating a conversion. The timebase for this delay is the ADC clock (12 MHz) divided by the ADC_DIV divisor. The ADC_DELAY1 field has 9 bits, which corresponds to a maximum delay of 511 clock periods.

ADC_DIV The ADC Clock Divisor field specifies the divisor applied to the ADC clock (12 MHz) to generate the clock used to drive the DELAY1 block. The field is biased by 1, so the divisor selected by the ADC_DIV field may be 1, 2, 3, or 4. With a module clock of 12 MHz, the maximum delay which can be provided by ADC_DIV and ADC_DELAY settings is:

$$\left(\frac{1}{12\ \text{MHz}}\right) \times 4 \times 511 = 170\ \text{us}$$

23.4.6 ADC Result Register (ADCRESLT)

The ADCRESLT register is a 16-bit, read-only register that includes the software-visible end of a 4-word FIFO. Conversion results are loaded into the FIFO from the ADC and unloaded when software reads the ADCRESLT register. The ADCRESLT register is read-only. The defined fields in this register are cleared when the register is read. After reset, this register is clear.

15	14	13	12	11 0
ADC_DONE	ADC_OFLW	Reserved	SIGN	ADC_RESULT

ADC_RESULT The ADC Result field holds a 12-bit value for the conversion result. If the ADC_DONE bit is clear, there is no valid result in this field, and the field will have a value of 0. The ADC_RESULT field and the SIGN bit together form the software-visible end of the ADC FIFO.

SIGN The Sign bit indicates whether the (minus) – input has a voltage greater than the (plus) + input (differential mode only). For example if ADCGCR.MUX_CFG is 000b, ADC0 is the + input and ADC1 is the – input. If the voltage on ADC0 is greater than the voltage on ADC1, the SIGN bit will be 0; if the voltage on ADC0 is less than the voltage on ADC1, the SIGN bit will be 1. In single-ended mode, this bit always reads as 0.
0 – In differential mode, + input has a voltage greater than the – input. In single-ended mode, this bit is always 0.
1 – In differential mode, – input has a voltage greater than the + input.

ADC_OFLW The ADC FIFO Overflow bit indicates whether the 4-word FIFO behind the ADCRESLT register has overflowed. When this occurs, the most recent conversion result is lost. This bit is cleared when the ADCRESLT register is read.
0 – FIFO overflow has not occurred.
1 – FIFO overflow has occurred.

ADC_DONE	The ADC Done bit indicates when an ADC conversion has completed. When this bit is set, the data in the ADC_RESULT field is valid. When this bit is clear, there is no valid data in the ADC_RESULT field. The Done bit is cleared when the ADCRESLT register is read, but if there are queued conversion results in the FIFO, the Done bit will become set again after one PCLK Clock period. 0 – No ADC conversion has completed since the ADCRESLT register was last read. 1 – An ADC conversion has completed since the ADCRESLT register was last read.

24 Advanced Audio Interface

The Advanced Audio Interface (AAI) provides a serial synchronous, full duplex interface to codecs and similar serial devices. The transmit and receive paths may operate asynchronously with respect to each other. Each path uses a 3-wire interface consisting of a bit clock, a frame synchronization signal, and a data signal.

The CPU interface can be either interrupt-driven or DMA. If the interface is configured for interrupt-driven I/O, data is buffered in the receive and transmit FIFOs. If the interface is configured for DMA, the data is buffered in registers.

The AAI is functionally similar to a Motorola™ Synchronous Serial Interface (SSI). Compared to a standard SSI implementation, the AAI interface does not support the so called “On-demand Mode”. It also does not allow gating of the shift clocks, so the receive and transmit shift clocks are always active while the AAI is enabled. The AAI also does not support 12-bit and 24-bit data word length or more than 4 slots (words) per frame. The reduction of supported modes is acceptable, because the main purpose of the AAI is to connect to audio codecs, rather than to other processors (DSPs).

The implementation of a FIFO as a 16-word receive and transmit buffer is an additional feature, which simplifies communication and reduces interrupt load. Independent DMA is provided for each of the four supported audio channels (slots). The AAI also provides special features and operating modes to simplify gain control in an external codec and to connect to an ISDN controller through an IOM-2 compatible interface.

24.1 Audio Interface Signals

24.1.1 Serial Transmit Data (STD)

The STD pin is used to transmit data from the serial transmit shift register (ATSR). The STD pin is an output when data is being transmitted and is in high-impedance mode when no data is being transmitted. The data on the STD pin changes on the positive edge of the transmit shift clock (SCK). The STD pin goes into high-impedance mode on the negative edge of SCK of the last bit of the data word to be transmitted, assuming no other data word follows immediately. If another data word follows immediately, the STD pin remains active rather than going to the high-impedance mode.

24.1.2 Serial Transmit Clock (SCK)

The SCK pin is a bidirectional signal that provides the serial shift clock. In asynchronous mode, this clock is used only by the transmitter to shift out data on the positive edge. The serial shift clock may be generated internally or it may be provided by an external clock source. In synchronous mode, the SCK pin is used by both the transmitter and the receiver. Data is shifted out from the STD pin on the positive edge, and data is sampled on the SRD pin on the negative edge.

24.1.3 Serial Transmit Frame Sync (SFS)

The SFS pin is a bidirectional signal which provides frame synchronization. In asynchronous mode, this signal is used as frame sync only by the transmitter. In synchronous mode, this signal is used as frame sync by both the transmitter and receiver. The frame sync signal may be generated internally, or it may be provided by an external source.

24.1.4 Serial Receive Data (SRD)

The SRD pin is used as an input when data is shifted into the Audio Receive Shift Register (ARSR). In asynchronous mode, data on the SRD pin is sampled on the negative edge of the serial receive shift clock (SRCLK). In synchronous mode, data on the SRD pin is sampled on the negative edge of the serial shift clock (SCK). The data is shifted into ARSR with the most significant bit (MSB) first.

24.1.5 Serial Receive Clock (SRCLK)

The SRCLK pin is a bidirectional signal that provides the receive serial shift clock in asynchronous mode. In this mode, data is sampled on the negative edge of SRCLK. The SRCLK signal may be generated internally or it may be provided by an external clock source. In synchronous mode, the SCK pin is used as shift clock for both the receiver and transmitter, so the SRCLK pin is available for use as a general-purpose port pin or an auxiliary frame sync signal to access multiple slave devices (e.g. codecs) within a network (see Network mode).

24.1.6 Serial Receive Frame Sync (SRFS)

The SRFS pin is a bidirectional signal that provides frame synchronization for the receiver in asynchronous mode. The frame sync signal may be generated internally, or it may be provided by an external source. In synchronous mode, the SFS signal is used as the frame sync signal for both the transmitter and receiver, so the SRFS pin is available for use as a general-purpose port pin or an auxiliary frame sync signal to access multiple slave devices (e.g. codecs) within a network (see Network mode).

24.2 Audio Interface Modes

There are two clocking modes: asynchronous mode and synchronous mode. These modes differ in the source and timing of the clock signals used to transfer data. When the AAI is generating the bit shift clock and frame sync signals internally, synchronous mode must be used.

There are two framing modes: normal mode and network mode. In normal mode, one word is transferred per frame. In network mode, up to four words are transferred per frame. A word may be 8 or 16 bits. The part of the frame which carries a word is called a slot. Network mode supports multiple external devices sharing the interface, in which each device is assigned its own slot. Separate frame sync signals are provided, so that each device is triggered to send or receive its data during its assigned slot.

24.2.1 Asynchronous Mode

In asynchronous mode, the receive and transmit paths of the audio interface operate independently, with each path using its own bit clock and frame sync signal. The frame sync signals must be supplied externally.

24.2.2 Synchronous Mode

In synchronous mode, the receive and transmit paths of the audio interface use the same shift clock and frame sync signal. The bit shift clock and frame sync signal for both paths are derived from the same set of clock prescalers.

24.2.3 Normal Mode

In normal mode, each rising edge on the frame sync signal marks the beginning of a new frame and also the beginning of a new slot. A slot does not necessarily occupy the entire frame. (A frame can be longer than the data word transmitted after the frame sync pulse.) Typically, a codec starts transmitting a fixed length data word (e.g. 8-bit log PCM data) with the frame sync signal, then the codec's transmit pin returns to the high-impedance state for the remainder of the frame.

The Audio Receive Shift Register (ARSR) deserializes data received on the SRD pin (serial receiver data). Only the data sampled after the frame sync signal are treated as valid. If the interface is interrupt-driven, valid data bits are transferred from the ARSR to the receive FIFO. If the interface is configured for DMA, the data is transferred to the receive DMA register 0 (ARDR0).

The serial transmit data (STD) pin is only an active output while data is shifted out. After the defined number of data bits have been shifted out, the STD pin returns to the high-impedance state.

For operation in normal mode, the Slot Count Select field in the Global Configuration register (AGCR) must be loaded with 00b (one slot per frame). In addition, the Slot Assignment field for receive and transmit must select slot 0.

If the interface is configured for DMA, the DMA slot assignment bits must select slot 0. In this case, the audio data is transferred to or from the receive or transmit DMA register 0 (ARDR0/ATDR0).

Figure 24-1 shows the frame timing while operating in normal mode with a long frame sync interval.

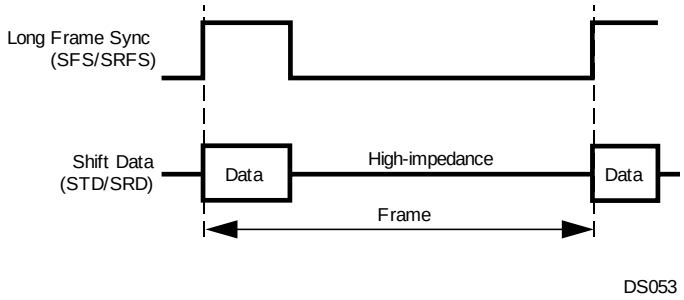


Figure 24-1. Normal Mode Frame

24.2.3.1 IRQ Support

If the receiver interface is configured for interrupt-driven I/O (RXDSA0 = 0), all received data are loaded into the receive FIFO. An IRQ is asserted as soon as the number of data bytes or words in the receive FIFO is greater than a programmable warning limit.

If the transmitter interface is configured for interrupt-driven I/O (TXDSA0 = 0), all data to be transmitted is read from the transmit FIFO. An IRQ is asserted as soon as the number data bytes or words available in the transmit FIFO is equal or less than a programmable warning limit.

24.2.3.2 DMA Support

If the receiver interface is configured for DMA (RXDSA0 = 1), received data is transferred from the ARSR into the DMA receive buffer 0 (ARDR0). A DMA request is asserted when the ARDR0 register is full. If the transmitter interface is configured for DMA (TXDSA0 = 1), data to be transmitted are read from the DMA transmit buffer 0 (ATDR0). A DMA request is asserted to the DMA controller when the ATDR0 register is empty.

Figure 24-2 shows the data flow for IRQ and DMA mode in normal Mode.

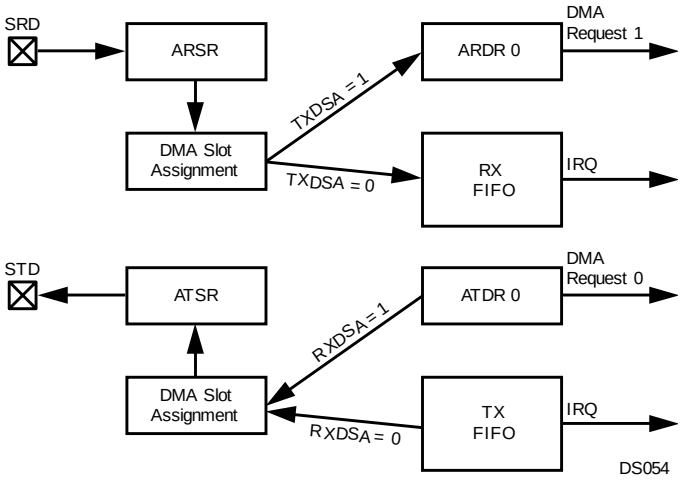


Figure 24-2. IRQ/DMA Support in Normal Mode

24.2.3.3 Network Mode

In network mode, each frame is composed of multiple slots. Each slot may transfer 8 or 16 bits. All of the slots in a frame must have the same length. In network mode, the sync signal marks the beginning of a new frame.

More than two devices can communicate within a network using the same clock and data lines. The devices connected to the same bus use a time-multiplexed approach to share access to the bus. Each device has certain slots assigned to it, in which only that device is allowed to transfer data. One master device provides the bit clock and the frame sync signal(s). On all other (slave) devices, the bit clock and frame sync pins are inputs.

Up to four slots can be assigned to the AAI, as it supports up to four slots per frame. Any other slots within the frame are reserved for other devices.

The transmitter only drives data on the STD pin during slots which have been assigned to the AAI. During all other slots, the STD output is in high-impedance mode, and data can be driven by other devices. The assignment of slots to the transmitter is specified by the Transmit Slot Assignment bits (TXSA) in the ATCR register. It can also be specified whether the data to be transmitted is transferred from the transmit FIFO or the corresponding DMA transmit register. There is one DMA transmit register (ATDRn) for each of the maximum four data slots. Each slot can be configured independently.

On the receiver side, only the valid data bits which were received during the slots assigned to this interface are copied into the receive FIFO or DMA registers. The assignment of slots to the receiver is specified by the Receive Slot Assignment bits (RXSA) in the ATCR register. It can also be specified whether the received data is copied into the receive FIFO or into the corresponding DMA receive register. There is one DMA receive register (ARDRn) for each of the maximum four data slots. Each slot may be configured individually.

Figure 24-3 shows the frame timing while operating in network mode with four slots per frame, slot 1 assigned to the interface, and a long frame sync interval.

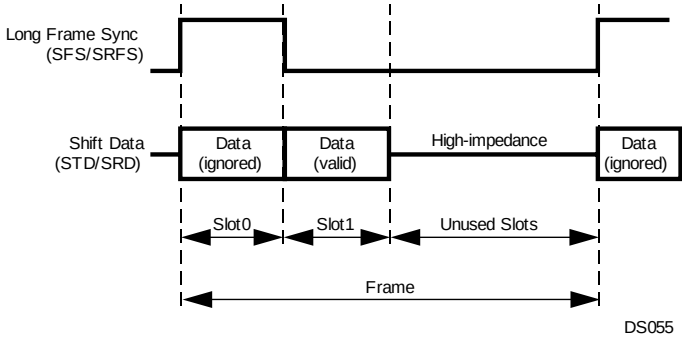


Figure 24-3. Network Mode Frame

24.2.3.4 Interrupt Support

If DMA is not enabled for a receive slot n (RXDSAn = 0), all data received in this slot is loaded into the receive FIFO. The interrupt request is asserted as soon as the number of data bytes or words in the receive FIFO is greater than a programmable warning limit.

If DMA is not enabled for a transmit slot n (TXDSAn = 0), all data to be transmitted in this slot are read from the transmit FIFO. The interrupt request is asserted as soon as the number data bytes or words available in the transmit FIFO is equal or less than a programmable warning limit.

Because the interrupt request can be handled by either the CPU or DSP interrupt controllers, the interrupt request is passed through the Audio Subsystem Controller (ASC), which is programmed to route it to the interrupt controller. To use the codec interrupt, it must be enabled at three levels:

- *AAI module*—the interrupt must be enabled in the AISCR register.
- *ASC module*—the AAI interrupt is assigned to ASC interrupt channel 7, so bit 7 in the ASCINTSEL register must be clear to select CPU interrupt controller or set to select the DSP interrupt controller
- *Interrupt controller*—the interrupt request must be enabled in the interrupt controller. For the CPU interrupt controller, this is interrupt request IRQ27

24.2.3.5 DMA Support

If DMA support is enabled for a receive slot n (RXDSA0 = 1), all data received in this slot is only transferred from the ARSR into the corresponding DMA receive register (ARDRn). A DMA request is asserted when the ARDRn register is full.

If DMA is enabled for a transmit slot n (TXDSAn = 1), all data to be transmitted in slot n are read from the corresponding DMA transmit register (ATDRn). A DMA request is asserted to the DMA controller when the ATDRn register is empty

Figure 24-4 illustrates the data flow for IRQ and DMA support in network mode, using four slots per frame and DMA support enabled for slots 0 and 1 in receive and transmit direction.

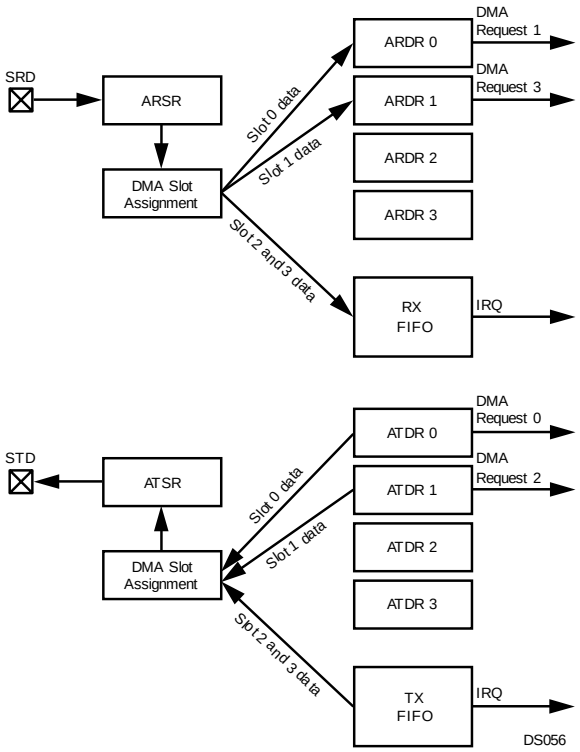


Figure 24-4. IRQ/DMA Support in Network Mode

If the interface operates in synchronous mode, the receiver uses the transmit bit clock (SCK) and transmit frame sync signal (SFS). This allows the pins used for the receive bit clock (SRCLK) and receive frame sync (SRFS) to be used as additional frame sync signals in network mode. The extra frame sync signals are useful when the audio interface communicates to more than one codec, because codecs typically start transmission immediately after the frame sync pulse. The SRCLK pin is driven with a frame sync pulse at the beginning of the second slot (slot 1), and the SRFS pin is driven with a frame sync pulse at the beginning of slot 2. Figure 24-5 shows a frame timing diagram for this configuration, using the additional frame sync signals on SRCLK and SRFS to address up to three devices.

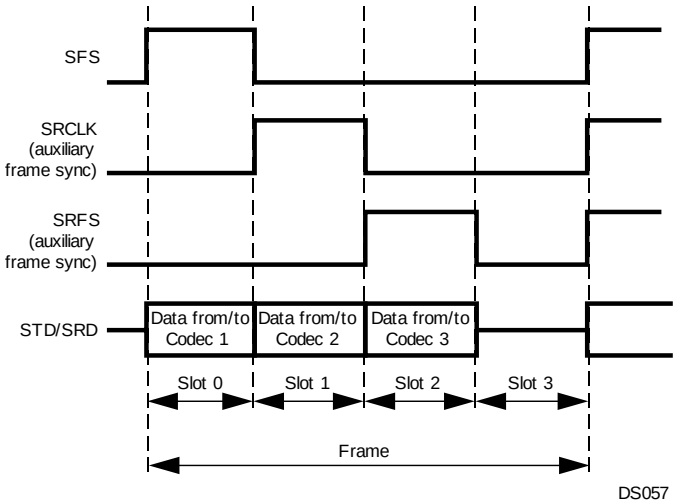


Figure 24-5. Accessing Three Devices in Network Mode

Because four of the six AAI DMA requests can be handled by either the CPU or DSP DMA controllers, these DMA requests are passed through the Audio Subsystem Controller (ASC), which is programmed to route them to the DMA controller. To use these AAI DMA requests, they must be enabled at three levels:

- *AAI module*—the DMA requests must be enabled in the ADMACR register.
- *ASC module*—the AAI DMA requests are assigned to ASC DMA channels 6 (ARDR0), 7 (ATDR0), 8 (ARDR1), and 9 (ATDR1). These channels are controlled by bits 6, 7, 8, and 9 in the ASCDMASEL0 register, respectively. The bits must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be enabled in the ASCDDMASELn registers.
- *DMA controller*—the DMA request must be enabled in the DMA controller. For the CPU DMA controller, these are DMA requests 6 (ARDR0), 7 (ATDR0), 8 (ARDR1), and 9 (ATDR1).

24.3 Bit Clock Generation

An 8-bit prescaler is provided to divide the audio interface input clock down to the required bit clock rate. Software can choose between two input clock sources, Auxiliary Clock 1 and Auxiliary Clock 8.

The input clock is divided by the value of the prescaler BCPRS7:0 + 1 to generate the bit clock.

The bit clock rate f_{bit} can be calculated by the following equation:

$f_{bit} = n \times f_{Sample} \times$ Data Length
n = Number of Slots per Frame
f_{Sample} = Sample Frequency in Hz
Data Length = Length of data word in multiples of 8 bits

The ideal required prescaler value P_{ideal} can be calculated as follows:

$P_{ideal} = f_{Audio}$ In / f_{bit}
$f_{Audio\ In}$ = Source Clock Frequency in Hz

The real prescaler must be set to an integer value, which should be as close as possible to the ideal prescaler value, to minimize the bit clock error, f_{bit_error}.

f_{bit_error} [%] = ($f_{bit} - f_{Audio\ In}/P_{real}$) / $f_{bit} \times 100$

Example:

The audio interface is used to transfer 13-bit linear PCM data for one audio channel at a sample rate of 8k samples per second. The input clock of the audio interface is 12MHz. Furthermore, the codec requires a minimum bit clock of 256 kHz to operate properly. Therefore, the number of slots per frame must be set to 2 (network mode) although actually only one slot (slot 0) is used. The codec and the audio interface will put their data transmit pins in TRI-STATE mode after the PCM data word has been transferred. The required bit clock rate f_{bit} can be calculated by the following equation:

$f_{bit} = n \times f_{Sample} \times \text{Data Length} = 2 \times 8 \text{ kHz} \times 16 = 256 \text{ kHz}$

The ideal required prescaler value $_{Pideal}$ can be calculated as follows:

$P_{ideal} = f_{Audio\ In} / f_{bit} = 12 \text{ MHz} / 256 \text{ kHz} = 46.875$

Therefore, the real prescaler value is 47. This results in a bit clock error equal to:

$f_{bit\_error} = (f_{bit} – f_{Audio\ In}/P_{real})/f_{bit} \times 100 = (256 \text{ kHz} – 12 \text{ MHz}/47) / 256 \text{ kHz} \times 100 = 0.27\%$

24.4 Frame Clock Generation

The clock for the frame synchronization signals is derived from the bit clock of the audio interface. A 7-bit prescaler is used to divide the bit clock to generate the frame sync clock for the receive and transmit operations. The bit clock is divided by FCPRS + 1. In other words, the value software must write into the ACCR.FCPRS field is equal to the bit number per frame minus one. The frame may be longer than the valid data word but it must be equal to or larger than the 8-bit or 16-bit word. Even if 13-, 14-, or 15-bit data is being used, the frame width must always be at least 16 bits wide.

In addition, software can specify the length of a long frame sync signal. A long frame sync signal can be either 6, 13, 14, 15, or 16 bits long, depending on the external codec being used. The frame sync length can be configured by the Frame Sync Length field (FSL) in the AGCR register.

24.5 Audio Interface Operation

24.5.1 Clock Configuration

An auxiliary clock (generated by the Clock module described in Section 14.7) must be configured to provide a 12 MHz input clock as a time base for the AAI module. The CSS bit in the ACCR register selects whether the input clock is Auxiliary Clock 1 or Auxiliary Clock 8.

24.5.2 Interrupts

The interrupt logic of the AAI combines up to four interrupt sources and generates one interrupt request signal to the Interrupt Control Unit (ICU).

The four interrupt sources are:

- RX FIFO Overrun – AISCR.RXEIP = 1
- RX FIFO Almost Full (Warning Level) – AISCR.RXIP = 1
- TX FIFO Under run – AISCR.TXEIP = 1
- TX FIFO Almost Empty (Warning Level) – AISCR.TXIP=1

In addition to the dedicated input to the ICU for handling these interrupt sources, the Serial Frame Sync (SFS) signal is the WUI26 input to the MIWU (see Section 17), which can be programmed to assert edge-triggered interrupts.

Figure 24-6 shows the interrupt structure of the AAI.

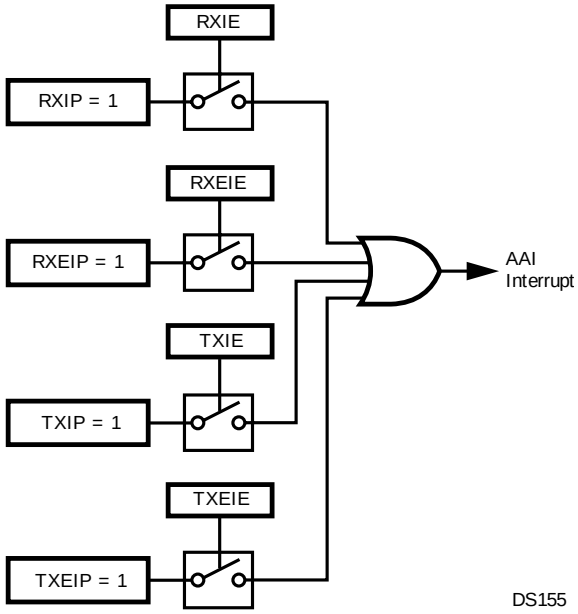


Figure 24-6. AAI Interrupt Structure

24.5.3 Normal Mode

In normal mode, each frame sync signal marks the beginning of a new frame and also the beginning of a new slot, since each frame only consists of one slot. All 16 receive and transmit FIFO locations hold data for the same (and only) slot of a frame. If 8-bit data are transferred, only the low byte of each 16-bit FIFO location holds valid data.

24.5.4 Transmit

Once the interface has been enabled, transmit transfers are initiated automatically at the beginning of every frame. The beginning of a new frame is identified by a frame sync pulse. Following the frame sync pulse, the data is shifted out from the ATSR to the STD pin on the positive edge of the transmit data shift clock (SCK).

DMA Operation

When a complete data word has been transmitted through the STD pin, a new data word is reloaded from the transmit DMA register 0 (ATDR0). A DMA request is asserted when the ATDR0 register is empty. If a new data word must be transmitted while the ATDR0 register is still empty, the previous data will be re-transmitted.

FIFO Operation

When a complete data word has been transmitted through the STD pin, a new data word is loaded from the transmit FIFO from the current location of the Transmit FIFO Read Pointer (TRP). After that, the TRP is automatically incremented by 1.

A write to the Audio Transmit FIFO Register (ATFR) results in a write to the transmit FIFO at the current location of the Transmit FIFO Write Pointer (TWP). After every write operation to the transmit FIFO, TWP is automatically incremented by 1.

When the TRP is equal to the TWP and the last access to the FIFO was a read operation (a transfer to the ATSR), the transmit FIFO is empty. When an additional read operation from the FIFO to ATSR is performed (while the FIFO is already empty), a transmit FIFO underrun occurs. In this event, the read pointer (TRP) will be decremented by 1 (incremented by 15) and the previous data word will be transmitted again. A transmit FIFO underrun is indicated by the TXU bit in the Audio Interface Transmit Status and Control Register (ATSCR). Also, no transmit interrupt will be generated.

When the TRP is equal to the TWP and the last access to the FIFO was a write operation (to the ATFR), the FIFO is full. If an additional write to ATFR is performed, a transmit FIFO overrun occurs. This error condition is not prevented by hardware. Software must ensure that no transmit overrun occurs.

The transmit frame synchronization pulse on the SFS pin and the transmit shift clock on the SCK pin may be generated internally, or they can be supplied by an external source.

24.5.5 Receive

At the receiver, the received data on the SRD pin is shifted into ARSR on the negative edge of SRCLK (or SCK in synchronous mode), following the receive frame sync pulse, SRFS (or SFS in synchronous mode).

DMA Operation
When a complete data word has been received through the SRD pin, the new data word is copied to the receive DMA register 0 (ARDR0). A DMA request is asserted when the ARDR0 register is full. If a new data word is received while the ARDR0 register is still full, the ARDR0 register will be overwritten with the new data.

FIFO Operation
When a complete word has been received, it is transferred to the receive FIFO at the current location of the Receive FIFO Write Pointer (RWP). Then, the RWP is automatically incremented by 1.

A read from the Audio Receive FIFO Register (ARFR) results in a read from the receive FIFO at the current location of the Receive FIFO Read Pointer (RRP). After every read operation from the receive FIFO, the RRP is automatically incremented by 1.

When the RRP is equal to the RWP and the last access to the FIFO was a copy operation from the ARFR, the receive FIFO is full. When a new complete data word has been shifted into ARSR while the receive FIFO was already full, the shift register overruns. In this case, the new data in the ARSR will not be copied into the FIFO and the RWP will not be incremented. A receive FIFO overrun is indicated by the RXO bit in the Audio Interface Receive Status and Control Register (ARSCR). No receive interrupt will be generated.

When the RWP is equal to the RRP and the last access to the receive FIFO was a read from the ARFR, a receive FIFO underrun has occurred. This error condition is not prevented by hardware. Software must ensure that no receive underrun occurs.

The receive frame synchronization pulse on the SRFS pin (or SFS in synchronous mode) and the receive shift clock on the SRCLK (or SCK in synchronous mode) may be generated internally, or they can be supplied by an external source.

24.5.6 Network Mode

In network mode, each frame sync signal marks the beginning of new frame. Each frame can consist of up to four slots. The audio interface operates in a similar way to normal mode, however, in network mode the transmitter and receiver can be assigned to specific slots within each frame as described below.

24.5.7 Transmit

The transmitter only shifts out data during the assigned slot. During all other slots the STD output is in TRI-STATE mode.

24.5.7.1 DMA Operation

When a complete data word has been transmitted through the STD pin, a new data word is reloaded from the corresponding transmit DMA register n (ATDRn). A DMA request is asserted when ATDRn is empty. If a new data word must be transmitted in a slot n while ATDRn is still empty, the previous slot n data will be retransmitted.

24.5.7.2 FIFO Operation

When a complete data word has been transmitted through the STD pin, a new data word is reloaded from the transmit FIFO from the current location of the Transmit FIFO Read Pointer (TRP). After that, the TRP is automatically incremented by 1. Therefore, the audio data to be transmitted in the next slot of the frame is read from the next FIFO location.

A write to the Audio Transmit FIFO Register (ATFR) results in a write to the transmit FIFO at the current location of the Transmit FIFO Write Pointer (TWP). After every write operation to the transmit FIFO, the TWP is automatically incremented by 1.

When the TRP is equal to the TWP and the last access to the FIFO was a read operation (transfer to the ATSR), the transmit FIFO is empty. When an additional read operation from the FIFO to the ATSR is performed (while the FIFO is already empty), a transmit FIFO underrun occurs. In this case, the read pointer (TRP) will be decremented by 1 (incremented by 15) and the previous data word will be transmitted again. A transmit FIFO underrun is indicated by the TXU bit in the Audio Interface Transmit Status and Control Register (ATSCR). No transmit interrupt will be generated (even if enabled).

If the current TRP is equal to the TWP and the last access to the FIFO was a write operation (to the ATFR), the FIFO is full. If an additional write to the ATFR is performed, a transmit FIFO overrun occurs. This error condition is not prevented by hardware. Software must ensure that no transmit overrun occurs.

The transmit frame synchronization pulse on the SFS pin and the transmit shift clock on the SCK pin may be generated internally, or they can be supplied by an external source.

24.5.8 Receive

The receive shift register (ARSR) receives data words of all slots in the frame, regardless of the slot assignment of the interface. However, only those ARSR contents are transferred to the receive FIFO or DMA receive register which were received during the assigned time slots. A receive interrupt or DMA request is initiated when this occurs.

24.5.8.1 DMA Operation

When a complete data word has been received through the SRD pin in a slot n, the new data word is transferred to the corresponding receive DMA register n (ARDRn). A DMA request is asserted when the ARDRn register is full. If a new slot n data word is received while the ARDRn register is still full, the ARDRn register will be overwritten with the new data.

24.5.8.2 FIFO Operation

When a complete word has been received, it is transferred to the receive FIFO at the current location of the Receive FIFO Write Pointer (RWP). After that, the RWP is automatically incremented by 1. Therefore, data received in the next slot is copied to the next higher FIFO location.

A read from the Audio Receive FIFO Register (ARFR) results in a read from the receive FIFO at the current location of the Receive FIFO Read Pointer (RRP). After every read operation from the receive FIFO, the RRP is automatically incremented by 1.

When the RRP is equal to the RWP and the last access to the FIFO was a transfer to the ARFR, the receive FIFO is full. When a new complete data word has been shifted into the ARSR while the receive FIFO was already full, the shift register overruns. In this case, the new data in the ARSR will not be transferred to the FIFO and the RWP will not be incremented. A receive FIFO overrun is indicated by the RXO bit in the Audio Interface Receive Status and Control Register (ARSCR). No receive interrupt will be generated (even if enabled).

When the current RWP is equal to the TWP and the last access to the receive FIFO was a read from ARFR, a receive FIFO underrun has occurred. This error condition is not prevented by hardware. Software must ensure that no receive underrun occurs.

The receive frame synchronization pulse on the SRFS pin (or SFS in synchronous mode) and the receive shift clock on the SRCLK (or SCK in synchronous mode) may be generated internally, or they can be supplied by an external source.

24.6 Communication Options

24.6.1 Data Word Length

The word length of the audio data can be selected to be either 8 or 16 bits. In 16-bit mode, all 16 bits of the transmit and receive shift registers (ATSR and ARSR) are used. In 8-bit mode, only the lower 8 bits of the transmit and receive shift registers (ATSR and ARSR) are used.

24.6.2 Frame Sync Signal

The audio interface can be configured to use either long or short frame sync signals to mark the beginning of a new data frame. If the corresponding Frame Sync Select (FSS) bit in the Audio Control and Status register is clear, the receive and/or transmit path generates or recognizes short frame sync pulses with a length of one bit shift clock period. When these short frame sync pulses are used, the transfer of the first data bit or the first slot begins at the first positive edge of the shift clock after the negative edge on the frame sync pulse.

If the corresponding Frame Sync Select (FSS) bit in the Audio Control and Status register is set, the receive and/or transmit path generates or recognizes long frame sync pulses. For 8-bit data, the frame sync pulse generated will be 6 bit shift clock periods long, and for 16-bit data the frame sync pulse can be configured to be 13, 14, 15, or 16 bit shift clock periods long. When receiving frame sync, it should be active on the first bit of data and stay active for a least two bit clock periods. It must go low for at least one bit clock period before starting a new frame. When long frame sync pulses are used, the transfer of the first word (first slot) begins at the first positive edge of the bit shift clock after the positive edge of the frame sync pulse. Figure 24-7 shows examples of short and long frame sync pulses.

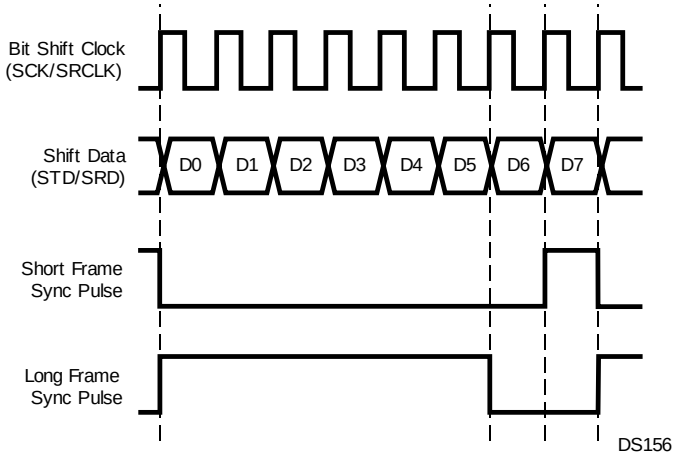


Figure 24-7. Short and Long Frame Sync Pulses

Some codecs require an inverted frame sync signal. This is available by setting the Inverted Frame Sync bit in the AGCR register.

24.6.3 Audio Control Data

The audio interface provides the option to fill a 16-bit slot with up to three data bits if only 13, 14, or 15 PCM data bits are transmitted. These additional bits are called audio control data and are appended to the PCM data stream. The AAI can be configured to append either 1, 2, or 3 audio control bits to the PCM data stream. The number of audio data bits to be used is specified by the 2-bit Audio Control On (ACO) field. If the ACO field is not equal to 0, the specified number of bits are taken from the Audio Control Data field (ACD) and appended to the data stream during every transmit operation. The ADC0 bit is the first bit added to the transmit data stream after the last PCM data bit. Typically, these bits are used for gain control, if this feature is supported by the external PCM codec. Figure 24-8 shows a 16-bit slot comprising a 13-bit PCM data word plus three audio control bits.

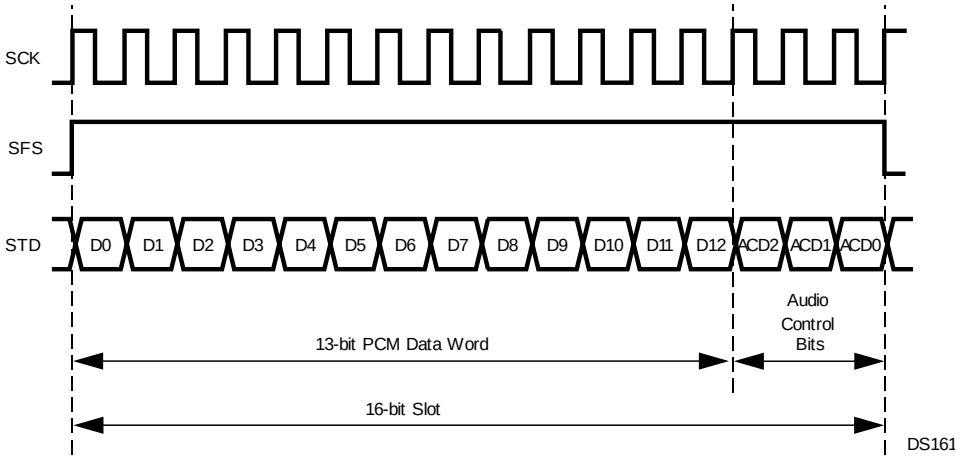


Figure 24-8. Audio Slot with Audio Control Data

24.6.4 IOM-2 Mode

The AAI can operate in a special IOM-2 compatible mode to allow to connect to an external ISDN controller device. In this IOM-2 mode, the AAI can only operate as a slave, i.e., the bit clock and frame sync signal is provided by the ISDN controller. The AAI only supports the B1 and B2 data of the IOM-2 channel 0, but ignores the other two IOM-2 channels. The AAI handles the B1 and B2 data as one 16-bit data word.

The IOM-2 interface has the following properties:

- Bit clock of 1536 kHz (output from the ISDN controller)
- Frame repetition rate of 8 ksps (output from the ISDN controller)
- Double-speed bit clock (one data bit is two bit clocks wide)
- B1 and B2 data use 8-bit log PCM format
- Long frame sync pulse

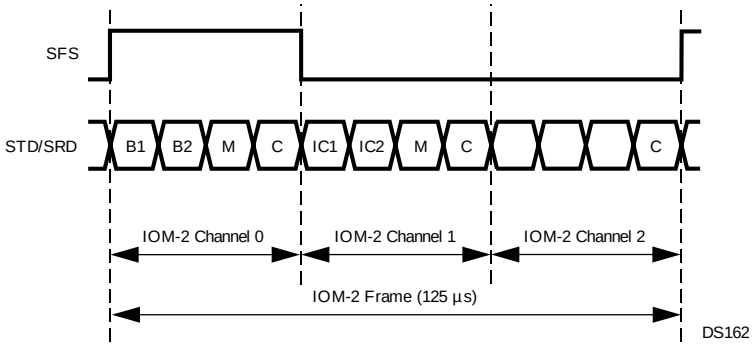


Figure 24-9. IOM-2 Frame Structure

Figure 24-10 shows the connections between an ISDN controller and a CP3SP33 using a standard IOM-2 interface for the B1/B2 data communication and the external bus interface (IO Expansion) for controlling the ISDN controller.

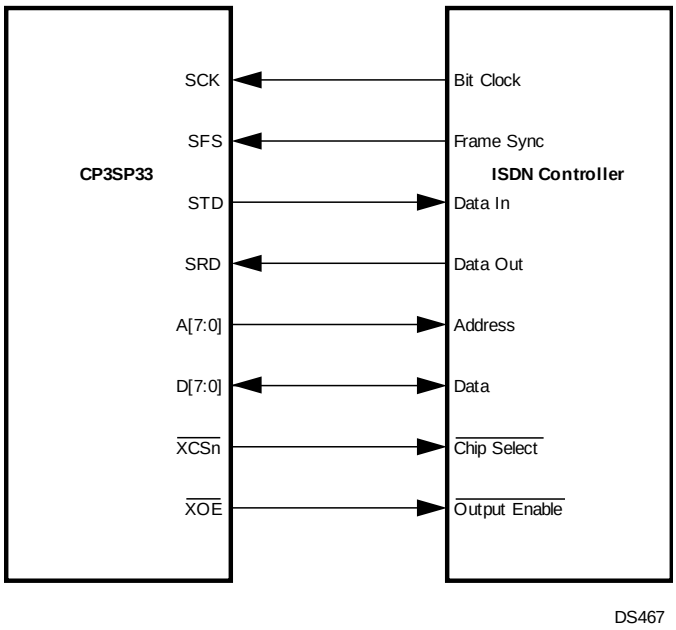


Figure 24-10. CP3SP33/ISDN Controller Connections

To connect the AAI to an ISDN controller through an IOM-2 compatible interface, the AAI needs to be configured in this way:

- The AAI must be in IOM-2 Mode (AGCR.IOM2 = 1).
- The AAI operates in synchronous mode (AGCR.ASS = 0).
- The AAI operates as a slave, therefore the bit clock and frame sync source selection must be set to external (ACGR.IEFS = 1, ACGR.IEBC = 1).
- The frame sync length must be set to long frame sync (ACGR.FSS = 1).
- The data word length must be set to 16-bit (AGCR.DWL = 1).
- The AAI must be set to normal mode (AGCR.SCS[1:0] = 0).

24.6.5 Loopback Mode

In loopback mode, the STD and SRD pins are internally connected together, so data shifted out through the ATSR register will be shifted into the ARSR register. This mode may be used for development, but it also allows testing the transmit and receive path without external circuitry, for example during Built-In-Self-Test (BIST).

24.6.6 Freeze Mode

When the Freeze mode is entered, the audio interface exhibits the following behavior:

- The receive FIFO or receive DMA registers are not updated with new data.
- The receive status bits (RXO, RXE, RXF, and RXAF) are not changed, even though the receive FIFO or receive DMA registers are read
- The transmit shift register (ATSR) is not updated with new data from the transmit FIFO or transmit DMA registers.
- The transmit status bits (TXU, TXF, TXE, and TXAE) are not changed, even though the transmit FIFO or transmit DMA registers are written.

24.7 Audio Interface Registers

Table 24-1. Audio Interface Registers

NAME	ADDRESS	DESCRIPTION
ARFR	FF 5000h	Audio Receive FIFO Register
ARDR0	FF 5004h	Audio Receive DMA Register 0
ARDR1	FF 5008h	Audio Receive DMA Register 1
ARDR2	FF 500Ch	Audio Receive DMA Register 2
ARDR3	FF 5010h	Audio Receive DMA Register 3
ATFR	FF 5014h	Audio Transmit FIFO Register
ATDR0	FF 5018h	Audio Transmit DMA Register 0
ATDR1	FF 501Ch	Audio Transmit DMA Register 1
ATDR2	FF 5020h	Audio Transmit DMA Register 2
ATDR3	FF 5024h	Audio Transmit DMA Register 3
AGCR	FF 5028h	Audio Global Configuration Register
AISCR	FF 502Ch	Audio Interrupt Status and Control Register
ARSCR	FF 5030h	Audio Receive Status and Control Register
ATSCR	FF 5034h	Audio Transmit Status and Control Register
ACCR	FF 5038h	Audio Clock Control Register
ADMACR	FF 503Ch	Audio DMA Control Register

24.7.1 Audio Receive FIFO Register (ARFR)

The Audio Receive FIFO register shows the receive FIFO location currently addressed by the Receive FIFO Read Pointer (RRP). The receive FIFO receives 8-bit or 16-bit data from the Audio Receive Shift Register (ARSR), when the ARSR is full.

In 8-bit mode, only the lower byte of the ARFR is used, and the upper byte contains undefined data. In 16-bit mode, a 16-bit word is copied from ARSR into the receive FIFO. The APB bus master has read-only access to the receive FIFO, represented by the ARFR register. After reset, the receive FIFO (ARFR) contains undefined data.

15	8	7	0
ARFH		ARFL	

ARFL The Audio Receive FIFO Low Byte shows the lower byte of the receive FIFO location currently addressed by the Receive FIFO Read Pointer (RRP).

ARFH The Audio Receive FIFO High Byte shows the upper byte of the receive FIFO location currently addressed by the Receive FIFO Read Pointer (RRP). In 8-bit mode, ARFH contains undefined data.

24.7.2 Audio Receive DMA Register n (ARDRn)

The ARDRn register contains the data received within slot n, assigned for DMA support. In 8-bit mode, only the lower 8-bit portion of the ARDRn register is used, and the upper byte contains undefined data. In 16-bit mode, a 16-bit word is transferred from the Audio Receive Shift Register (ARSR) into the ARDRn register. The APB bus master, typically a DMA controller, has read-only access to the receive DMA registers. After reset, these registers are clear.

15	8	7	0
ARDH		ARDL	

ARDL The Audio Receive DMA Low Byte field receives the lower byte of the audio data copied from the ARSR.

ARDH In 16-bit mode, the Audio Receive DMA High Byte field receives the upper byte of the audio data word copied from ARSR. In 8-bit mode, the ARDH register holds undefined data.

24.7.3 Audio Transmit FIFO Register (ATFR)

The ATFR register shows the transmit FIFO location currently addressed by the Transmit FIFO Write Pointer (TWP). The Audio Transmit Shift Register (ATSR) receives 8-bit or 16-bit data from the transmit FIFO, when the ATSR is empty. In 8-bit mode, only the lower 8-bit portion of the ATSR is used, and the upper byte is ignored (not transferred into the ATSR). In 16-bit mode, a 16-bit word is copied from the transmit FIFO into the ATSR. The APB bus master has write-only access to the transmit FIFO, represented by the ATFR register. After reset, the transmit FIFO (ATFR) contains undefined data.

15	8	7	0
ATFH		ATFL	

ATFL The Audio Transmit Low Byte field represents the lower byte of the transmit FIFO location currently addressed by the Transmit FIFO Write Pointer (TWP).

ATFH In 16-bit mode, the Audio Transmit FIFO High Byte field represents the upper byte of the transmit FIFO location currently addressed by the Transmit FIFO Write Pointer (TWP). In 8-bit mode, the ATFH field is not used.

24.7.4 Audio Transmit DMA Register n (ATDRn)

The ATDRn register contains the data to be transmitted in slot n, assigned for DMA support. In 8-bit mode, only the lower 8-bit portion of the ATDRn register is used, and the upper byte is ignored (not transferred into the ATSR). In 16-bit mode, the whole 16-bit word is transferred into the ATSR. The APB bus master, typically a DMA controller, has write-only access to the transmit DMA registers. After reset, these registers are clear.

15	8	7	0
ATDH		ATDL	

ATDL The Audio Transmit DMA Low Byte field holds the lower byte of the audio data.

ATDH In 16-bit mode, the Audio Transmit DMA High Byte field holds the upper byte of the audio data word. In 8-bit mode, the ATDH field is ignored.

24.7.5 Audio Global Configuration Register (AGCR)

The AGCR register controls the basic operation of the interface. The APB bus master has read/write access to the AGCR register. After reset, this register is clear.

7	6	5	4	3	2	1	0
IEBC	FSS	IEFS	SCS		LPB	DWL	ASS

15	14	13	12	11	10	9	8
CLKEN	AAIEN	IOM2	IFS	FSL		CTF	CRF

ASS The Asynchronous/Synchronous Mode Select bit controls whether the audio interface operates in Asynchronous or in Synchronous mode.
0 – Synchronous mode.
1 – Asynchronous mode.

DWL The Data Word Length bit controls whether the data word has a length of 8 or 16 bits.
0 – 8-bit length.
1 – 16-bit length.

LPB The Loop Back bit enables the loop back mode. In this mode, the SRD and STD pins are internally connected. After reset the LPB bit is clear, so by default the loop back mode is disabled.
0 – Loop back mode disabled.
1 – Loop back mode enabled.

SCS The Slot Count Select field specifies the number of slots within each frame. If the number of slots per frame is equal to 1, the audio interface operates in normal mode. If the number of slots per frame is greater than 1, the interface operates in network mode.

SCS	NUMBER OF SLOTS PER FRAME	MODE
0	1	Normal mode
1	2	Network mode
10	3	Network mode
11	4	Network mode

IEFS The Internal/External Frame Sync bit controls, whether the frame sync signal for the receiver and transmitter are generated internally or provided from an external source.
0 – Internal frame synchronization signal.
1 – External frame synchronization signal.

FSS The Frame Sync Select bit controls whether the interface (receiver and transmitter) uses long or short frame synchronization signals.
0 – Short (bit length) frame synchronization signal.
1 – Long (word length) frame synchronization signal.

IEBC The Internal/External Bit Clock bit controls whether the bit clocks for receiver and transmitter are generated internally or provided from an external source.
0 – Internal bit clock.
1 – External bit clock.

CRF The Clear Receive FIFO bit is used to clear the receive FIFO. When this bit is written with a 1, all pointers of the receive FIFO are set to their reset state. After updating the pointers, the CRF bit will automatically be cleared again.
0 – Writing 0 has no effect.
1 – Writing 1 clears the receive FIFO.

CTF The Clear Transmit FIFO bit is used to clear the transmit FIFO. When this bit is written with a 1, all pointers of the transmit FIFO are set to their reset state. After updating the pointers, the CTF bit will automatically be cleared again.
0 – Writing 0 has no effect.
1 – Writing 1 clears the transmit FIFO.

FSL The Frame Sync Length field specifies the length of the frame synchronization signal, when a long frame sync signal (FSS = 1) and a 16-bit data word length
(DWL = 1) are used. If an 8-bit data word length is used, long frame syncs are always 6 bit clocks in length.

FSL	FRAME SYNC LENGTH
00	13 bit clocks
01	14 bit clocks
10	15 bit clocks
11	16 bit clocks

IFS The Inverted Frame Sync bit controls the polarity of the frame sync signal.
0 – Active-high frame sync signal.
1 – Active-low frame sync signal.

IOME The IOM-2 Mode bit selects the normal PCM interface mode or a special IOM-2 mode used to connect to external ISDN controller devices. The AAI can only operate as a slave in the IOM-2 mode, i.e. the bit clock and frame sync signals are provided by the ISDN controller. If the IOM2 bit is clear, the AAI operates in the normal PCM interface mode used to connect to external PCM codecs and other PCM audio devices.
0 – IOM-2 mode disabled.
1 – IOM-2 mode enabled.

AAIEN The AAI Enable bit controls whether the Advanced Audio Interface is enabled. When the AAI is disabled, all AAI registers remain accessible.
0 – AAI module disabled.
1 – AAI module enabled.

CLKEN The Clock Enable bit controls whether the Advanced Audio Interface clock is enabled. The CLKEN bit must be set to allow access to any AAI register. It must also be set before any other bit of the AGCR can be set. The CLKEN bit is clear after reset.
0 – AAI module clock disabled.
1 – AAI module clock enabled.

24.7.6 Audio Interrupt Status and Control Register (AISCR)

The AISCR register is used to specify the source and the conditions, when the audio interface interrupt is asserted to the Interrupt Control Unit. It also holds the interrupt pending bits and the corresponding interrupt clear bits for each audio interface interrupt source. The APB bus master has read/ write access to the AISCR register. After reset, this register is clear.

7	6	5	4	3	2	1	0
TXEIP	TXIP	RXEIP	RXIP	TXEIE	TXIE	RXEIE	RXIE

15	12	11	10	9	8
Reserved		TXEIC	TXIC	RXEIC	RXIC

RXIC The Receive Interrupt Enable bit controls whether receive interrupts are generated. If the RXIE bit is clear, no receive interrupt will be generated.
0 – Receive interrupt disabled.
1 – Receive interrupt enabled.

RXEIE The Receive Error Interrupt Enable bit controls whether receive error interrupts are generated. Setting this bit enables a receive error interrupt, when the Receive Buffer Overrun (RXOR) bit is set. If the RXEIE bit is clear, no receive error interrupt will be generated.
0 – Receive error interrupt disabled.
1 – Receive error interrupt enabled.

TXIE The Transmit Interrupt Enable bit controls whether transmit interrupts are generated. Setting this bit enables a transmit interrupt, when the Transmit Buffer Almost Empty (TXAE) bit is set. If the TXIE bit is clear, no interrupt will be generated.
0 – Transmit interrupt disabled.
1 – Transmit interrupt enabled.

TXEIE The Transmit Error Interrupt Enable bit controls whether transmit error interrupts are generated. Setting this bit to 1 enables a transmit error interrupt, when the Transmit Buffer Underrun (TXUR) bit is set. If the TXEIE bit is clear, no transmit error interrupt will be generated.
0 – Transmit error interrupt disabled.
1 – Transmit error interrupt enabled.

RSIP The Receive Interrupt Pending bit indicates that a receive interrupt is currently pending. The RXIP bit is cleared by writing a 1 to the RXIC bit. The RXIP bit provides read-only access.
0 – No receive interrupt pending.
1 – Receive interrupt pending.

RXEIP The Receive Error Interrupt Pending bit indicates that a receive error interrupt is currently pending. The RXEIP bit is cleared by writing a 1 to the RXEIC bit. The RXEIP bit provides read-only access.
0 – No receive error interrupt pending.
1 – Receive error interrupt pending.

TXIP The Transmit Interrupt Pending bit indicates that a transmit interrupt is currently pending. The TXIP bit is cleared by writing a 1 to the TXIC bit. The TXIP bit provides read-only access.
– No transmit interrupt pending.
1 – Transmit interrupt pending.

TXEIP Transmit Error Interrupt Pending. This bit indicates that a transmit error interrupt is currently pending. The TXEIP bit is cleared by software by writing a 1 to the TXEIC bit. The TXEIP bit provides read-only access.
0 – No transmit error interrupt pending.
1 – Transmit error interrupt pending.

RXIC The Receive Interrupt Clear bit is used to clear the RXIP bit.
0 – Writing a 0 to the RXIC bit is ignored.
1 – Writing a 1 clears the RXIP bit.

RXEIC The Receive Error Interrupt Clear bit is used to clear the RXEIP bit.
0 – Writing a 0 to the RXEIC bit is ignored.
1 – Writing a 1 clears the RXEIP bit.

TXIC The Transmit Interrupt Clear bit is used to clear the TXIP bit.
0 – Writing a 0 to the TXIC bit is ignored.
1 – Writing a 1 clears the TXIP bit.

TXEIC The Transmit Error Interrupt Clear bit is used to clear the TXEIP bit.
0 – Writing a 0 to the TXEIC bit is ignored.
1 – Writing a 1 clears the TXEIP bit.

24.7.7 Audio Receive Status and Control Register (ARSCR)

The ARSCR register is used to control the operation of the receiver path of the audio interface. It also holds bits which report the current status of the receive FIFO. The APB bus master has read/write access to the ARSCR register. At reset, this register is cleared to 0000h, but after enabling the AAI clock it becomes 0004h.

15 12	11 8	7 4	3	2	1	0
RXFWL	RXDSA	RXSA	RXO	RXE	RXF	RXAF

RXAF The Receive Buffer Almost Full bit is set when the number of data bytes/words in the receive buffer is equal to the specified warning limit.
0 – Receive FIFO below warning limit.
1 – Receive FIFO is almost full.

RXF The Receive Buffer Full bit is set when the receive buffer is full. The RXF bit is set when the RWP is equal to the RRP and the last access was a write to the FIFO.
0 – Receive FIFO is not full.
1 – Receive FIFO full.

RXE The Receive Buffer Empty bit is set when the RRP is equal to the RWP and the last access to the FIFO was a read operation (read from ARDR).
0 – Receive FIFO is not empty.
1 – Receive FIFO is empty.

RXO The Receive Overflow bit indicates that a receive shift register has overrun. This occurs, when a completed data word has been shifted into ARSR, while the receive FIFO was already full (the RXF bit was set). In this case, the new data in ARSR will not be copied into the FIFO and the RWP will not be incremented. Also, no receive interrupt and DMA request will generated.
0 – No overflow has occurred.
1 – Overflow has occurred.

RXSA The Receive Slot Assignment field specifies which slots are recognized by the receiver of the audio interface. Multiple slots may be enabled. If the frame consists of less than 4 slots, the RXSA bits for unused slots are ignored. For example, if a frame only consists of 2 slots, RXSA bits 2 and 3 are ignored. The following table shows the slot assignment scheme.

RXSA BIT	SLOTS ENABLED
RXSA0	0
RXSA1	1
RXSA2	2
RXSA3	3

After reset the RXSA field is clear, so software must load the correct slot assignment.

RXDSA The Receive DMA Slot Assignment field specifies which slots (audio channels) are supported by DMA. If the RXDSA bit is set for an assigned slot (RXSAn = 1), the data received within this slot will not be transferred into the receive FIFO, but will instead be written into the corresponding Receive DMA data register (ARDRn). If the ARDRn register is full and the RMA bit is set, a DMA request is asserted. If the RXSA bit for a slot is clear, the RXDSA bit is ignored. The following table shows the DMA slot assignment scheme. RXDSA3 must be 0.

RXDSA BIT	SLOTS ENABLED FOR DMA
RXDSA0	0
RXDSA1	1
RXDSA2	2

RXFWL The Receive FIFO Warning Level field specifies when a receive interrupt is asserted. A receive interrupt is asserted, when the number of bytes/words in the receive FIFO is greater than the warning level value. An RXFWL value of 0 means that a receive interrupt is asserted if one or more bytes/words are in the RX FIFO. After reset, the RXFWL bit is clear.

24.7.8 Audio Transmit Status and Control Register (ATSCR)

The ATSCR register controls the basic operation of the interface. It also holds bits which report the current status of the audio communication. The APB bus master has read/ write access to the ATSCR register. At reset, this register is loaded with F000h, but after enabling the AAI clock it becomes F003h.

15 12	11 8	7 4	3	2	1	0
TXFWL	TXDSA	TXSA	TXU	TXF	TXF	TXAF

TXAE The Transmit FIFO Almost Empty bit is set when the number of data bytes/words in transmit buffer is equal to the specified warning limit.
0 – Transmit FIFO above warning limit.
1 – Transmit FIFO at or below warning limit.

TXE The Transmit FIFO Empty bit is set when the transmit buffer is empty. The TXE bit is set to one every time the TRP is equal to the TWP and the last access to the FIFO was read operation (into ATSR).
0 – Transmit FIFO not empty.
1 – Transmit FIFO empty.

TXF The Transmit FIFO Full bit is set when the TWP is equal to the TRP and the last access to the FIFO was write operation.
0 – Transmit FIFO not full.
1 – Transmit FIFO full.

TXU The Transmit Underflow bit indicates that the transmit shift register (ATSR) has underrun. This occurs when the transmit FIFO was already empty and a complete data word has been transferred. In this case, the TRP will be decremented by 1 and the previous data will be retransmitted. No transmit interrupt and no DMA request will be generated.
0 – Transmit underrun occurred.
1 – Transmit underrun did not occur.

TXSA The Transmit Slot Assignment field specifies during which slots the transmitter is active and drives data through the STD pin. The STD pin is in high impedance state during all other slots. If the frame consists of less than 4 slots, the TXSA bits for unused slots are ignored. For example, if a frame only consists of 2 slots, TXSA bits 2 and 3 are ignored. The following table shows the slot assignment scheme.

TXSA BIT	SLOTS ENABLED
TXSA0	0
TXSA1	1
TXSA2	2
TXSA3	3

After reset the TXSA field is clear, so software must load the correct slot assignment.

TXDSA The Transmit DMA Slot Assignment field specifies which slots (audio channels) are supported by DMA. If the TXDSA bit is set for an assigned slot (TXSAn = 1), the data to be transmitted within this slot will not be read from the transmit FIFO, but will instead be read from the corresponding Transmit DMA data register (ATDRn). A DMA request is asserted when the ATDRn register is empty. If the TXSA bit for a slot is clear, the TXDSA bit is ignored. The following table shows the DMA slot assignment scheme. TXDSA3 must be 0.

TXDSA BIT	SLOTS ENABLED FOR DMA
TXDSA0	0
TXDSA1	1
TXDSA2	2

TXFWL The Transmit FIFO Warning Level field specifies when a transmit interrupt is asserted. A transmit interrupt is asserted when the number of bytes or words in the transmit FIFO is equal or less than the warning level value. A TXFWL value of Fh means that a transmit interrupt is asserted if one or more bytes or words are available in the transmit FIFO. At reset, the TXFWL field is loaded with Fh.

24.7.9 Audio Clock Control Register (ACCR)

The ACCR register is used to control the bit timing of the audio interface. After reset, this register is clear.

15 – 8	7 – 1	0
BCPRS	FCPRS	CSS

CSS The Clock Source Select bit selects one out of two possible clock sources for the audio interface.
0 – Auxiliary Clock 1.
1 – Auxiliary Clock 8.

FCPRS The Frame Clock Prescaler is used to divide the bit clock to generate the frame clock for the receive and transmit operations. The bit clock is divided by (FCPRS + 1). After reset, the FCPRS field is clear. The maximum allowed bit clock rate to achieve an 8 kHz frame clock is 1024 kHz. This value must be set correctly even if the frame sync is generated externally.

BCPRS The Bit Clock Prescaler is used to divide the audio interface clock (selected by the CSS bit) to generate the bit clock for the receive and transmit operations. The audio interface input clock is divided by (BCPRS + 1). After reset, the BCPRS7:0 bits are clear.

24.7.10 Audio DMA Control Register (ADMACR)

The ADMACR register is used to control the DMA support of the audio interface. In addition, it is used to configure the automatic transmission of the audio control bits. After reset, this register is clear.

15 – 13	12 – 11	10 – 8	7 – 4	3 – 0
Reserved	ACO	ACD	TMD	RMD

RMD The Receive Master DMA field specify which slots (audio channels) are supported by DMA, i.e., when a DMA request is asserted to the DMA controller. If the RMDn bit is set for an assigned slot (RXDSAn = 1), a DMA request is asserted when the ARDRn register is full. If the RXDSAn bit for a slot is clear, the RMDn bit is ignored. The following table shows the receive DMA request scheme.

RMD	DMA REQUEST CONDITION
0000	None
0001	ARDR0 full
0010	ARDR1 full
0011	ARDR0 full or ARDR1 full
x1xx	Not supported on CP3SP33
1xxx	

TMD The Transmit Master DMA field specifies which slots (audio channels) are supported by DMA, i.e., when a DMA request is asserted to the DMA controller. If the TMD field is set for an assigned slot (TXDSAn = 1), a DMA request is asserted when the ATDRn register is empty. If the TXDSA bit for a slot is clear, the TMD field is ignored. The following table shows the transmit DMA request scheme.

TMD FIELD	DMA REQUEST CONDITION
0000	None
0001	ATDR0 empty
0010	ATDR1 empty
0011	ATDR0 empty or ATDR1 empty
x1xx	Not supported on CP3SP33
1xxx	

ACD The Audio Control Data field is used to fill the remaining bits of a 16-bit slot if only 13, 14, or 15 bits of PCM audio data are transmitted.

ACO The Audio Control Output field controls the number of control bits appended to the PCM data word.
00 – No Audio Control bits are appended.
01 – Append ACD0.
10 – Append ACD1:0.
11 – Append ACD2:0.

24.8 Usage Example

The following example shows the AAI being used to interface to two single-channel codecs. The interface has the following characteristics:

- Synchronous mode
- 8-bit data word
- Network mode with 4 slots per frame
- Slot 0 is assigned for Audio interface - Codec 1 communication (receive and transmit)
- Slot 1 assigned for Audio interface - Codec 2 communication (receive and transmit)
- Slots 2 and 3 unused
- Internal long frame synch pulse
- Internal shift clock, adjusted to achieve a 64 kb/s log PCM audio quality
- Frame rate set to 8 kHz (see below)
- Bit clock set to 256 kHz (see below)

It is not currently possible for the AAI to produce an exact 8 kHz frame rate together with an exact 256 kHz bit rate from a 12-MHz clock input. If this is used in a system containing a component that requires data at a frame rate of precisely 8 kHz (including the CVSD/PCM converter), then an audio artifact may be produced. If a PLL clock generator is available, it can be used to provide an input clock of the frequency required to provide suitable derivative frequencies.

Figure 24-11 shows the connections between the AAI and the external codecs. Figure 24-12 shows the timing of the interface.

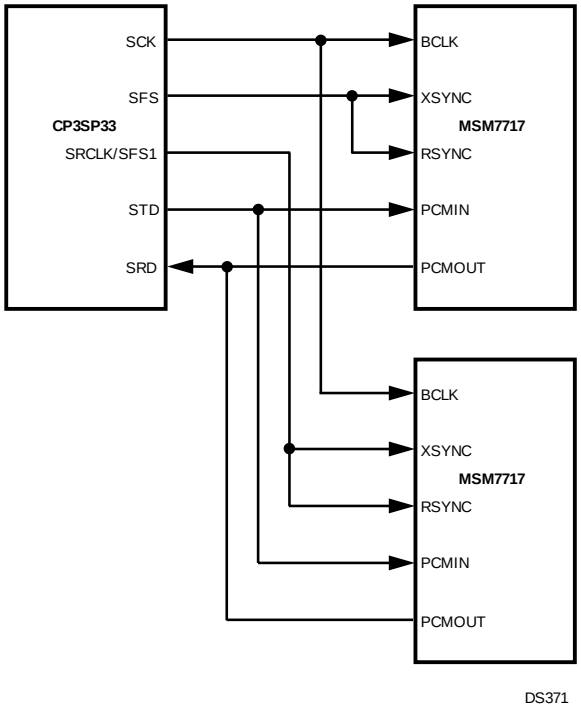


Figure 24-11. Interface to Two External Codecs

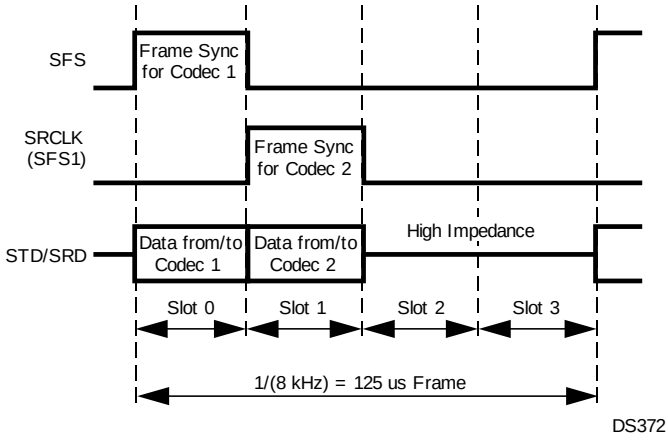


Figure 24-12. Codec Data on Two Slots

25 I²S Interface

The Inter-IC Sound (I^2S) bus is a popular 3-wire serial bus for interface to audio chips, such as codecs. This is a simple data interface, without any form of address or device selection. On an I^2S bus, there is only one bus master and one transmitter, but the master is not necessarily a transmitter or a receiver. The master may be a transmitter, a receiver, or a controller for data transfers between other devices acting as transmitter and receiver. In high-quality audio applications involving a codec, the codec is typically the master so that it has precise control over the I^2S bus clock.

The I^2S bus carries two channels, left and right, which are typically used to carry stereo audio data streams. The data alternates between left and right channels, as controlled by a word select signal driven by the bus master.

Four 8-level FIFOs are provided to buffer the receive and transmit directions of the two channels. Programmable thresholds are provided to trigger interrupt or DMA requests when the FIFOs require loading or unloading.

The I^{2}S module provides a three-pin unidirectional or four-pin bidirectional serial interface consisting of:

- *Serial Clock (I2SCLK)*—bit clock for serial data transfer. The transmitter drives the data on the falling edge. The receiver latches the data on the rising edge. The bus master drives this signal.
- *Word Select (I2SSWS)*—alternately selects between left and right channels, and defines the beginning and end of a word. The high and low phases must be equal length. The bus master drives this signal.
- *Serial Data (I2SSDI and I2SSDO)*—bit stream between the transmitter and receiver. The MSB is always sent first. In the CP3SP33 implementation, separate pins are provided for serial data input (I2SSDI) and serial data output (I2SSDO).

The I^{2}S module supports three common audio interface formats:

- *I^{2}S Format*—MSB transmitted one clock period after I2SSWS toggles. I2SSWS is low for the left channel and high for the right channel.
- *Left-Justified Format*—MSB transmitted following the clock edge when I2SSWS toggles. I2SSWS is high for the left channel and low for the right channel.
- *Right-Justified Format*—LSB transmitted during the clock period before I2SSWS toggles. I2SSWS is high for the left channel and low for the right channel.

The number of data bits is programmable from 8 to 24 bits per word, and the word length is programmable from 8 to 32 bits. The data is positioned in the word according to the format. In the left-justified and I^{2}S formats, the data starts with the MSB in the first or second clock periods, respectively. If the word length is less than the number of data bits, the trailing LSBs are truncated. In the right-justified format, the number of data bits and the word length are used to determine when to start sending or receiving the data bits, so it is critical to program these values correctly. In this format, the word length must be longer than the number of data bits, otherwise the transceiver will default to left-justified mode.

Figure 25-1 shows the timing of an I^{2}S format data transfer.

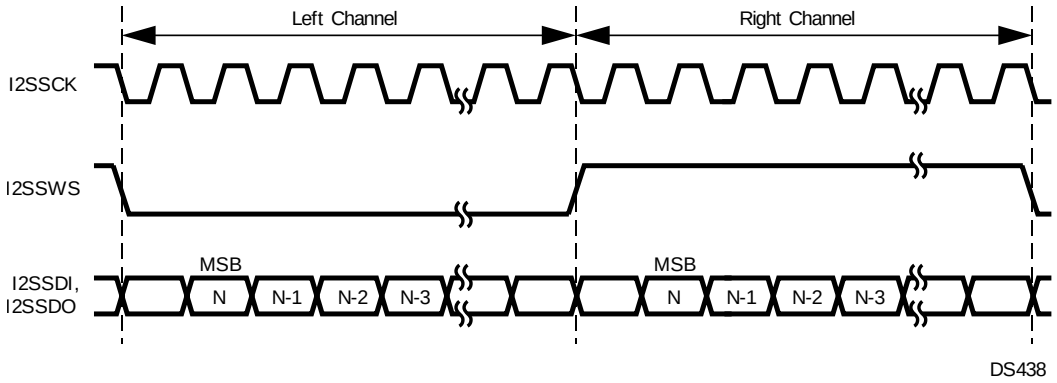


Figure 25-1. I^{2}S Format

Figure 25-2 shows the timing of a left-justified data transfer.

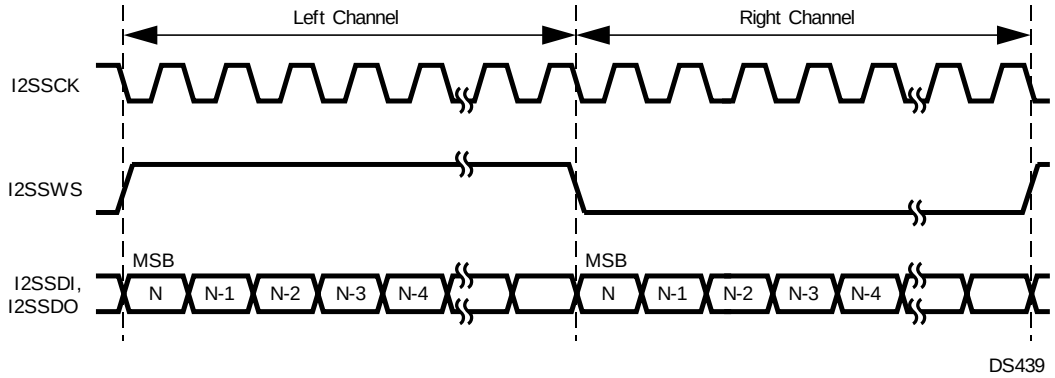


Figure 25-2. Left-Justified Format

Figure 25-3 shows the timing of a right-justified data transfer.

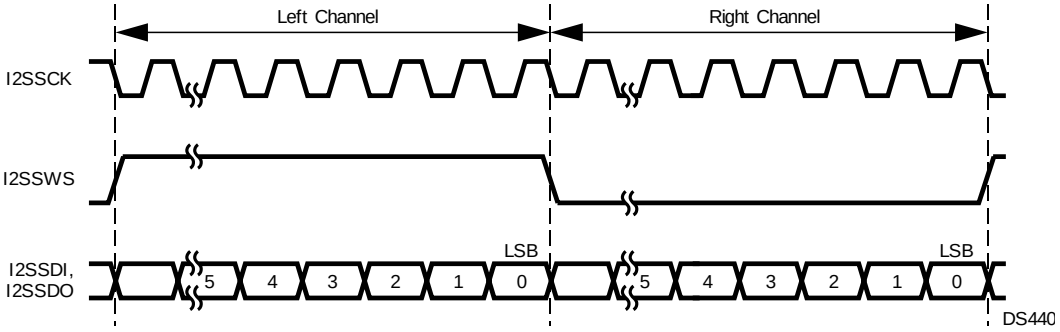


Figure 25-3. Right-Justified Format

25.1 Interrupts and DMA

Programmable thresholds are provided for the transmit and receive FIFOs to assert interrupt or DMA requests when the transmit FIFO can accept new data or the receive FIFO needs to be unloaded. Because the requests can be handled by either the CPU or DSP interrupt and DMA controllers, the requests are passed through the Audio Subsystem Controller (ASC). To use the interrupt or DMA requests, they must be enabled at three levels:

- *I^2S Module*—the RXLI and RXRI bits in the I2SRXCTL register must be set to enable the receiver interrupt requests for the left and right channels, respectively. The RXLD and RXRD bits enable the receiver DMA requests. The TXLI and TXRI bits in the I2STXCTL register enable the transmitter interrupt requests. The TXLD and TXRD bits enable the transmitter DMA requests.
- *ASC Module*—the I^2S interrupt is assigned to ASC interrupt channel. Bit 6 in the ASCINTSEL register must be clear to select CPU interrupt controller or set to select the DSP interrupt controller. The I^2S DMA requests are assigned to ASC DMA channels 14, 15, 16, and 17. For the receiver DMA requests, ASCDMASEL0 register bits 14 and 15 for the left and right channels, respectively, must be clear to select CPU DMA controller or set to select the DSP DMA controller. For the transmitter DMA requests, ASCDMASEL1 register bits 0 and 1 for the left and right channels, respectively, must be clear to select CPU DMA controller or set to select the DSP DMA controller.
- *Interrupt or DMA Controller*—the request must be enabled in the interrupt or DMA controller. For the CPU interrupt controller, this is interrupt request IRQ26. For the CPU DMA controller, DMA requests 14 and 15 are used by the receiver left and right channels, respectively, and DMA requests 16 and 17 are used by the transmitter left and right channels.

Error interrupts may be enabled for receive FIFO overrun and transmit FIFO underrun. A receive FIFO overrun occurs on writing to a full receive FIFO. A transmit FIFO underrun occurs on reading from an empty transmit FIFO. The RXER bit in the I2SRXCTL register enables receive error interrupts, and the TXER bit enables transmit error interrupts.

To avoid a transmit FIFO underrun, the TXFIFOENL and TXFIFOENR bits allow enabling the transmit FIFOs for loading before enabling the transmitter. However, as soon as the I2SCLK and I2SSWS signals are active, the right transmit FIFO is drained. In slave mode, these signals can be disabled in the GPIO registers until transmission is required. In master mode, the MS bit can be used to control the signals.

25.2 Data Alignment

Four alignment modes are available for the data registers, as shown in Figure 25-4. The RXALIGN field in the I2SRXCTL register specifies the receiver mode, and the TXALIGN field in the I2STXCTL register specifies the transmitter mode. If the receiver format has more bits than the mode, the least significant bits are truncated. If the transmitter format has more bits than the mode, the least significant bits are padded with the value in the LSBFILL bit.

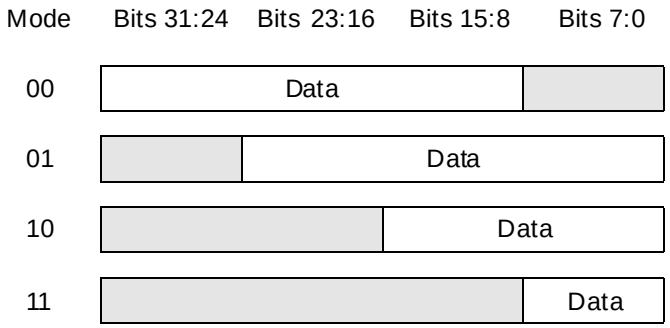


Figure 25-4. Data Register Alignment Modes

25.3 I²S Interface Registers

Table 25-1 lists the registers in the I²S interface.

Table 25-1. I²S Interface Registers

NAME	ADDRESS	DESCRIPTION
I2SCLK	FF 4000h	I²S Clock Register
I2SRXCTL	FF 4004h	I²S Receiver Control Register
I2STXCTL	FF 4008h	I²S Transmitter Control Register
I2SSTAT	FF 401Ch	I²S Status Register
I2SRXDATALEFT	FF 4014h	I²S Receiver Left Channel Data Register
I2SRXDATARIGHT	FF 4018h	I2S Receiver Right Channel Data Register
I2STXDATALEFT	FF 400Ch	I²S Transmitter Left Channel Data Register
I2STXDATARIGHT	FF 4010h	I²S Transmitter Right Channel Data Register

25.3.1 I²S Clock Register (I2SCLK)

The I2SCLK register is a 32-bit, read/write register that controls clock configuration, master/slave select, and word length. **Before enabling master mode (MS bit = 1), the CLKSEL, CLKDIV, and WSRES fields must be loaded.** These fields can only be loaded in slave mode (MS bit = 0), so after enabling master mode, the interface must be returned to slave mode to change any of their values. At reset, this register is initialized to 0007 0000h.

31 – 24	23 – 21	20 – 16	15 – 8	7 – 4	3	2 – 1	0
Reserved	Reserved	WSRES	CLKDIV	Reserved	MS	CLKSEL	CLKEN

CLKEN The Clock Enable bit enables clock to the module.
0 – Module disabled.
1 – Module enabled.

CLKSEL The Clock Select field selects the clock source for the module.
00 – PCLK Clock.
01 – Auxiliary clock 5.
10 – Reserved.
11 – Reserved

MS The Master Select bit selects whether the interface is operating in master or slave mode.
0 – Slave mode.
1 – Master mode.

CLKDIV The Clock Select field holds the divisor (expressed in half-periods of the clock source selected by CLKSEL) for generating the I^2S bus clock output I2SCLK in master mode. The field is biased by 1, so the clock source is divided by ((CLKDIV + 1) ÷ 2) to obtain I2SCLK. Zero is a reserved value for this field.

WSRES The Word Select Resolution field specifies the length of one phase (one-half period) of the word select output I2SSWS in master mode. The length is specified in terms of I^2S bus clock output I2SCLK periods. The field is biased by 1, so the default value of 7 results in a length of 8 clock periods. This field is only defined for values from 00111b to 11111b.

25.3.2 I^2S Receiver Control Register (I2SRXCTL)

The I2SRXCTL register is a 32-bit, read/write register that controls the I^2S receiver. At reset, this register is initialized to 0000 001Ch.

13 – 12	11	10	9	8	7	6 – 2	1 – 0
Reserved	RXLD	RXRD	RXLI	RXRI	RXER	RXRES	RXMOD

31 – 24	23 – 21	20	19	18 – 16	15 – 14
Reserved	Reserved	RXST	Reserved	RXFIFOTHRESH	RXALIGN

RXMOD The Receiver Mode field selects the format used by the receiver.
00 – Receiver disabled.
01 – I^2S format.
10 – Left-justified format.
11 – Right-justified format.

RXRES The Receiver Resolution field specifies the number of data bits to receive. In right-justified mode, this field also controls data alignment. It is particularly important to set RXRES correctly in right-justified mode, otherwise the MSB of the received serial data will not be aligned. For I^2S- and left-justified modes, data will always be correctly aligned (MSB in the correct position), however if RXRES specifies fewer data bits than are available on the bus, the LSBs will be truncated. The field is biased by 1, so the default value of 7 results in 8 bits. This field is only defined for values from 00111b to 10111b.

RXER The Receive Error bit enables the error interrupt from the receiver. A receive error occurs when the receiver attempts to load data into a full FIFO for the left or right channel. The status of the error condition is indicated by the RXERIRQ bit in the I2SSTAT register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RXRI — The Right Channel Receive Interrupt Enable bit enables an interrupt request when the number of empty words in the right receive FIFO is equal to or less than the receive FIFO threshold specified in the RXFIFOTHRES field. The status of the interrupt is indicated by the RXRIRQ bit in the I2SSTAT register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RXLI — The Left Channel Receive Interrupt Enable bit enables an interrupt request when the number of empty words in the left receive FIFO is equal to or less than the receive FIFO threshold specified in the RXFIFOTHRES field. The status of the interrupt is indicated by the RXLIRQ bit in the I2SSTAT register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RXRD — The Right Channel Receive DMA Enable bit enables a DMA request when the number of empty words in the right receive FIFO is equal to or less than the receive FIFO threshold specified in the RXFIFOTHRES field. If the interrupt and DMA requests are both enabled for this condition, only the DMA request is asserted.
0 – DMA disabled.
1 – DMA enabled.

RXLD — The Left Channel Receive DMA Enable bit enables a DMA request when the number of empty words in the left receive FIFO is equal to or less than the receive FIFO threshold specified in the RXFIFOTHRES field. If the interrupt and DMA requests are both enabled for this condition, only the DMA request is asserted.
0 – DMA disabled.
1 – DMA enabled.

RXALIGN — The Receiver Data Alignment field controls the position of the data loaded into the I2SRXDATALEFT and I2SRXDATARIGHT data registers, as described in Section 25.2.
00 – 24-bit data loaded in upper 3 bytes. Lowest byte cleared.
01 – 24-bit data loaded in lower 3 bytes.
10 – 16-bit data loaded into lower 2 bytes.
11 – 8-bit data loaded into lowest byte.

RXFIFOTHRESH — The Receive FIFO Threshold specifies the number of empty words in the receive FIFOs before an interrupt or DMA request is asserted.

RXST — The Receiver Status bit controls whether the I2SSTAT.RXSTATUSR and I2SSTAT.RXSTATUSL fields indicate the number of empty words in the receive FIFOs rather than the number of filled words. This field does not affect the interpretation of the RXFIFOTHRES field or the generation of interrupt and DMA requests.
0 – Status fields indicate the number of filled words in the FIFOs.
1 – Status fields indicate the number of empty words in the FIFOs.

25.3.3 I^2S Transmitter Control Register (I2STXCTL)

The I2STXCTL register is a 32-bit, read/write register that controls the I^2S transmitter. At reset, this register is initialized to 0000 001Ch.

13	12	11	10	9	8	7	6 2	1 0
TXFIFOENL	TXFIFOENR	TXLD	TXRD	TXLI	TXRI	TXER	TXRES	TXMOD

31 24	23	22	21	20	19	18 16	15 14
Reserved	Reserved	LSBFILL	FLUSH	TXST	Reserved	TXFIFOTHRESH	TXALIGN

TXMOD	The Transmitter Mode field selects the format used by the transmitter. 00 – Transmitter disabled. 01 – I^2S format. 10 – Left-justified format. 11 – Right-justified format.
TXRES	The Transmitter Resolution field specifies the number of data bits when the right-justified format is used. In right-justified mode, TXRES must specify fewer bits than the word length, otherwise the transceiver will default to left-justified mode. In I2S- and left-justified modes, if TXRES specifies more data bits than available in the word, the LSBs are truncated. The field is biased by 1, so the default value of 7 results in 8 bits. This field is only defined for values from 00111b to 10111b. This field is not defined for reading
TXER	The Transmit Error bit enables the error interrupt from the transmitter. A transmit error occurs when the transmitter attempts to unload data from an empty FIFO for the left or right channel. The status of the error condition is indicated by the TXERIRQ bit in the I2SSTAT register. 0 – Interrupt disabled. 1 – Interrupt enabled.
TXRI	The Right Channel Transmit Interrupt Enable bit enables an interrupt request when the number of filled words in the right transmit FIFO is equal to or less than the transmit FIFO threshold specified in the TXFIFOTHRES field. The status of the interrupt is indicated by the TXRIRQ bit in the I2SSTAT register. 0 – Interrupt disabled. 1 – Interrupt enabled.
TXLI	The Left Channel Transmit Interrupt Enable bit enables an interrupt request when the number of filled words in the left transmit FIFO is equal to or less than the transmit FIFO threshold specified in the TXFIFOTHRES field. The status of the interrupt is indicated by the TXLIRQ bit in the I2SSTAT register. 0 – Interrupt disabled. 1 – Interrupt enabled.
TXRD	The Right Channel Transmit DMA Enable bit enables a DMA request when the number of filled words in the right transmit FIFO is equal to or less than the transmit FIFO threshold specified in the TXFIFOTHRES field. If the interrupt and DMA requests are both enabled for this condition, only the DMA request is asserted. 0 – DMA disabled. 1 – DMA enabled.
TXLD	The Left Channel Transmit DMA Enable bit enables a DMA request when the number of filled words in the left transmit FIFO is equal to or less than the transmit FIFO threshold specified in the TXFIFOTHRES field. If the interrupt and DMA requests are both enabled for this condition, only the DMA request is asserted. 0 – DMA disabled. 1 – DMA enabled.
TXFIFOENR	The Left Channel Transmit DMA Enable bit enables a DMA request when the number of filled words in the left transmit FIFO is equal to or less than the transmit FIFO threshold specified in the TXFIFOTHRES field. If the interrupt and DMA requests are both enabled for this condition, only the DMA request is asserted. 0 – DMA disabled. 1 – DMA enabled.
TXFIFOENL	The Transmit FIFO Enable Right Channel bit enables the right channel transmit FIFO. The FIFO must be enabled before the transmitter, so it can be filled in advance of transmission. 0 – FIFO disabled. 1 – FIFO enabled.

TXALIGN The Transmitter Data Alignment field controls the position of the data read from the I2STXDATALEFT and I2STXDATARIGHT data registers, as described in Section 25.2.
00 – 24-bit data read from upper 3 bytes.
01 – 24-bit data read from lower 3 bytes.
10 – 16-bit data read from lower 2 bytes.
11 – 8-bit data read from lowest byte.

TXFIFOTHRESH The Transmit FIFO Threshold specifies the number of filled words in the transmit FIFOs before an interrupt or DMA request is asserted.

TXST The Transmitter Status bit controls whether the I2SSTAT.TXSTATUSR and I2SSTAT.TXSTATUSL fields indicate the number of filled words in the transmit FIFOs rather than the number of empty words. This field does not affect the interpretation of the TXFIFOTHRES field or the generation of interrupt and DMA requests.
0 – Status fields indicate the number of empty words in the FIFOs.
1 – Status fields indicate the number of filled words in the FIFOs.

FLUSH The Flush bit clears the transmit FIFOs. Software must clear the bit after flushing the FIFOs to enable normal operation.
0 – Normal operation.
1 – Transmit FIFOs are cleared.

LSBFILL The LSB Fill bit specifies the value of trailing bits when the number of data bits is less than the number of bits in the data format.

25.3.4 I^2S Status Register (I2SSTAT)

The I2SSTAT register is a 32-bit, read/write register that indicates the status of I^2S module and clears interrupt requests. The TXRIRQ, TXLIRQ, TXERIRQ, RXRIRQ, RXLIRQ, and RXERIRQ bits are sticky bits that remain set until the condition which sets the bit has been removed and software has cleared the bit. The state of a condition is always visible in the I2SSTAT register, but it will not assert an interrupt request unless it is enabled in the corresponding control register (either I2SRXCTL or I2STXCTL). At reset, this register is initialized to 0000 0808h.

15	14	13	12 – 8	7	6	5	4 – 0
RXERIRQ	RXLIRQ	RXRIRQ	TXSTATUSL	TXERIRQ	TXLIRQ	TXRIRQ	TXSTATUSR

31	30 – 29	28 – 24	23 – 21	20 – 16
Reserved	WSSTATUS4:3	RXSTATUSL	WSSTATUS2:0	RXSTATUSR

TXSTATUSR The Transmit FIFO Status Right Channel field indicates the current fill level of the right channel transmit FIFO. This field is not sticky, so an underrun condition may disappear before software reads this field. Use the TXERIRQ bit to determine whether a FIFO underrun has occurred. The TXST bit in the I2STXCTL register controls the interpretation of this field.

TXSTATUSR FIELD	TXST BIT	DESCRIPTION
1Fh	X	Overrun error (written while full)
1Eh	X	Underrun error (read while empty)
00h to 08h	0	0 to 8 empty words
00h to 08h	1	0 to 8 filled words

TXRIRQ The Transmit Right Channel Interrupt Request Pending bit is set when the I2STXCTL.TXRI bit is set and the right channel transmit FIFO fill level falls below the threshold specified in the I2STXCTL.TXFIFOTHRESH field. Once set, the TXRIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

TXLIRQ The Transmit Left Channel Interrupt Request Pending bit is set when the I2STXCTL.TXLI bit is set and the left channel transmit FIFO fill level falls below the threshold specified in the I2STXCTL.TXFIFOTHRESH field. Once set, the TXLIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

TXERIRQ The Transmit Error Interrupt Request Pending bit is set when the I2STXCTL.TXER bit is set and either transmit FIFO has been underrun. Once set, the TXERIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

TXSTATUSL The Transmit FIFO Status Left Channel field indicates the current fill level of the left channel transmit FIFO. This field is not sticky, so an underrun condition may disappear before software reads this field. Use the TXERIRQ bit to determine whether a FIFO underrun has occurred. The TXST bit in the I2STXCTL register controls the interpretation of this field.

TXSTATUSL FIELD	TXST BIT	DESCRIPTION
1Fh	X	Overrun error (written while full)
1Eh	X	Underrun error (read while empty)
00h to 08h	0	0 to 8 empty words
00h to 08h	1	0 to 8 filled words

RXRIRQ The Receive Right Channel Interrupt Request Pending bit is set when the I2SRXCTL.RXRI bit is set and the right channel receive FIFO fill level rises above the threshold specified in the I2SRXCTL.RXFIFOTHRESH field. Once set, the RXRIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

RXLIRQ The Receive Left Channel Interrupt Request Pending bit is set when the I2SRXCTL.RXLI bit is set and the left channel receive FIFO fill level rises above the threshold specified in the I2SRXCTL.RXFIFOTHRESH field. Once set, the RXLIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

RXERIRQ The Receive Error Interrupt Request Pending bit is set when the I2SRXCTL.RXER bit is set and either receive FIFO has been overrun. Once set, the RXERIRQ bit remains set until cleared by writing 1 to the bit.
0 – Interrupt request deasserted.
1 – Interrupt request asserted.

RXSTATUSR The Receive FIFO Status Right Channel field indicates the current fill level of the right channel receive FIFO. This field is not sticky, so an overrun condition may disappear before software reads this field. Use the RXERIRQ bit to determine whether a FIFO overrun has occurred. The RXST bit in the I2SRXCTL register controls the interpretation of this field.

RXSTATUSR FIELD	RXST BIT	DESCRIPTION
1Fh	X	Overrun error (written while full)
1Eh	X	Underrun error (read while empty)
00h to 08h	0	0 to 8 empty words
00h to 08h	1	0 to 8 filled words

RXSTATUSL The Receive FIFO Status Left Channel field indicates the current fill level of the left channel receive FIFO. This field is not sticky, so an overrun condition may disappear before software reads this field. Use the RXERIRQ bit to determine whether a FIFO overrun has occurred. The RXST bit in the I2SRXCTL register controls the interpretation of this field.

RXSTATUSL FIELD	RXST BIT	DESCRIPTION
1Fh	X	Overrun error (written while full)
1Eh	X	Underrun error (read while empty)
00h to 08h	0	0 to 8 empty words
00h to 08h	1	0 to 8 filled words

WSSTATUS The Word Select Status field indicates the length of one phase of the word select signal I2SSWS in terms of the number of periods of the I^2S bus clock I2SCLK. This is useful in slave mode, so that software can determine the number of bits being sent by the master.

25.3.5 I^2S Receiver Left Channel Data Register (I2SRXDATALEFT)

The I2SRXDATALEFT register is a 32-bit, read-only register used to unload the left channel receive FIFO. At reset, this register is initialized to 0000 000h.

31	0
Reserved	

25.3.6 I^2S Receiver Right Channel Data Register (I2SRXDATARIGHT)

The I2SRXDATARIGHT register is a 32-bit, read-only register used to unload the right channel receive FIFO. At reset, this register is initialized to 0000 000h.

31	0
Reserved	

25.3.7 I^2S Transmitter Left Channel Data Register (I2STXDATALEFT)

The I2STXDATALEFT register is a 32-bit, write-only register used to load the left channel transmit FIFO. At reset, this register is initialized to 0000 000h.

31	0
Reserved	

25.3.8 I^2S Transmitter Right Channel Data Register (I2STXDATARIGHT)

The I2STXDATARIGHT register is a 32-bit, write-only register used to load the right channel transmit FIFO. At reset, this register is initialized to 0000 000h.

31	0
Reserved	

26 Dual CVSD/PCM Conversion Modules

The CVSD/PCM modules perform conversion between CVSD data and PCM data, in which the CVSD encoding is that used in Bluetooth communication and the PCM encoding may be 8-bit µ-Law, 8-bit A-Law, or 13-bit to 16-bit linear.

A CVSD module can operate in either of two modes:

- *Fixed-rate mode*—125 µs (8 kHz) per PCM sample, based on a 2 MHz module clock frequency. Intended for exchanging data with an external codec.
- *Free-running mode*—runs at any module clock frequency up to the PCLK Clock frequency. CVSD transcoder activity is governed by DMA acknowledgment on CVSDIN and CVSDOUT channels. Intended for exchanging PCM data over a Bluetooth channel.

On the CVSD side, read and write FIFOs buffer up to 8 words of data. On the PCM side, double-buffered registers are provided. The intended use is to move CVSD data with an interrupt handler, and to move PCM data with DMA. Figure 26-1 is a block diagram of a CVSD/PCM converter module.

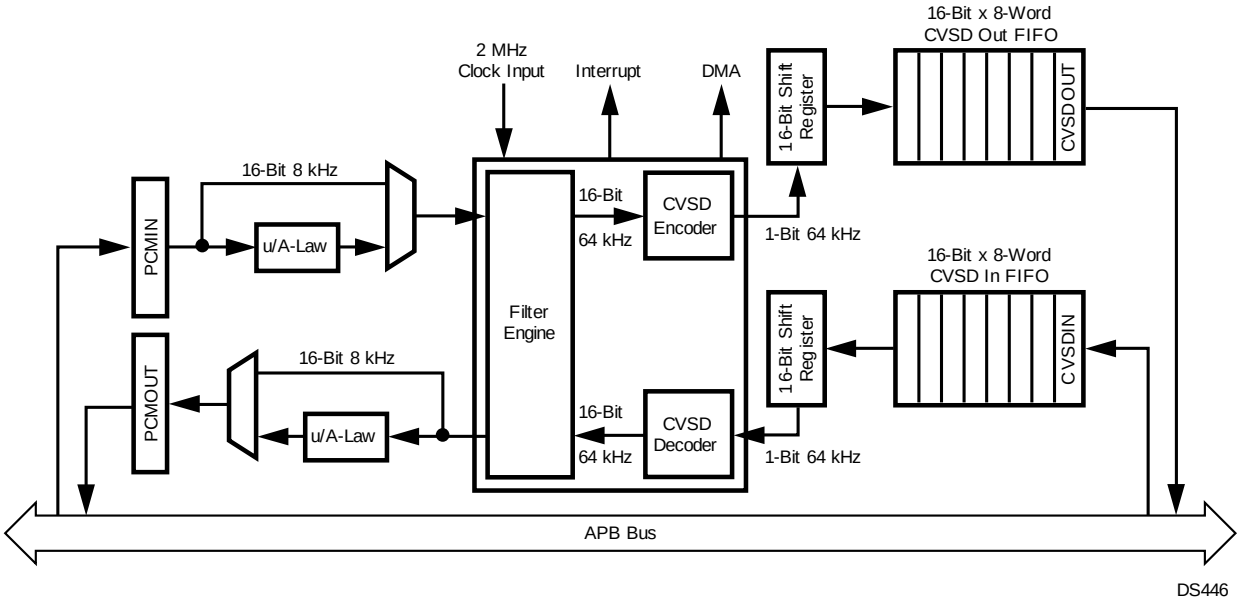


Figure 26-1. CVSD/PCM Converter Block Diagram

26.1 Operation

In fixed-rate mode, the module converts between PCM data and CVSD data at a fixed PCM data rate of 8 kHz. Due to compression, the data rate on the CVSD side is only 4 kHz.

If PCM interrupts are enabled, every 125 µs (8 kHz) an interrupt will occur and the interrupt handler can operate on some or all of the four audio streams: CVSD in, CVSD out, PCM in, and PCM out. Alternatively, a DMA request is issued every 125 µs and the DMA controller is used to move the PCM data between the CVSD/PCM module and the audio interface.

If CVSD interrupts are enabled, an interrupt is issued when either one of the CVSD FIFOs is almost empty or almost full. On the PCM data side there is double buffering, and on the CVSD side there is an eight word (8-bit × 16-bit) FIFO for the read and write paths.

Inside the module, a filter engine receives the 8-kHz stream of 16-bit samples and interpolates to generate a 64-kHz stream of 16-bit samples. This goes into a CVSD encoder which converts the data into a single-bit delta stream using the CVSD parameters as defined by the Bluetooth specification. There is a similar path that reverses this process, converting the CVSD 64-kHz bit stream into a 64-kHz 16-bit data stream. The filter engine then decimates this stream into an 8-kHz, 16-bit data stream.

26.2 PCM Conversions

During conversion between CVSD and PCM, any PCM format changes are done automatically depending on whether the PCM data is µ-Law, A-Law, or linear. In addition to this, a separate function can be used to convert between the various PCM formats as required. Conversion is performed by setting up the control bit CVCTRLn.PCMCONV to define the conversion and then writing to the LOGINn and LINEARINn registers and reading from the LOGOUTn and LINEAROUTn registers. There is no delay in the conversion operation, and it does not have to operate at a fixed rate. It will only convert between µ-Law/A-Law and linear, not directly between µ-Law and A-Law. (This could easily be performed by converting between µ-Law and linear and between linear and A-Law.)

If a conversion is performed between linear and µ-Law log PCM data, the linear PCM data are treated in the left-aligned 14-bit linear data format with the two LSBs unused. If a conversion is performed between linear and A-Law log PCM data, the linear PCM data are treated in the left-aligned 13-bit linear data format with the three LSBs unused.

If the module is only used for PCM conversions, the CVSD clock can be disabled by clearing the CVSD Clock Enable bit (CLKEN) in the control register.

26.3 CVSD Conversion

The CVSD/PCM converter module transforms either 8-bit logarithmic or 13-bit to 16-bit linear PCM samples at a fixed rate of 8 ksps. The CVSD to PCM conversion format must be specified by the CVSDCONV control bits in the CVSD Control register (CVCTRLn).

The CVSD algorithm is designed for 2's complement 16-bit data and is tuned for best performance with typical voice data. Mild distortion will occur for peak signals greater than -6 dB. The Bluetooth CVSD standard is designed for best performance with typical voice signals: nominally –6 dB with occasional peaks to 0 dB rather than full-scale inputs. Distortion of signals greater than –6 dB is not considered detrimental to subjective quality tests for voice-band applications and allows for greater clarity for signals below –6 dB. The gain of the input device should be tuned with this in mind.

If required, the RESOLUTION field of the CVCTRLn register can be used to optimize the level of the 16-bit linear input data by providing attenuations (right-shifts with sign extension) of 1, 2, or 3 bits.

Log data is always 8 bit, but to perform the CVSD conversion, the log data is first converted to 16-bit 2's complement linear data. A-law and µ-law conversion can also slightly affect the optimum gain of the input data. The CVCTRL.RESOLUTION field can be used to attenuate the data if required.

If the resolution is not set properly, the audio signal may be clipped or attenuated.

26.4 Fixed-Rate PCM-to-CVSD Conversion

The converter core reads the double-buffered PCMIN register every 125 µs and writes a new 16-bit CVSD data stream into the CVSD Out FIFO every 250 µs. If the PCMIN buffer has not been updated with a new PCM sample between two reads from the CVSD core, the old PCM data is used again to maintain a fixed conversion rate. Once a new 16-bit CVSD data stream has been calculated, it is copied into the 8 × 16-bit wide CVSD Out FIFO.

If there are only three empty words (16-bit) left in the FIFO, the nearly full bit (CVNF) is set, and, if enabled (CVSDINT = 1), an interrupt request is asserted.

If the CVSD Out FIFO is full, the full bit (CVF) is set, and, if enabled (CVSDERRINT = 1), an interrupt request is asserted. In this case, the CVSD Out FIFO remains unchanged.

Within the interrupt handler, the CPU can read out the new CVSD data. If the CPU reads from an already empty CVSD Out FIFO, a lockup of the FIFO logic may occur which persists until the next reset. Software must check the CVOUTST field of the CVSTAT register to read the number of valid words in the FIFO. Software must not use the CVNF bit as an indication of the number of valid words in the FIFO.

26.5 Fixed-Rate CVSD-to-PCM Conversion

The converter core reads from the CVSD In FIFO every 250 µs and writes a new PCM sample into the PCMOUT buffer every 125 µs. If the previous PCM data has not yet been transferred to the audio interface, it will be overwritten with the new PCM sample.

If there are only three unread words left, the CVSD In Nearly Empty bit (CVNE) is set and, if enabled (CVSDINT = 1), an interrupt request is generated.

If the CVSD In FIFO is empty, the CVSD In Empty bit (CVE) is set and, if enabled (CVSDERRINT = 1), an interrupt request is generated. If the converter core reads from an already empty CVSD In FIFO, the FIFO automatically returns a checkerboard pattern to specify a minimum level of distortion of the audio stream.

26.6 Free-Running Mode

The free-running mode of the CVSD/PCM converter is enabled by setting the CVFR, DMACI, and DMACO bits of the CVSD Control Register (CVCTRL). In this mode, the CVSD/ PCM converter is expected to be serviced by DMA operations on both its PCM and CVSD input and output channels. In contrast to the fixed-rate operating mode, the clock frequency for free-running operation is not restricted to 2 MHz. In free-running mode, activity within the CVSD/PCM transcoder is governed by the acknowledgment of DMA requests on the CVSDIN and CVSDOUT channels. If a DMA request on either of these channels goes unacknowledged then the CVSD/PCM module will stall its internal operation just before the start of the next transcoding cycle. When the outstanding DMA requests are eventually acknowledged then the CVSD/PCM module will resume its operation in a seamless fashion.

26.7 Interrupt Generation

An interrupt is asserted in any of the following cases:

- When a new PCM sample has been written into the PCMOUT register and the CVCTRLn.PCMINT bit is set.
- When a new PCM sample has been read from the PCMIN register and the CVCTRLn.PCMINT bit is set.
- When the CVSD In FIFO is nearly empty (CVSTATn.CVNE = 1) and the CVCTRLn.CVSDINT bit is set.
- When the CVSD Out FIFO is nearly full (CVSTATn.CVNF = 1) and the CVCTRLn.CVSDINT bit is set.
- When the CVSD In FIFO is empty (CVSTATn.CVE = 1) and the CVCTRLn.CVSDERRINT bit is set.
- When the CVSD Out FIFO is full (CVSTATn.CVF = 1) and the CVCTRLn.CVSDERRINT bit is set

Both the CVSD In and CVSD Out FIFOs have a size of 8 bit × 16 bit (8 words). The warning limits for the two FIFOs are set at 5 words. (The CVSD In FIFO interrupt will occur when there are 3 words left in the FIFO, and the CVSD Out FIFO interrupt will occur when there are 3 or less empty words left in the FIFO.) The limit is set to 5 words because Bluetooth audio data is transferred in packages of 10 bytes or multiples of 10 bytes.

Because the PCM I/O interrupt requests can be handled by either the CPU or DSP interrupt controllers, the interrupt requests are passed through the Audio Subsystem Controller (ASC), which is programmed to route them to the interrupt controller. To use the PCM I/O interrupts, they must be enabled at three levels:

- *CVSD/PCM module*—the interrupts must be enabled using the PCMINT bits in the CVCTRLn registers.
- *ASC module*—the PCM I/O interrupts are assigned to ASC interrupt channels 5 (CVSD/PCM module 0) and 4 (CVSD/PCM module 1), so bits 5 and 4 in the ASCINTSEL register must be clear to select the CPU interrupt controller or set to select the DSP interrupt controller.
- *Interrupt controller*—the interrupt requests must be enabled in the interrupt controller. For the CPU interrupt controller, these are interrupt requests IRQ25 (CVSD/ PCM module 0) and IRQ24 (CVSD/PCM module 1).

26.8 DMA Support

Each CVSD/PCM converter has four DMA channels for processor independent operation. Both receive and transmit for CVSD data and PCM data can be enabled individually. A CVSD/PCM module asserts a DMA request to the on-chip DMA controller under the following conditions:

- The DMAPO bit is set and the PCMOUTn register is full, because it has been updated by the converter core with a new PCM sample. (The DMA controller can read out one PCM data word from the PCMOUTn register.)
- The DMAPI bit is set and the PCMINn register is empty, because it has been read by the converter core. (The DMA controller can write one new PCM data word into the PCMINn register.)
- The DMACO bit is set and a new 16-bit CVSD data stream has been copied into the CVSD Out FIFO. (The DMA controller can read out one 16-bit CVSD data word from the CVSD Out FIFO.)
- The DMACI bit is set and a 16-bit CVSD data stream has been read from the CVSD In FIFO. (The DMA controller can write one new 16-bit CVSD data word into the CVSD In FIFO.)

The DMA controller only supports indirect DMA transfers. Therefore, transferring data between a CVSD/PCM module and another on-chip module requires two bus cycles.

The trigger for DMA may also trigger an interrupt if the corresponding enable bits in the CVCTRLn register is set. Therefore, care must be taken when setting the desired interrupt and DMA enable bits. The following conditions must be avoided:

- Setting the PCMINT bit and either of the DMAPO or DMAPI bits.
- Setting the CVSDINT bit and either of the DMACO or DMACI bits.

Because the PCM I/O DMA requests can be handled by either the CPU or DSP DMA controllers, the DMA requests are passed through the Audio Subsystem Controller (ASC), which is programmed to route them to the DMA controller. To use the PCM I/O DMA requests, they must be enabled at three levels:

- *CVSD/PCM module*—the PCM I/O DMA requests must be enabled using the DMAPI and DMAPO bits in the CVCTRLn registers.
- *ASC module*—the PCM I/O DMA requests are assigned to ASC DMA channels 3 (PCMIN0), 2 (PCMOUT0), 5 (PCMIN1), and 4 (PCMOUT1). These channels are controlled by bits 3, 2, 5, and 4, respectively, in the ASCDMASEL0 register. The bits must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be enabled in the ASCDDMASELn registers.
- *DMA controller*—the DMA requests must be enabled in the DMA controller. For the CPU DMA controller, these are DMA requests 3 (PCMIN0), 2 (PCMOUT0), 5 (PCMIN1), and 4 (PCMOUT1).

The CVSD I/O DMA requests are only provided to the CPU DMA controller.

26.9 CVSD/PCM Audio Data Flow

CVSD and PCM data can be transferred to and from the CVSD/PCM modules using DMA channels. Alternatively, CVSD data can be transferred to and from the CVSD/PCM modules by software directly accessing the FIFOs, through the CVSDINn and CVSDOUTn registers.

The CVSDOUT and CVSDIN DMA channels are linked to the operation of the CVSD/PCM module internal FIFOs. Each time the CVSDOUT FIFO reaches its almost full level of 5 entries, a DMA request is generated. Initially when the CVSDIN FIFO contains 3 or fewer entries, a single DMA transfer is requested. Further DMA requests are generated each time the CVSD In FIFO is read by the CVSD/PCM converter.

Three of the four AAI channels are supported by independent DMA channels for both the receive and transmit data paths. Figure 26-2 shows the flow of audio data when both CVSD/PCM modules are used to support data exchange between the Bluetooth LLC and two external codec channels accessed through the AAI.

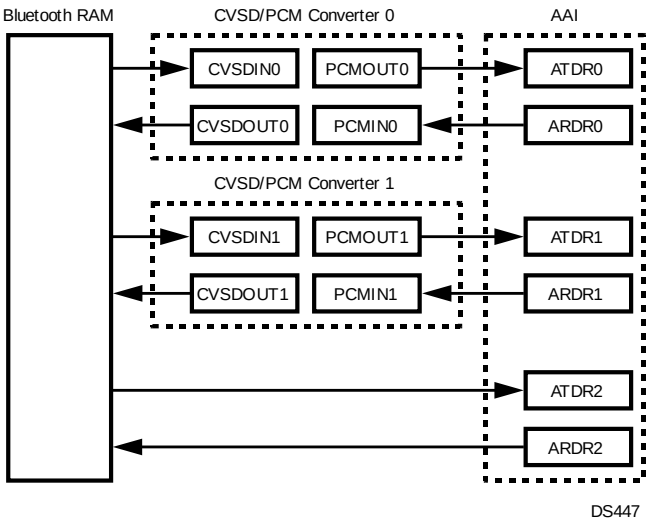


Figure 26-2. CVSD/PCM To/From AAI Data Flow

The CVSD/PCM modules can operate in two basic modes: fixed-rate mode and free-run mode. In the fixed-rate mode, the clock frequency supplied to CVSD/PCM module 0 on Auxiliary Clock 2 (or Auxiliary Clock 3 for CVSD/PCM module 1) must be exactly 2 MHz. In the fixed-rate mode, the CVSD/PCM module will expect to receive 8000 PCM samples per second on its PCMIN input, and it will produce 4000 words per second on the CVSDOUT output. Each word holds 16 bits of the CVSD-encoded bit stream. It will also expect to receive 16-bit CVSD-encoded data words at a rate of 4000 per second on its CVSDIN input and will produce PCM samples at a rate of 8000 per second on its PCMOUT output.

In the free-run mode the Auxiliary Clocks can be set to any frequency up to a maximum equal to the PCLK Clock frequency. In this mode, the CVSD/PCM module will request a new PCM sample every 250 cycles of the Auxiliary Clock, and it will produce a new CVSD data word every 500 cycles of the Auxiliary Clock.

The following examples illustrate the flow of CVSD data in typical applications.

26.9.1 Dual CVSD Channels to AAI

This model is illustrated in Figure 26-3 and has the following characteristics.

- Two bidirectional CVSD channels through the Bluetooth LLC.
- PCM data received by the AAI.
- CVSD/PCM conversion done on-chip.
- PCM audio data is transferred between the AAI and the CVSD/PCM module by means of DMA.
- CVSD audio data is transferred between the CVSD/PCM module and the Bluetooth RAM by means of DMA.
- CVSD/PCM modules operate in fixed-rate mode using a 2 MHz clock supplied on Auxiliary Clocks 2 and 3.

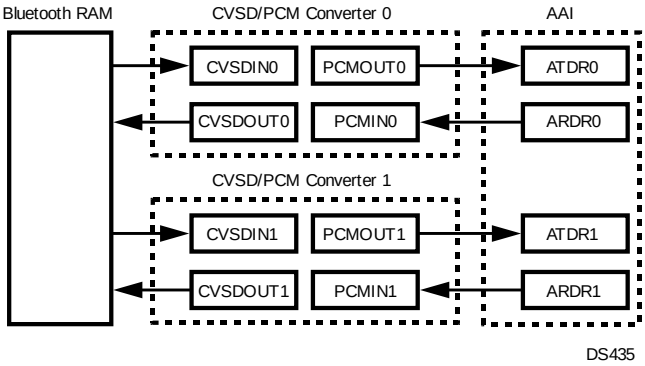


Figure 26-3. Dual CVSD Channels to AAI

26.9.1.1 PCM Transmit Data Path

The two AAI receive registers, ARDR0 and ARDR1, receive codec data for transmission over a Bluetooth link. These registers are assigned to separate DMA channels. The DMA data transfers from the AAI to the CVSD/PCM modules use indirect transfers, in which each transfer consists of two adjacent bus cycles (read/write). The operation of the ARDR0 DMA channel is described below. The ARDR1 DMA channel operates in a similar way.

The AAI requests a DMA transfer when ARDR0 is full. If ARDR0 is assigned to Device A and the DIR bit in the DMA channel control register is clear, the DMA controller reads data from ARDR0 (device A) and writes the data into the PCMIN register of CVSD0 (device B). Automatic address update (increment/decrement) must be disabled. With an audio sample rate of 8 ksps, ARDR0 will request a DMA transfer every 125 ms (every 1500 CPU clock cycles at a 12 MHz PCLK Clock rate).

26.9.1.2 PCM Receive Data Path

The two AAI transmit registers, ATDR0 and ATDR1, are assigned to separate DMA channels. The DMA data transfers from the CVSD/PCM modules to the audio interface use indirect transfers, in which each transfer consists of two adjacent bus cycles (read/write). The operation of the ATDR0 DMA channel is described below. The ARDR1 DMA channel operates in a similar way.

The AAI requests a DMA transfer when ATDR0 is empty. If ATDR0 is assigned to Device A and the DIR bit in the DMA channel control register is set, then the DMA controller reads data from the PCMOUT register of CVSD0 (device B) and writes the data into ATDR0 (device A). Automatic address update (increment/decrement) must be disabled.

With an audio sample rate of 8 ksps, ATDR0 will request a DMA transfer every 125 ms (every 1500 CPU clock cycles at a 12 MHz PCLK Clock rate).

26.9.1.3 CVSD Transmit/Receive Paths

The CVSD/PCM modules operate in their fixed-rate mode with DMA channels assigned to their CVSDINn and CVSDOUTn registers. The activation of these DMA channels is linked to the operation of the CVSDINn and CVSDOUTn FIFOs. The operation of the CVSD/PCM module 0 DMA channels is described below. The CVSD/PCM module 1 DMA channels operate in a similar way.

Initially when the CVSDIN0 FIFO contains 3 or fewer entries, a single DMA transfer is requested. Further DMA requests are generated each time the CVSDIN0 FIFO is read by the CVSD/PCM converter. If CVSDIN0 is assigned to device A and the DIR bit in the DMA channel control register is set, the DMA controller reads data from the Bluetooth RAM (device B) and writes the data into CVSDIN0 (device A). The DMA channel must be configured so that automatic update of the device A (CVSDIN0) address is disabled. The address of device B (Bluetooth RAM) must be configured to automatically increment or decrement by 2 after each DMA transfer.

When the CVSDOUT0 contains 5 entries or more, a DMA transfer is requested. If CVSDOUT0 is assigned to device A and the DIR bit in the DMA channel control register is clear, the DMA controller reads data from the CVSDOUT0 (device A) and writes the data into the Bluetooth RAM (device B).

The CVSD/PCM module introduces an effective compression ratio of 2 to 1, so an audio sample rate of 8 ksps results in DMA transfers on both the CVSDIN and CVSDOUT DMA channels at a rate of 4k transfers per second.

26.9.2 Dual CVSD Channels to Bluetooth HCI

This usage model is illustrated in Figure 26-4 and consists of the following operations.

- Two bidirectional CVSD channels through the Bluetooth LLC.
- CVSD/PCM conversion done on-chip.
- PCM audio data is transferred between the CVSD/PCM modules and the system RAM by DMA.
- CVSD audio data is transferred between the CVSD/PCM modules and the Bluetooth RAM by DMA.
- CVSD/PCM modules operate in free-running mode, in which the sample throughput is determined by the frequency of Auxiliary Clocks 2 and 3.

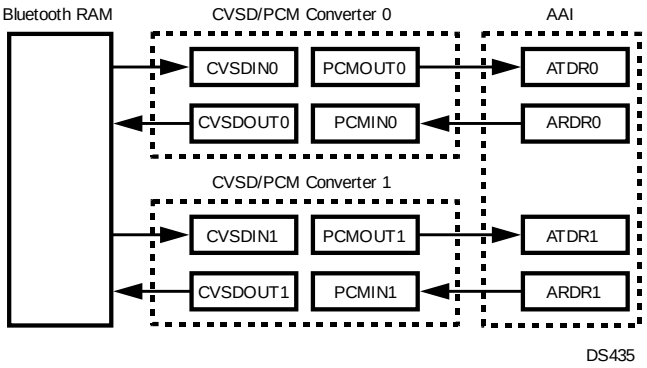


Figure 26-4. Dual CVSD Channels to Bluetooth HCI

26.9.2.1 PCM Transmit Data Path

The CVSD/PCM modules have DMA channels assigned to their PCMINn registers. DMA data transfers from system RAM to the CVSD/PCM modules use indirect transfers, in which each transfer is performed with two adjacent bus cycles (read/write). The operation of the PCMIN0 DMA channel of CVSD/PCM module 0 is described below. The PCMIN1 DMA channel operates in a similar way.

The CVSD/PCM module requests a DMA transfer each time a PCM sample has been processed. If PCMIN0 is assigned to device A and the DIR bit in the DMA channel control register is set, the DMA controller reads data from system RAM (device B) and writes the data into the PCMIN0 register (device A). The DMA controller must be configured so that automatic update of the device A (PCMIN0) address is disabled. The address of device B (system RAM) must be configured to automatically increment or decrement by 2 after each DMA transfer.

CVSD/PCM module 0 requires 250 clock cycles of Auxiliary Clock 2 (Auxiliary Clock 3 for CVSD/PCM module 1) to encode a single PCM sample. The effective compression ratio of 2 to 1 from PCM to CVSD means that a new 16-bit CVSD-encoded data word is produced every 500 cycles. Every 250 cycles, a DMA operation is requested to transfer a PCM sample for the next cycle of the encoder. Similarly, every 500 cycles, a DMA operation is requested to transfer a CVSD data word from the CVSD/PCM module to memory. The frequency of the Auxiliary Clock determines the time interval between successive DMA requests.

For example, if the Auxiliary Clock is 2 MHz, the CVSD/PCM module will request a DMA transfer on the PCMINn channel at a rate of 8 ksps. If the Auxiliary Clock is 12 MHz, the DMA transfer rate will be 48 ksps.

The operation of the CVSD/PCM module (i.e., PCMINn to CVSDOUTn) is controlled by DMA request/acknowledge activity on the CVSDOUTn and CVSDINn DMA channels. The number of DMA transfers required to fill the CVSD buffer resident in Bluetooth RAM must be programmed into the DMA controllers block length register. When the required number of DMA transfers has been executed, the DMA controller will not acknowledge further DMA requests on the CVSDOUTn channel. Similarly, the DMA requests on the CVSDINn channel are no longer acknowledged when the terminal count is reached on the CVSDINn DMA operation. If a DMA request on either the CVSDOUTn or CVSDINn channel remains unacknowledged, the CVSD/PCM module will suspend operation just before the end of the current CVSD encoding cycle. When new CVSD data becomes available, the outstanding DMA requests will be acknowledged and the CVSD/PCM module will restart.

26.9.2.2 PCM Receive Path

The CVSD/PCM modules have DMA channels assigned to their PCMOUTn registers. The DMA data transfers from the CVSD/PCM modules to system RAM use indirect transfer, in which each transfer is performed with two adjacent bus cycles (read/write). The operation of the PCMOUT0 DMA channel of CVSD/PCM module 0 is described below. The PCMOUT1 DMA channel operates in a similar way.

The CVSD/PCM module requests a DMA transfer each time a new PCM sample is available in PCMOUT0. If PCMOUT0 is assigned to device A and the DIR bit in the DMA channel control register is clear, the DMA controller reads data from the PCMOUT0 register (device A) and writes the data into system RAM (device B). The DMA controller must be configured so that automatic update of the device A (PCMOUT0) address is disabled. The address of device B (system RAM) must be configured to automatically increment or decrement by 2 after each DMA transfer.

The operation of the CVSD/PCM module (i.e. CVSDINn to PCMOUTn) is controlled by DMA request/acknowledge activity on the CVSDINn and CVSDOUTn DMA channels. The number of DMA transfers required to fill the PCM transmit buffer in system RAM must be specified in the DMA block length register. When the required number of DMA transfers have been performed, the DMA controller will not acknowledge further DMA requests on the PCMINn channel. Similarly, further requests on the CVSDINn and CVSDOUTn channels will not be acknowledged when the terminal count is reached on their DMA transfers. The CVSD/PCM module will suspend operation when DMA requests on the CVSDINn or CVDOUTn channels are no longer acknowledged by the DMA controller.

The CVSD/PCM module requires 250 clock cycles of the Auxiliary Clock to decode a PCM sample. The frequency of the Auxiliary Clock determines the rate at which DMA requests are made on the PCMOUTn Channel.

26.9.2.3 CVSD Transmit/Receive Paths

The CVSD/PCM modules operate in their free-running mode with DMA channels assigned to their CVSDINn and CVSDOUTn registers. Activity on these DMA channels is linked to the CVSDINn and CVSDOUTn FIFOs. The operation of the CVSD/PCM module 0 channels is described below. The CVSD/PCM module 1 channels operate in a similar way.

Initially when the CVSD/PCM module 0 is enabled and a DMA request on CVSDIN0 is asserted, a 16-bit CVSD-encoded data word is transferred from memory to the CVSDIN0 FIFO. Further DMA requests are asserted each time the CVSDIN0 FIFO is read by the CVSD/PCM converter. The CVSD/PCM converter requires 250 clock cycles of the Auxiliary Clock 2 (Auxiliary Clock 3 for CVSD/PCM module 1) to produce a single PCM sample. The effective compression ratio of 2:1 results in a 16-bit word of CVSD data every 500 auxiliary clock cycles. At the start of each decoding cycle (500 auxiliary clock cycles), a DMA request is made to transfer a CVSD-encoded data word from Bluetooth RAM to the CVSDIN0 FIFO. The frequency of the auxiliary clock determines the time interval between successive DMA requests.

The number of DMA transfers required to service the CVSD buffer in Bluetooth RAM must be specified in the DMA block length register. When the required number of DMA transfers has been executed, the DMA controller will not acknowledge further DMA requests on the CVSDIN0 channel. The CVSD/PCM module will suspend operation just before the end of the current CVSD decoding cycle. When a new CVSD buffer becomes available, then the outstanding DMA requests will be acknowledged and the CVSD/PCM module will restart.

When the CVSDOUT0 FIFO contains 5 entries or more, a DMA transfer is requested. If CVSDIN0 is assigned to device A and the DIR bit in the DMA channel control register is clear, the DMA controller reads data from the CVSDOUT0 register (device A) and writes the data into the Bluetooth RAM (device B).

26.10 Bus Bandwidth and Latency Considerations

The DMA controller has a burst buffer, which can be enabled on a per-channel basis. When the buffer is enabled, DMA operations occur as four-cycle burst transfers. These burst transfers optimize bus bandwidth, because they eliminate extra cycles used to arbitrate for control of the bus (as compared to four single-cycle transfers).

However, burst mode is not recommended for audio data because of its impact on latency. Three additional data words will be needed to fill the buffer, so for example at an 8 ksps rate, 3 × 125 ms = 375 ms additional latency.

Latency may also be impacted by competition with the CPU and DSP for access to shared resources such as the shared RAM and the external bus interface. However, the bandwidth required even for high-quality audio signals is quite low compared to the bandwidth available from the on-chip buses. For example, a DMA transfer between two peripherals for a 48 ksps audio channel on an APB bus operating at half the speed of the CPU or DSP core bus would require 6 × 48000 = 0.288M core bus cycles/second. Compared to the 96M core bus cycles/second available on the CPU and DSP buses, this is a very small number. It can become significant, however, if the core buses are operated at low clock rates to reduce power consumption.

When the CPU and DSP both attempt to access the same device at the same time, the CPU has priority.

Access to slow external devices by the CPU or DSP will not block on-chip peripheral-to-peripheral or peripheral-to-memory DMA transfers. because the EBIU has a write buffer to shield the on-chip buses from external bus write latency and the core buses are split-transaction buses, so external bus reads from slow devices are non-blocking.

26.11 Freeze Mode

When Freeze mode is entered, the CVSD/PCM converters will exhibit the following behavior:

- CVSD In FIFO will not have data removed by the converter core.
- CVSD Out FIFO will not have data added by the converter core.
- PCM Out buffer will not be updated by the converter core.
- The automatic clear-on-read function of the following status bits in the CVSTATn register is disabled:
  - PCMINT
  - CVE
  - CVF

26.12 CVSD/PCM Converter Registers

Table 26-1 lists the CVSD/PCM registers

Table 26-1. CVSD/PCM Registers

NAME	ADDRESS	DESCRIPTION
CVSDIN0	FF 4800h	CVSD Data Input Register 0
CVSDIN1	FF 4C00h	CVSD Data Input Register 1
CVSDOUT0	FF 4804h	CVSD Data Output Register 0
CVSDOUT1	FF 4C04h	CVSD Data Output Register 1
PCMIN0	FF 4808h	PCM Data Input Register 0
PCMIN1	FF 4C08h	PCM Data Input Register 1
PCMOUT0	FF 480Ch	PCM Data Output Register 0
PCMOUT1	FF 4C0Ch	PCM Data Output Register 1
LOGIN0	FF 4810h	Logarithmic PCM Data Output Register 0
LOGIN1	FF 4C10h	Logarithmic PCM Data Output Register 1
LOGOUT0	FF 4814h	Logarithmic PCM Data Output Register 0
LOGOUT1	FF 4C14h	Logarithmic PCM Data Output Register 1
LINEARIN0	FF 4818h	Linear PCM Data Input Register 0
LINEARIN1	FF 4C18h	Linear PCM Data Input Register 1
LINEAROUT0	FF 481Ch	Linear PCM Data Output Register 0
LINEAROUT1	FF 4C1Ch	Linear PCM Data Output Register 1
CVCTRL0	FF 4820h	CVSD Control Register 0
CVCTRL1	FF 4C20h	CVSD Control Register 1
CVSTAT0	FF 4824h	CVSD Status Register 0
CVSTAT1	FF 4C24h	CVSD Status Register 1

26.12.1 CVSD Data Input Register n (CVSDINn)

The CVSDINn registers are 16-bit, write-only registers. They are used to write CVSD data into the CVSD to PCM converter FIFOs. The FIFOs are 8 words deep. The CVSDIN bit 15 represents the CVSD data bit at t = t_0, CVSDIN bit 0 represents the CVSD data bit at t = t_0 – 250 µs.

15	0
CVSDIN	

26.12.2 CVSD Data Output Register n (CVSDOUTn)

The CVSDOUTn registers are 16-bit, read-only registers. They are used to read the CVSD data from the PCM to CVSD converter FIFOs. The FIFOs are 8 words deep. Reading a CVSDOUTn register after reset returns undefined data.

15	0
CVSDOUT	

26.12.3 PCM Data Input Register n (PCMINn)

The PCMINn registers are 16-bit, write-only registers. They are used to write PCM data to the PCM to CVSD converter via the APB bus. They are double-buffered, providing a 125 µs period for an interrupt or DMA request to respond.

15															0
PCMIN															

26.12.4 PCM Data Output Register (PCMOUTn)

The PCMOUTn registers are 16-bit, read-only registers. They are used to read PCM data from the CVSD to PCM converter. They are double-buffered, providing a 125 µs period for an interrupt or DMA request to respond. After reset the PCMOUT registers are clear.

15															0
PCMOUT															

26.12.5 Logarithmic PCM Data Input Register n (LOGINn)

The LOGINn registers are 8-bit, write-only registers. They are used to receive 8-bit logarithmic PCM data from the APB bus and convert it into 13-bit linear PCM data.

7							0
LOGIN							

26.12.6 Logarithmic PCM Data Output Register n (LOGOUTn)

The LOGOUTn registers are 8-bit, read-only registers. They hold logarithmic PCM data that has been converted from linear PCM data. After reset, the LOGOUT registers are clear.

7							0
LOGOUT							

26.12.7 Linear PCM Data Input Register n (LINEARINn)

The LINEARINn registers are 16-bit, write-only registers. The data is left-aligned. When converting to A-law, bits 2:0 are ignored. When converting to µ-law, bits 1:0 are ignored

15															0
LINEARIN															

26.12.8 Linear PCM Data Output Register n (LINEAROUTn)

The LINEAROUTn registers are 16-bit, read-only registers. The data is left-aligned. When converting from A-law, bits 2:0 are clear. When converting from µ-law, bits 1:0 are clear. After reset, this register is clear.

15															0
LINEAROUT															

26.12.9 CVSD Control Register n (CVCTRLn)

The CVCTRLn registers are 16-bit, read/write registers that control interrupts, DMA, and modes of operation. At reset, all implemented bits are cleared.

7	6	5	4	3	2	1	0
DMAPO	DMACI	DMACO	CVSDERRINT	CVSDINT	PCMINT	CLKEN	CVEN

15 14	13 12	11	10 9	8
Reserved	RESOLUTION	PCMCONV	CVSDCONV	DMAPI

CVEN	The Module Enable bit enables or disables the CVSD conversion module interface. When the bit is set, the interface is enabled which allows read and write operations to the rest of the module. When the bit is clear, the module is disabled. When the module is disabled the status register CVSTAT will be cleared to its reset state. 0 – CVSD module enabled. 1 – CVSD module disabled.
CLKEN	The CVSD Clock Enable bit enables the 2-MHz clock to the filter engine and CVSD encoders and decoders. 0 – CVSD module clock disabled. 1 – CVSD module clock enabled.
PCMINT	The PCM Interrupt Enable bit controls generation of the PCM interrupt. If set, this bit enables the PCM interrupt. If the PCMINT bit is clear, the PCM interrupt is disabled. 0 – PCM interrupt disabled. 1 – PCM interrupt enabled.
CVSDINT	The CVSD FIFO Interrupt Enable bit controls generation of the CVSD interrupt. If set, this bit enables the CVSD interrupt that occurs if the CVSD In FIFO is nearly empty or the CVSD Out FIFO is nearly full. If the CVSDINT bit is clear, the CVSD nearly full/nearly empty interrupt is disabled. 0 – CVSD interrupt disabled. 1 – CVSD interrupt enabled.
CVSDERRINT	The CVSD FIFO Error Interrupt Enable bit controls generation of the CVSD error interrupt. If set, this bit enables an interrupt to occur when the CVSD Out FIFO is full or the CVSD In FIFO is empty. If the CVSDERRINT bit is clear, the CVSD full/empty interrupt is disabled. 0 – CVSD error interrupt disabled. 1 – CVSD error interrupt enabled.
DMACO	The DMA Enable for CVSD Out bit enables hardware DMA control for reading CVSD data from the CVSD Out FIFO. If clear, DMA support is disabled. 0 – CVSD output DMA disabled. 1 – CVSD output DMA enabled.
DMACI	The DMA Enable for CVSD In bit enables hardware DMA control for writing CVSD data into the CVSD In FIFO. If clear, DMA support is disabled. 0 – CVSD input DMA disabled. 1 – CVSD input DMA enabled.
DMAPO	The DMA Enable for PCM Out bit enables hardware DMA control for reading PCM data from the PCMOUT register. If clear, DMA support is disabled. 0 – PCM output DMA disabled. 1 – PCM output DMA enabled.
DMAPI	The DMA Enable for PCM In bit enables hardware DMA control for writing PCM data into the PCMIN register. If cleared, DMA support is disabled. 0 – PCM input DMA disabled. 1 – PCM input DMA enabled.
CVSDCONV	The CVSD to PCM Conversion Format field specifies the PCM format for CVSD/PCM conversions. 00 – CVSD <-> 8-bit μ-Law PCM. 01 – CVSD <-> 8-bit A-Law PCM. 10 – CVSD <-> Linear PCM. 11 – Reserved.

PCMCONV — The PCM to PCM Conversion Format bit selects the PCM format for PCM/PCM conversions.
0 – Linear PCM <-> 8-bit µ-Law PCM
1 – Linear PCM <-> 8-bit A-Law PCM

RESOLUTION — The Linear PCM Resolution field specifies the attenuation of the PCM data for the linear PCM to CVSD conversions by right shifting and sign extending the data. This affects the log PCM data as well as the linear PCM data. The log data is converted to either left-justified zero-stuffed 13-bit (A-law) or 14-bit (µ-law). The RESOLUTION field can be used to compensate for any change in average levels resulting from this conversion.
00 – No shift.
01 – 1-bit attenuation.
10 – 2-bit attenuation.
11 – 3-bit attenuation.

26.12.10 CVSD Status Register n (CVSTATn)

The CVSTATn registers are 16-bit, read-only registers that hold the status information of the CVSD/PCM modules. At reset or when the module is disabled (CVCTRLn.CVEN bit is cleared), all implemented bits are cleared.

15 11	10 8	7 5	4	3	2	1	0
Reserved	CVOUTST	CVINST	CVF	CVE	PCMINT	CVNF	CVNE

CVNE — The CVSD In FIFO Nearly Empty bit indicates when only three CVSD data words are left in the CVSD In FIFO, so new CVSD data should be written into the CVSD In FIFO. If the CVSDINT bit is set, an interrupt will be asserted when the CVNE bit is set. If the DMACI bit is set, a DMA request will be asserted when this bit is set. The CVNE bit is cleared when the CVSTATn register is read.
0 – CVSD In FIFO is not nearly empty.
1 – CVSD In FIFO is nearly empty.

CVNF — The CVSD Out FIFO Nearly Full bit indicates when only three empty word locations are left in the CVSD Out FIFO, so the CVSD Out FIFO should be read. If the CVSDINT bit is set, an interrupt will be asserted when the CVNF bit is set. If the DMACO bit is set, a DMA request will be asserted when this bit is set. Software must not rely on the CVNF bit as an indicator of the number of valid words in the FIFO. Software must check the CVOUTST field to read the number of valid words in the FIFO. The CVNF bit is cleared when the CVSTATn register is read.
0 – CVSD Out FIFO is not nearly full.
1 – CVSD Out FIFO is nearly full.

PCMINT — The PCM Interrupt bit set indicates that the PCMOUTn register is full and needs to be read or the PCMINn register is empty and needs to be loaded with new PCM data. The PCMINT bit is cleared when the CVSTATn register is read, unless the device is in Freeze mode.
0 – PCM does not require service.
1 – PCM requires loading or unloading.

CVE — The CVSD In FIFO Empty bit indicates when the CVSD In FIFO has been read by the CVSD converter while the FIFO was already empty. If the CVSDERRINT bit is set, an interrupt will be asserted when the CVE bit is set. The CVE bit is cleared when the CVSTATn register is read, unless the device is in Freeze mode.
0 – CVSD In FIFO has not been read while empty.
1 – CVSD In FIFO has been read while empty.

CVF	The CVSD Out FIFO Full bit set indicates whether the CVSD Out FIFO has been written by the CVSD converter while the FIFO was already full. If the CVSDERRINT bit is set, an interrupt will be asserted when the CVF bit is set. The CVF bit is cleared when the CVSTATn register is read, unless the device is in Freeze mode. 0 – CVSD Out FIFO has not been written while full. 1 – CVSD Out FIFO has been written while full.
CVINST	The CVSD In FIFO Status field indicates the current number of empty 16-bit word locations in the CVSD In FIFO. The field is biased by 1, so when there are 8 empty words (FIFO is empty), the CVINST field will read as 111b. When there are 1 or 0 empty words (FIFO holds 7 or 8 words of data), the CVINST field will read as 000b.
CVOUTST	CVSD Out FIFO Status field indicates the current number of valid 16-bit CVSD data words in the CVSD Out FIFO. When the FIFO is empty, the CVOUTST field will read as 000b. When the FIFO holds 7 or 8 words of data, the CVOUTST field will read as 111b.

27 Dual/Quad UART

The CP3SP33 provides four Universal Asynchronous Receiver/Transmitter (UART) modules in the FBGA-224 package, and two UART modules in the FBGA-144 package. Each module supports a wide range of software-programmable baud rates (up to 3.072 Mbaud) and data formats. It handles automatic parity generation and several error detection schemes.

All UART modules offer the following features:

- Full-duplex double-buffered receiver/transmitter
- Asynchronous operation
- Programmable baud rate
- Programmable framing formats: 7, 8, or 9 data bits; even, odd, or no parity; one or two stop bits (mark or space)
- Hardware parity generation for data transmission and parity check for data reception
- Interrupt or DMA requests on transmit ready and receive ready conditions, separately enabled
- Software-controlled break transmission and detection
- Internal diagnostic capability
- Automatic detection of parity, framing, and overrun errors

The UART modules are referred to as UART0, UART1, UART2, and UART3. UART2 and UART3 are not available in the FBGA-144 package.

UART0, UART1, and UART2 reside on the CPU peripheral APB bus. UART3 is on the shared audio peripheral APB bus.

UART0 offers synchronous operation using the CKX external clock pin in the FBGA-224 package. The CKX signal is not available in the FBGA-144 package.

UART0, UART1, and UART2 support hardware flow control using the $\overline{\text{CTSn}}$ and $\overline{\text{RTSn}}$ signals.

27.1 Functional Overview

Figure 27-1 is a block diagram of the UART module showing the basic functional units in the UART:

- Transmitter
- Receiver
- Baud Rate Generator
- Control and Error Detection

The Transmitter block consists of an 8-bit transmit shift register and an 8-bit transmit buffer. Data bytes are loaded in parallel from the buffer into the shift register and then shifted out serially on the TXDn pin.

The Receiver block consists of an 8-bit receive shift register and an 8-bit receive buffer. Data is received serially on the RXDn pin and shifted into the shift register. Once eight bits have been received, the contents of the shift register are transferred in parallel to the receive buffer.

The Transmitter and Receiver blocks both contain extensions for 9-bit data transfers, as required by the 9-bit and loopback operating modes.

The Baud Rate Generator generates the clock for the synchronous and asynchronous operating modes. It consists of two registers and a two-stage counter. The registers are used to specify a prescaler value and a baud rate divisor. The first stage of the counter divides the UART clock based on the value of the programmed prescaler to create a slower clock. The second stage of the counter creates the baud rate clock by dividing the output of the first stage based on the programmed baud rate divisor.

The Control and Error Detection block contains the UART control registers, control logic, error detection circuit, parity generator/checker, and interrupt generation logic. The control registers and control logic determine the data format, mode of operation, clock source, and type of parity used. The error detection circuit generates parity bits and checks for parity, framing, and overrun errors.

The Flow Control Logic block provides the capability for hardware handshaking between the UART and a peripheral device. When the peripheral device needs to stop the flow of data from the UART, it deasserts the clear-to-send ($\overline{\text{CTSn}}$) signal which causes the UART to pause after sending the current frame (if any). The UART asserts the ready-to-send ($\overline{\text{RTSn}}$) signal to the peripheral when it is ready to send a character.

27.2 UART Operation

The UART has two basic modes of operation: synchronous and asynchronous. Synchronous mode is only supported for the UART0 module. In addition, there are two special-purpose modes, called attention and diagnostic. This section describes the operating modes of the UART.

27.2.1 Asynchronous Mode

The asynchronous mode of the UART enables the device to communicate with other devices using just two communication signals: transmit and receive.

In asynchronous mode, the transmit shift register (TSFT) and the transmit buffer (UnTBUF) double-buffer the data for transmission. To transmit a character, a data byte is loaded in the UnTBUF register. The data is then transferred to the TSFT register. While the TSFT register is shifting out the current character (LSB first) on the TXD pin, the UnTBUF register is loaded by software with the next byte to be transmitted. When TSFT finishes transmission of the last stop bit of the current frame, the contents of UnTBUF are transferred to the TSFT register and the Transmit Buffer Empty bit (UTBE) is set. The UTBE bit is automatically cleared by the UART when software loads a new character into the UnTBUF register. During transmission, the UXMIP bit is set high by the UART. This bit is reset only after the UART has sent the last stop bit of the current character and the UnTBUF register is empty. The UnTBUF register is a read/write register. The TSFT register is not software accessible.

In asynchronous mode, the input frequency to the UART is 16 times the baud rate. In other words, there are 16 clock cycles per bit time. In asynchronous mode, the baud rate generator is always the UART clock source.

The receive shift register (RSFT) and the receive buffer (UnRBUF) double buffer the data being received. The UART receiver continuously monitors the signal on the RXDn pin for a low level to detect the beginning of a start bit. On sensing this low level, the UART waits for seven input clock cycles and samples again three times. If all three samples still indicate a valid low, then the receiver considers this to be a valid start bit, and the remaining bits in the character frame are each sampled three times, around the mid-bit position. For any bit following the start bit, the logic value is found by majority voting, i.e. the two samples with the same value define the value of the data bit. Figure 27-2 illustrates the process of start bit detection and bit sampling.

Data bits are sensed by taking a majority vote of three samples latched near the midpoint of each baud (bit time). Normally, the position of the samples within the baud is determined automatically, but software can override the automatic selection by setting the USMD bit in the UnMDSL2 register and programming the UnSPOS register.

Serial data input on the RXDn pin is shifted into the RSFT register. On receiving the complete character, the contents of the RSFT register are copied into the UnRBUF register and the Receive Buffer Full bit (URBF) is set. The URBF bit is automatically cleared when software reads the character from the URBUF register. The RSFT register is not software accessible.

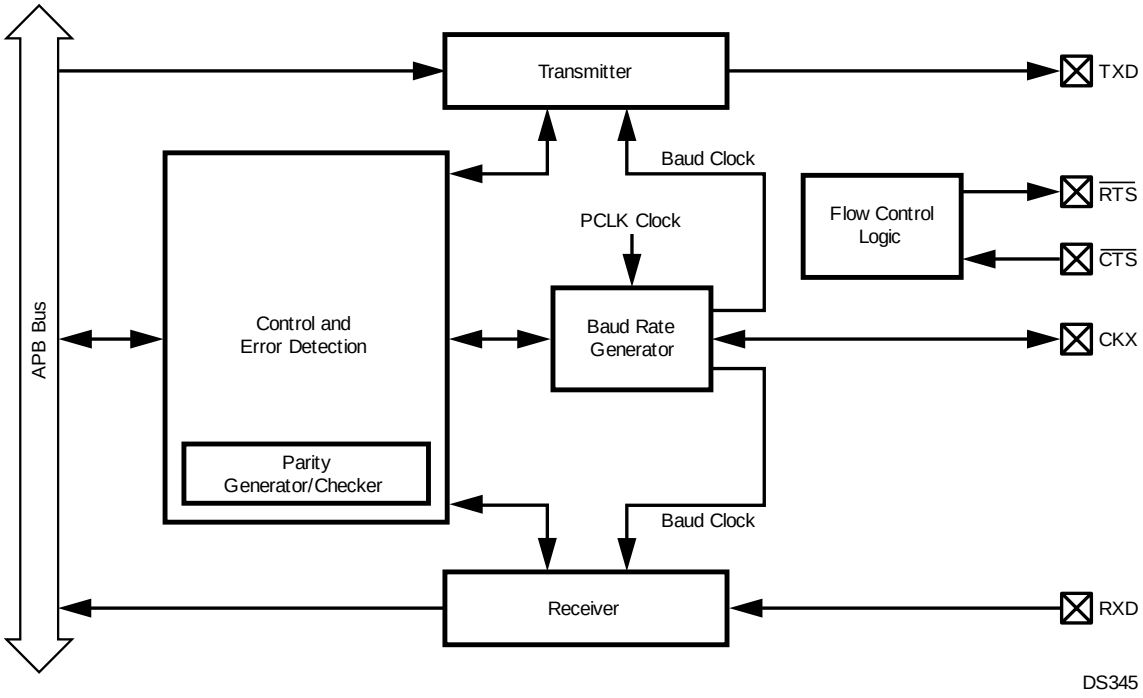


Figure 27-1. UART Block Diagram

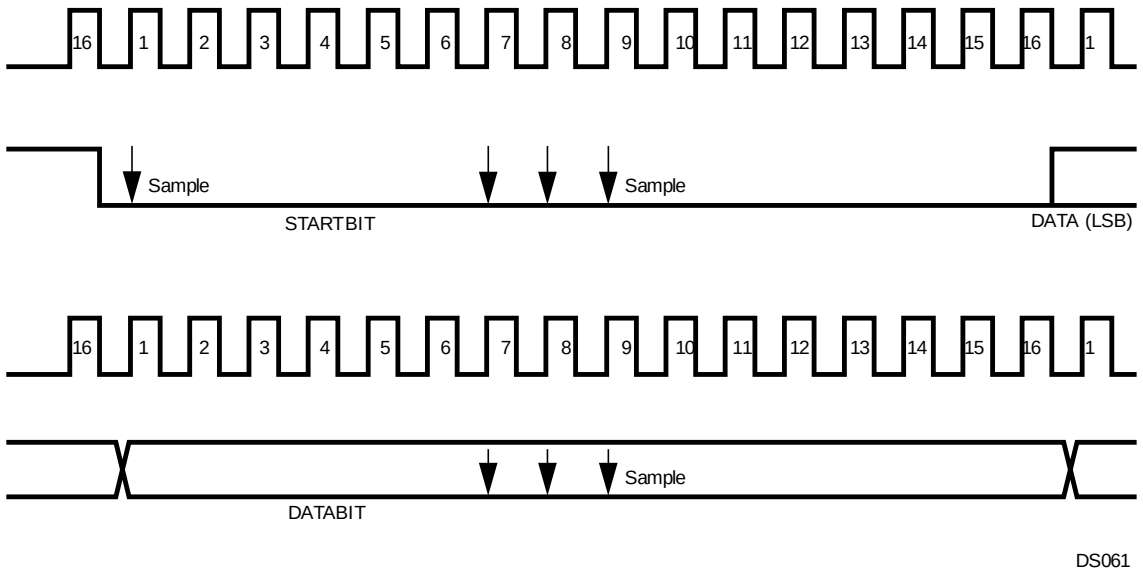


Figure 27-2. UART Asynchronous Communication

27.2.2 Synchronous Mode

The synchronous mode of UART0 enables the device to communicate with other devices using three communication signals: transmit, receive, and clock. In this mode, data bits are transferred synchronously with the UART0 clock signal. Data bits are transmitted on the rising edges and received on the falling edges of the clock signal, as shown in Figure 27-3. Data bytes are transmitted and received least significant bit (LSB) first.

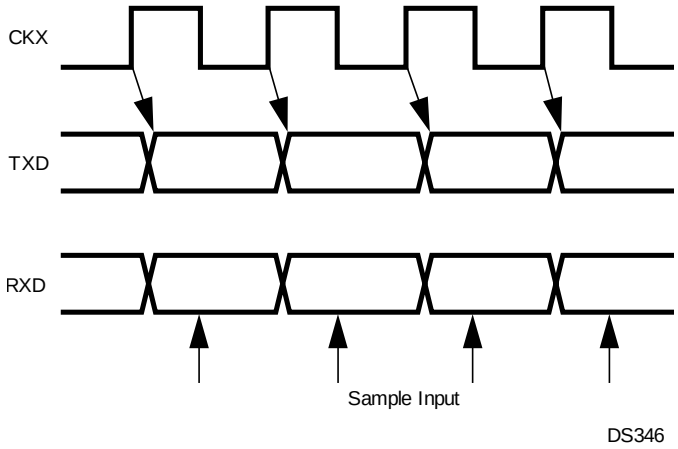


Figure 27-3. UART Synchronous Communication

In synchronous mode, the transmit shift register (TSFT) and the transmit buffer (U0TBUF) double-buffer the data for transmission. To transmit a character, a data byte is loaded in the U0TBUF register. The data is then transferred to the TSFT register. The TSFT register shifts out one bit of the current character, LSB first, on each rising edge of the clock. While the TSFT is shifting out the current character on the TXD0 pin, the U0TBUF register may be loaded by software with the next byte to be transmitted. When the TSFT finishes transmission of the last stop bit within the current frame, the contents of U0TBUF are transferred to the TSFT register and the Transmit Buffer Empty bit (UTBE) is set. The UTBE bit is automatically reset by the UART when software loads a new character into the U0TBUF register. During transmission, the UXMIP bit is set by the UART. This bit is cleared only after the UART has sent the last frame bit of the current character and the U0TBUF register is empty.

The receive shift register (RSFT) and the receive buffer (URBUF) double-buffer the data being received. Serial data received on the RXD0 pin is shifted into the RSFT register on the first falling edge of the clock. Each subsequent falling edge of the clock causes an additional bit to be shifted into the RSFT register. The UART assumes a complete character has been received after the correct number of rising edges on CKX (based on the selected frame format) have been detected. On receiving a complete character, the contents of the RSFT register are copied into the U0RBUF register and the Receive Buffer Full bit (URBF) is set. The URBF bit is automatically cleared when software reads the character from the U0RBUF register.

The transmitter and receiver may be clocked by either an external source provided to the CKX pin or the internal baud rate generator. In the latter case, the clock signal is placed on the CKX pin as an output.

27.2.3 Attention Mode

The Attention mode is available for networking this device with other processors. This mode requires the 9-bit data for- mat with no parity. The number of start bits and number of stop bits are programmable. In this mode, two types of 9-bit characters are sent on the network: address characters consisting of 8 address bits and a 1 in the ninth bit position and data characters consisting of 8 data bits and a 0 in the ninth bit position.

While in Attention mode, the UART receiver monitors the communication flow but ignores all characters until an ad- dress character is received. On receiving an address character, the contents of the receive shift register are copied to the receive buffer. The URBF bit is set and an interrupt (if enabled) is generated. The UATN bit is automatically cleared, and the UART begins receiving all subsequent characters. Software must examine the contents of the URBUF register and respond by accepting the subsequent characters (by leaving the UATN bit clear) or waiting for the next address character (by setting the UATN bit again).

The operation of the UART transmitter is not affected by the selection of this mode. The value of the ninth bit to be transmitted is programmed by setting or clearing the UXB9 bit in the UART Frame Select register. The value of the ninth bit received is read from URB9 in the UART Status Register.

27.2.4 Diagnostic Mode

The Diagnostic mode is available for testing of the UART. In this mode, the TXDn and RXDn pins are internally connected together, and data shifted out of the transmit shift register is immediately transferred to the receive shift register. This mode supports only the 9-bit data format with no parity. The number of start and stop bits is programmable.

27.2.5 Frame Format Selection

The format shown in Figure 27-4 consists of a start bit, seven data bits (excluding parity), and one or two stop bits. If parity bit generation is enabled by setting the UPEN bit, a parity bit is generated and transmitted following the seven data bits.

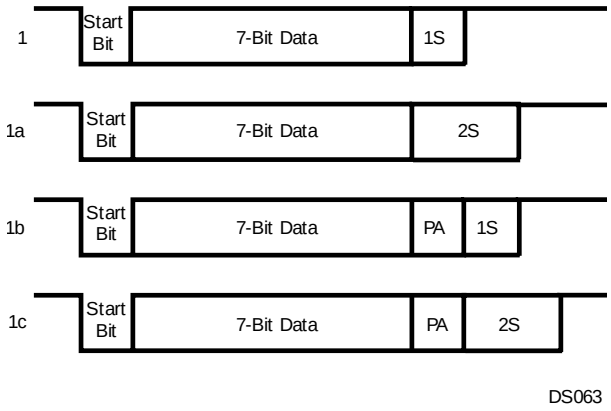


Figure 27-4. 7-Bit Data Frame Options

The format shown in Figure 27-5 consists of one start bit, eight data bits (excluding parity), and one or two stop bits. If parity bit generation is enabled by setting the UPEN bit, a parity bit is generated and transmitted following the eight data bits.

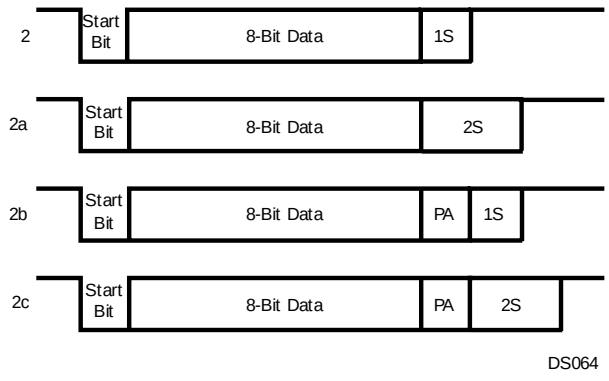


Figure 27-5. 8-Bit Data Frame Options

The format shown in Figure 27-6 consists of one start bit, nine data bits, and one or two stop bits. This format also supports the UART attention feature. When operating in this format, all eight bits of UnTBUF and UnRBUF are used for data. The ninth data bit is transmitted and received using two bits in the control registers, called UXB9 and URB9. Parity is not generated or verified in this mode.

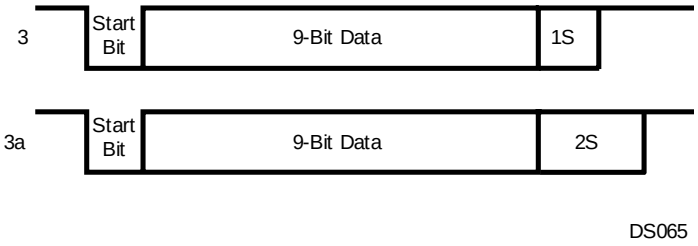


Figure 27-6. . 9-bit Data Frame Options

27.2.6 Baud Rate Generator

The Baud Rate Generator creates the basic baud clock from the PCLK Clock. The PCLK Clock is passed through a two-stage divider chain consisting of a 5-bit baud rate prescaler (UnPSC) and an 11-bit baud rate divisor (UnDIV).

The relationship between the 5-bit prescaler select (UnPSC) setting and the prescaler factors is shown in Table 27-1.

Table 27-1. Prescaler Factors

PRESCALER SELECT	PRESCALER FACTOR		PRESCALER SELECT	PRESCALER FACTOR
00000	No clock		10000	8.5
00001	1		10001	9
00010	1.5		10010	9.5
00011	2		10011	10
00100	2.5		10100	10.5
00101	3		10101	11
00110	3.5		10110	11.5
00111	4		10111	12
01000	4.5		11000	12.5
01001	5		11001	13
01010	5.5		11010	13.5
01011	6		11011	14
01100	6.5		11100	14.5
01101	7		11101	15
01110	7.5		11110	15.5
01111	8		11111	16

A prescaler factor of zero corresponds to “no clock.” The “no clock” condition is the UART power down mode, in which the UART clock is turned off to reduce power consumption. Software must select the “no clock” condition before entering a new baud rate. Otherwise, it could cause incorrect data to be received or transmitted. The UnPSR register must contain a value other than zero when an external clock is used at CKX.

27.2.7 Interrupts

The UART modules are capable of generating interrupts on:

- Receive Buffer Full
- Receive Error
- Transmit Buffer Empty
- Clear To Send

Figure 27-7 shows a diagram of the interrupt sources and associated enable bits.

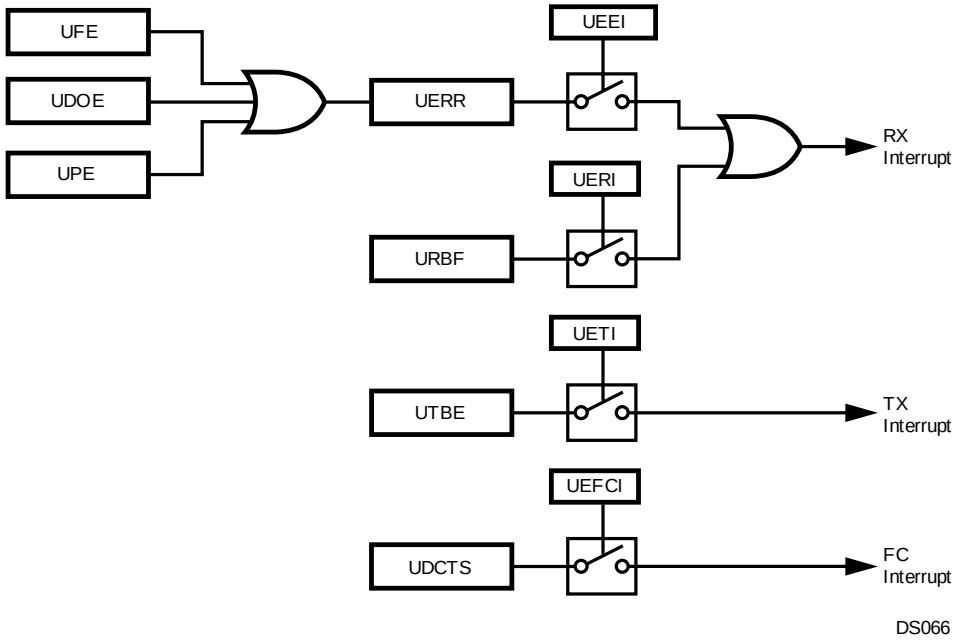


Figure 27-7. UART Interrupts

The interrupts can be individually enabled or disabled using the Enable Transmit Interrupt (UETI), Enable Receive Interrupt (UERI), and Enable Receive Error Interrupt (UEER) bits in the UnICTRL register.

A transmit interrupt is asserted when both the UTBE and UETI bits are set. To remove this interrupt, software must either disable the interrupt by clearing the UETI bit or write to the UnTBUF register (which clears the UTBE bit).

A receive interrupt is asserted on these conditions:

- Both the URBF and UERI bits are set. To remove this interrupt, software must either disable the interrupt by clearing the UERI bit or read from the URBUF register (which clears the URBF bit).
- Both the UERR and the UEEI bits are set. To remove this interrupt, software must either disable the interrupt by clearing the UEEI bit or read the UnSTAT register (which clears the UERR bit).

A flow control interrupt is asserted when both the UDCTS and the UEFCI bits are set. To remove this interrupt, software must either disable the interrupt by clearing the UEFCI bit or reading the UnICTRL register (which clears the UDCTS bit).

Because the UART3 transmit and receive interrupt requests can be handled by either the CPU or DSP interrupt controllers, the interrupt requests are passed through the Audio Subsystem Controller (ASC), which must be programmed to route it to the interrupt controller. To use these interrupts, they must be enabled at three levels:

- *UART3 module*—the interrupt must be enabled in the U3ICTRL register.

- *ASC module*—the UART3 interrupts are assigned to ASC interrupt channels 2 (transmit) and 3 (receive), so bits 2 and 3 in the ASCINTSEL register must be clear to select the CPU interrupt controller or set to select the DSP interrupt controller.
- *Interrupt controller*—the interrupt request must be enabled in the interrupt controller. For the CPU interrupt controller, these are interrupt requests IRQ12 (transmit) and IRQ13 (receive).

27.2.8 DMA Support

Any UART module can operate with one or two DMA chan- nels. For processor-independent full-duplex operation, two DMA channels must be used. Both receive and transmit DMA can be enabled simultaneously.

If transmit DMA is enabled (the UETD bit is set), the UART generates a DMA request when the UTBE bit changes state from clear to set. Enabling transmit DMA automatically dis- ables transmit interrupts, without regard to the state of the UETI bit.

If receive DMA is enabled (the UERD bit is set), the UART generates a DMA request when the URBF bit changes state from clear to set. Enabling receive DMA automatically dis- ables receive interrupts, without regard to the state of the UERI bit. However, receive error interrupts should be en- abled (the UEEI bit is set) to allow detection of receive errors when DMA is used.

Because the UART3 DMA requests can be handled by ei- ther the CPU or DSP DMA controllers, the DMA requests are passed through the Audio Subsystem Controller (ASC), which must be programmed to route them to the DMA con- troller. To use the UART3 DMA requests, they must be en- abled at three levels:

- *UART3 module*—the UART3 DMA requests must be enabled in the U3MDSL1 register.
- *ASC module*—the UART3 DMA requests are assigned to ASC DMA channels 0 (receive) and 1 (transmit). These channels are controlled by bits 0 and 1 in the ASCDMASEL0 register. The bits must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be programmed in the ASCDDMASELn registers.
- *DMA controller*—the DMA requests must be enabled in the DMA controller. For the CPU DMA controller, these are DMA requests 0 (receive) and 1 (transmit).

27.2.9 Break Generation and Detection

A line break is generated when the UBRK bit is set in the UnMDSL1 register. The TXDn line remains low until the program resets the UBRK bit.

A line break is detected if RXDn remains low for 10 bit times or longer after a missing stop bit is detected.

27.2.10 Parity Generation and Detection

Parity is only generated or checked with the 7-bit and 8-bit data formats. It is not generated or checked in the diagnostic loopback mode, the attention mode, or in normal mode with the 9-bit data format. Parity generation and checking are enabled and disabled using the PEN bit in the UnFRS register. The UPSEL bits in the UnFRS register are used to select odd, even, or no parity.

27.3 UART Registers

Software interacts with the UART modules by accessing the UART registers, as listed in Table 27-2.

Table 27-2. UART Registers

NAME	ADDRESS	DESCRIPTION
U0ICTRL	FF 9408h	UART0 Interrupt Control Register
U0STAT	FF 940Ch	UART0 Status Register
U0MDSL1	FF 9414h	UART0 Mode Select Register 1
U0MDSL2	FF 9424h	UART0 Mode Select Register 2

Table 27-2. UART Registers (continued)

NAME	ADDRESS	DESCRIPTION
U0PSR	FF 941Ch	UART0 Baud Rate Prescaler
U0BAUD	FF 9418h	UART0 Baud Rate Divisor
U0FRS	FF 9410h	UART0 Frame Select Register
U0SPOS	FF 9428h	UART0 Sample Position Register
U0OVR	FF 9420h	UART0 Oversample Rate Register
U0RBUF	FF 9404h	UART0 Receive Data Buffer
U0TBUF	FF 9400h	UART0 Transmit Data Buffer
U1ICTRL	FF 9808h	UART1 Interrupt Control Register
U1STAT	FF 980Ch	UART1 Status Register
U1MDSL1	FF 9814h	UART1 Mode Select Register 1
U1MDSL2	FF 9824h	UART1 Mode Select Register 2
U1PSR	FF 981Ch	UART1 Baud Rate Prescaler
U1BAUD	FF 9818h	UART1 Baud Rate Divisor
U1FRS	FF 9810h	UART1 Frame Select Register
U1SPOS	FF 9828h	UART1 Sample Position Register
U1OVR	FF 9820h	UART1 Oversample Rate Register
U1RBUF	FF 9804h	UART1 Receive Data Buffer
U1TBUF	FF 9800h	UART1 Transmit Data Buffer
U2ICTRL	FF 9C08h	UART2 Interrupt Control Register
U2STAT	FF 9C0Ch	UART2 Status Register
U2MDSL1	FF 9C14h	UART2 Mode Select Register 1
U2MDSL2	FF 9C24h	UART2 Mode Select Register 2
U2PSR	FF 9C1Ch	UART2 Baud Rate Prescaler
U2BAUD	FF 9C18h	UART2 Baud Rate Divisor
U2FRS	FF 9C10h	UART2 Frame Select Register
U2SPOS	FF 9C28h	UART2 Sample Position Register
U2OVR	FF 9C20h	UART2 Oversample Rate Register
U2RBUF	FF 9C04h	UART2 Receive Data Buffer
U2TBUF	FF 9C00h	UART2 Transmit Data Buffer
U3ICTRL	FF 5C08h	UART3 Interrupt Control Register
U3STAT	FF 5C0Ch	UART3 Status Register
U3MDSL1	FF 5C14h	UART3 Mode Select Register 1
U3MDSL2	FF 5C24h	UART3 Mode Select Register 2
U3PSR	FF 5C1Ch	UART3 Baud Rate Prescaler
U3BAUD	FF 5C18h	UART3 Baud Rate Divisor
U3FRS	FF 5C10h	UART3 Frame Select Register
U3SPOS	FF 5C28h	UART3 Sample Position Register
U3OVR	FF 5C20h	UART3 Oversample Rate Register
U3RBUF	FF 5C04h	UART3 Receive Data Buffer
U3TBUF	FF 5C00h	UART3 Transmit Data Buffer

27.3.1 UART Interrupt Control Register (UnICTRL)

The UnICTRL registers are 8-bit, read/write registers that contain the receive and transmit interrupt status bits (read-only bits) and the interrupt enable bits (read/write bits). The register is initialized to 01h at reset.

7	6	5	4	3	2	1	0
UEEI	UERI	UETI	UEFCI	UCTS	UDCTS	URBF	UTBE

UTBE The Transmit Buffer Empty bit is set by hard- ware when the UART transfers data from the UnTBUF register to the transmit shift register for transmission. It is automatically cleared by the hardware on the next write to the UnTBUF register.
0 – Transmit buffer is loaded.
1 – Transmit buffer is empty.

URBF The Receive Buffer Full bit is set by hardware when the UART has received a complete data frame and has transferred the data from the receive shift register to the UnRBUF register. It is automatically cleared by the hardware when the UnRBUF register is read.
0 – Receive buffer is empty.
1 – Receive buffer is loaded.

UDCTS The Delta Clear To Send bit indicates whether the $\overline{\text{CTSn}}$ input has changed state since the CPU last read this register. This bit is only used with UART0, UART1, and UART2.
0 – No change since last read.
1 – State has changed since last read.

UCTS UCTS The Clear To Send bit indicates the state on the CTS input. This bit is only used with UART0, UART1, and UART2.
0 – $\overline{\text{CTSn}}$ input is high.
1 – $\overline{\text{CTSn}}$ input is low.

UEFCI The Enable Flow Control Interrupt bit controls whether a flow control interrupt is asserted when the UDCTS bit changes from clear to set. This bit is only used with UART0, UART1, and UART2.
0 – Flow control interrupt disabled.
1 – Flow control interrupt enabled.

UETI The Enable Transmitter Interrupt bit, when set, enables generation of an interrupt when the hardware sets the UTBE bit.
0 – Transmit buffer empty interrupt disabled.
1 – Transmit buffer empty interrupt enabled.

UERI The Enable Receiver Interrupt bit, when set, enables generation of an interrupt when the hardware sets the URBF bit.
0 – Receive buffer full interrupt disabled.
1 – Receive buffer full interrupt enabled.

UEEI The Enable Receive Error Interrupt bit, when set, enables generation of an interrupt when the hardware sets the UERR bit in the Un-STAT register.
0 – Receive error interrupt disabled.
1 – Receive error interrupt enabled.

27.3.2 UART Status Register (UnSTAT)

The UnSTAT registers are 8-bit, read-only registers that contain the receive and transmit status bits. These registers are cleared at reset. Any attempt by software to write to these registers is ignored. The register format is shown below.

7	6	5	4	3	2	1	0
Reserved	UXMIP	URB9	UBKD	UERR	UDOE	UFE	UPE

UPE The Parity Error bit indicates whether a parity error is detected within a received character. This bit is automatically cleared by the hardware when the UnSTAT register is read.
0 – No parity error occurred.
1 – Parity error occurred.

UFE The Framing Error bit indicates whether the UART fails to receive a valid stop bit at the end of a frame. This bit is automatically cleared by the hardware when the UnSTAT register is read.
0 – No framing error occurred.
1 – Framing error occurred.

UDOE The Data Overrun Error bit is set when a new character is received and transferred to the UnRBUF register before software has read the previous character from the UnRBUF register. This bit is automatically cleared by the hardware when the UnSTAT register is read.
0 – No receive overrun error occurred.
1 – Receive overrun error occurred.

UERR The Error Status bit indicates when a parity, framing, or overrun error occurs (any time that the UPE, UFE, or UDOE bit is set). It is automatically cleared by the hardware when the UPE, UFE, and UDOE bits are all 0.
0 – No receive error occurred.
1 – Receive error occurred.

UBKD The Break Detect bit indicates when a line break condition occurs. This condition is detected if RXDn remains low for at least ten bit times after a missing stop bit has been detected at the end of a frame. The hardware automatically clears the UBKD bit on reading the UnSTAT register, but only if the break condition on RXDn no longer exists. If reading the UnSTAT register does not clear the UBKD bit because the break is still actively driven on the line, the hardware clears the bit as soon as the break condition no longer exists (when the RXDn input returns to a high level).
0 – No break condition occurred.
1 – Break condition occurred.

URB9 The Received 9th Data Bit holds the ninth data bit, when the UART is configured to operate in the 9-bit data format.

UXMIP The Transmit In Progress bit indicates when the UART is transmitting. The hardware sets this bit when the UART is transmitting data and clears the bit at the end of the last frame bit.
0 – UART is not transmitting.
1 – UART is transmitting.

27.3.3 UART Mode Select Register 1 (UnMDSL1)

The UnMDSL1 registers are 8-bit, read/write registers that select the clock source, synchronization mode, attention mode, and line break generation. These registers are cleared at reset. The register format is shown below.

7	6	5	4	3	2	1	0
URTS	UFCE	UERD	UETD	UCKS	UBRK	UATN	UMOD

UMOD The Mode bit selects between synchronous and asynchronous mode. Synchronous mode is only available for the UART0 module.
0 – Asynchronous mode.
1 – Synchronous mode.

UATN The Attention Mode bit is used to enable Attention mode. When set, this bit selects the attention mode of operation for the UART. When clear, the attention mode is disabled. The hardware clears this bit after an address frame is received. An address frame is a 9-bit character with a 1 in the ninth bit position.
0 – Attention mode disabled.
1 – Attention mode enabled.

UBRK The Force Transmission Break bit is used to force the TXD output low. Setting this bit to 1 causes the TXD pin to go low. TXD remains low until the UBRK bit is cleared by software.
0 – Normal operation.
1 – TXD pin forced low.

UCKS The Synchronous Clock Source bit controls the clock source when the UART operates in the synchronous mode (UMOD = 1). This functionality is only available for the UART0 module. If the UCKS bit is set, the UART operates from an external clock provided on the CKX pin. If the UCKS bit is clear, the UART operates from the baud rate clock produced by the UART on the CKX pin. This bit is ignored when the UART operates in the asynchronous mode.
0 – Internal baud rate clock is used.
1 – External clock is used.

UETD The Enable Transmit DMA bit controls whether DMA is used for UART transmit operations. Enabling transmit DMA automatically disables transmit interrupts, without regard to the state of the UETI bit.
0 – Transmit DMA disabled.
1 – Transmit DMA enabled.

UERD The Enable Receive DMA bit controls whether DMA is used for UART receive operations. Enabling receive DMA automatically disables receive interrupts, without regard to the state of the UERI bit. Receive error interrupts are unaffected by the UERD bit.
0 – Receive DMA disabled.
1 – Receive DMA enabled.

UFCE The Flow Control Enable bit controls whether flow control interrupts are enabled. This functionality is only available for the UART0, UART1, and UART2 modules.
0 – Flow control interrupts disabled.
1 – Flow control interrupts enabled.

URTS The Ready To Send bit directly controls the state of the $\overline{\text{RTS}}$ output. This functionality is only available for the UART0, UART1, and UART2 modules.
0 – $\overline{\text{RTS}}$ output is high.
1 – $\overline{\text{RTS}}$ output is low.

27.3.4 UART Mode Select Register 2 (UnMDSL2)

The UnMDSL2 registers are 8-bit, read/write registers that control the sample mode used to recover asynchronous data. At reset, the UnMDSL2 registers are cleared. The register format is shown below.

7	1	0
Reserved		USMD

USMD The USMD bit controls the sample mode for asynchronous transmission.
0 – UART determines the sample position automatically.
1 – The UnSPOS register determines the sample position.

27.3.5 UART Baud Rate Prescaler (UnPSR)

The UnPSR registers are 8-bit, read/write registers that contains the 5-bit clock prescaler and the upper three bits of the baud rate divisor. These registers are cleared at reset. The register format is shown below.

7	3	2	0
UPSC		UDIV10:8	

UPSC The Prescaler field specifies the prescaler value used for dividing the PCLK Clock in the first stage of the two-stage divider chain. For the prescaler factors corresponding to each 5-bit value, see Table 27-1.

UDIV10:8 The Baud Rate Divisor field holds the three most significant bits (bits 10, 9, and 8) of the UART baud rate divisor used in the second stage of the two-stage divider chain. The remaining bits of the baud rate divisor are held in the UnBAUD register.

27.3.6 UART Baud Rate Divisor (UnBAUD)

The UnBAUD registers are 8-bit, read/write registers that contain the lower eight bits of the baud rate divisor. The register contents are unknown at power-up and are left un- changed by a reset operation. The register format is shown below.

7	0
UDIV7:0	

UDIV7:0 The Baud Rate Divisor field holds the eight lowest-order bits of the UART baud rate divisor used in the second stage of the two-stage divider chain. The three most significant bits are held in the UnPSR register. The divisor value used is (UDIV[10:0] + 1).

27.3.7 UART Frame Select Register (UnFRS)

The UnFRS registers are 8-bit, read/write registers that con- trol the frame format, including the number of data bits, number of stop bits, and parity type. These registers are cleared at reset. The register format is shown below.

7	6	5 4	3	2	1 0
Reserved	UPEN	UPSEL	UXB9	USTP	UCHAR

UCHAR The Character Frame Format field selects the number of data bits per frame, not including the parity bit.
00 – 8 data bits per frame.
01 – 7 data bits per frame.
10 – 9 data bits per frame.
11 – Loop-back mode, 9 data bits per frame.

USTP The Stop Bits bit specifies the number of stop bits transmitted in each frame.
0 – One stop bit per frame.
1 – Two stop bits per frame.

UXB9 The Transmit 9th Data Bit holds the value of the ninth data bit, either 0 or 1, transmitted when the UART is configured to transmit nine data bits per frame. It has no effect when the UART is configured to transmit seven or eight data bits per frame.

UPSEL The Parity Select field selects the treatment of the parity bit. When the UART is configured to transmit nine data bits per frame, the parity bit is omitted and the UPSEL field is ignored.
00 – Odd parity.
01 – Even parity.
10 – No parity, transmit 1 (mark).
11 – No parity, transmit 0 (space).

UPEN The Parity Enable bit enables or disables parity generation and parity checking. When the UART is configured to transmit nine data bits per frame, there is no parity bit and the UnPEN bit is ignored.
0 – Parity generation and checking disabled.
1 – Parity generation and checking enabled.

27.3.8 UART Sample Position Register (UnSPOS)

The UnSPOS registers are 8-bit, read/write registers that specify the sample position when the USMD bit in the UnMDSL2 register is set. At reset, the UnSPOS registers are initialized to 06h. The register format is shown below.

7	4	3	0
Reserved		USAMP	

USAMP The Sample Position field specifies the oversample clock period at which to take the first of three samples for sensing the value of data bits. The clocks are numbered starting at 0 and may range up to 15 for 16× oversampling. The maximum value for this field is (oversampling rate – 3). The table below shows the clock period at which each of the three samples is taken, when automatic sampling is enabled (UnMDSL2.USMD = 0).

OVERSAMPLING RATE	SAMPLE POSITION		
	1	2	3
7	2	3	4
8	2	3	4
9	3	4	5
10	3	4	5
11	4	5	6
12	4	5	6
13	5	6	7
14	5	6	7
15	6	7	8
16	6	7	8

The USAMP field may be used to override the automatic selection, to choose any other clock period at which to start taking the three samples.

27.3.9 UART Oversample Rate Register (UnOVR)

The UnOVR registers are 8-bit, read/write registers that specify the oversample rate. At reset, the UnOVR registers are cleared. The register format is shown below.

7	4	3	0
Reserved		UOVSR	

UOVSO The Oversampling Rate field specifies the oversampling rate, as given in the following table.

UOVSR3:0	OVERSAMPLING RATE
0000–0110	16
111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

27.3.10 UART Receive Data Buffer (UnRBUF)

The UnRBUF registers are 8-bit, read-only registers used to receive each data byte.

7							0
URBUF							

27.3.11 UART Transmit Data Buffer (UnTBUF)

The UnTBUF registers are 8-bit, read/write registers used to transmit each data byte.

7							0
UnTBUF							

27.4 Baud Rate Calculations

The UART baud rate is determined by the PCLK Clock frequency and the values in the UnOVR, UnPSR, and Un- BAUD registers. Unless the PCLK Clock is an exact multiple of the baud rate, there will be a small amount of error in the resulting baud rate.

27.4.1 Asynchronous Mode

The equation to calculate the baud rate in asynchronous mode is:

$$BR = \frac{CLK}{(O \times N \times P)} \tag{7}$$

where BR is the baud rate, CLK is the PCLK Clock, O is the oversample rate, N is the baud rate divisor + 1, and P is the prescaler divisor selected by the UPSR register.

Assuming an PCLK Clock of 5 MHz, a desired baud rate of 9600, and an oversample rate of 16, the N × P term according to the equation above is:

$$N \times P = \frac{(5 \times 10^6)}{(16 \times 9600)} = 32.552 \tag{8}$$

The N × P term is then divided by each Prescaler Factor from Table 27-1 to obtain a value closest to an integer. The factor for this example is 6.5.

$$N = \frac{32.552}{6.5} = 5.008 \quad (N = 5) \tag{9}$$

The baud rate register is programmed with a baud rate divisor of 4 (N = baud rate divisor + 1). This produces a baud clock of:

$$BR = \frac{(5 \times 10^6)}{(16 \times 5 \times 6.5)} = 9615.385$$

$$\%error = \frac{(9615.385 - 9600)}{9600} = 0.16 \tag{10}$$

Note that the percent error is much lower than would be possible without the non-integer prescaler factor. Error greater than 3% is marginal and may result in unreliable operation. See Table 27-3 through Table 27-5 for recommended baud rate programming values.

27.4.2 Synchronous Mode

Synchronous mode is only available for the UART0 module. When synchronous mode is selected and the UCKS bit is set, the UART operates from a clock received on the CKX pin. When the UCKS bit is clear, the UART uses the clock from the internal baud rate generator which is also driven on the CKX pin. When the internal baud rate generator is used, the equation for calculating the baud rate is:

$$BR = \frac{CLK}{(2 \times N \times P)} \tag{11}$$

where BR is the baud rate, CLK is the PCLK Clock, N is the value of the baud rate divisor + 1, and P is the prescaler divide factor selected by the value in the UnPSR register. Oversampling is not used in synchronous mode.

Use the same procedure to determine the values of N and P as in the asynchronous mode. In this case, however, only integer prescaler values are allowed.

Table 27-3. Baud Rate Programming

Baud Rate	PCLK = 48 MHz				PCLK = 45 MHz				PCLK = 30 MHz				PCLK = 24 MHz			
	O	N	P	%err	O	N	P	%err	O	N	P	%err	O	N	P	%err
300	16	2000	5	0	16	1875	5	0	16	6250	1	0	16	2000	2.5	0
600	16	2000	2.5	0	16	1875	2.5	0	16	3125	1	0	16	1250	2	0
1200	16	1250	2	0	8	3125	1.5	0	8	3125	1	0	16	1250	1	0
1800	7	401	9.5	0	8	3125	1	0	13	1282	1	0	8	1111	1.5	0.01
2000	16	1500	1	0	8	1875	1.5	0	8	1875	1	0	16	750	1	0
2400	16	1250	1	0	10	1875	1	0	10	1250	1	0	16	625	1	0
3600	8	1111	1.5	0.01	8	625	2.5	0	13	641	1	0	12	101	5.5	0.01
4800	16	625	1	0	10	625	1.5	0	10	625	1	0	16	125	2.5	0
7200	12	101	5.5	0.01	10	625	1	0	9	463	1	0.01	11	303	1	0.01
9600	16	125	2.5	0	15	125	2.5	0	10	125	2.5	0	10	250	1	0
14400	11	202	1.5	0.01	10	125	2.5	0	7	35	8.5	0.04	11	101	1.5	0.01
19200	10	250	1	0	8	293	1	0.01	11	142	1	0.03	10	125	1	0
38400	10	125	1	0	11	71	1.5	0.03	11	71	1	0.03	10	25	2.5	0
56000	7	49	2.5	0.04	12	67	1	0.05	9	7	8.5	0.04	13	33	1	0.1
57600	14	7	8.5	0.04	11	71	1	0.03	16	5	6.5	0.16	7	7	8.5	0.04
115200	7	17	3.5	0.04	10	3	13	0.16	8	5	6.5	0.16	13	16	1	0.16
128000	15	25	1	0	8	4	11	0.12	9	2	13	0.16	15	5	2.5	0
230400	13	16	1	0.16	13	15	1	0.16	10	1	13	0.16	13	8	1	0.16
345600	9	1	15.5	0.44	13	10	1	0.16	7	1	12.5	0.79	10	7	1	0.79
460800	13	8	1	0.16	13	1	7.5	0.16	10	1	6.5	0.16	13	4	1	0.16
576000	8	7	1.5	0.79	13	6	1	0.16	8	1	6.5	0.16	12	1	3.5	0.79
691200	10	7	1	0.79	13	5	1	0.16	8	1	5.5	1.36	10	1	3.5	0.79
806400	7	1	8.5	0.04	8	7	1	0.35	15	1	2.5	0.79	15	2	1	0.79
921600	13	4	1	0.16	7	7	1	0.35	13	1	2.5	0.16	13	2	1	0.16
1105920	11	4	1	1.36	9	3	1.5	0.47	9	1	3	0.47	11	2	1	1.36
1382400	10	1	3.5	0.79	13	1	2.5	0.16	11	1	2	1.36	7	1	2.5	0.79
1536000	9	1	3.5	0.79	15	2	1	2.34	13	1	1.5	0.16	8	2	1	2.34
1843200	13	2	1	0.16	7	1	3.5	0.35	11	1	1.5	1.36				
2211840	11	2	1	1.36	10	2	1	1.73	9	1	1.5	0.47				
2764800	7	1	2.5	0.79	8	2	1	1.73								
3072000	8	1	2	2.34	10	1	1.5	2.34								

Table 27-4. Baud Rate Programming

Baud Rate	PCLK = 12 MHz				PCLK = 10 MHz				PCLK = 8 MHz				PCLK = 6 MHz			
	O	N	P	%err	O	N	P	%err	O	N	P	%err	O	N	P	%err
300	16	1250	2	0	13	1282	2	0	7	401	9.5	0	16	1250	1	0
600	16	1250	1	0	13	1282	1	0	12	1111	1	0.01	16	625	1	0
1200	16	625	1	0	13	641	1	0	12	101	5.5	0.01	16	125	2.5	0
1800	12	101	5.5	0.01	12	463	1	0.01	8	101	5.5	0.01	11	303	1	0.01
2000	16	250	1.5	0	16	125	2.5	0	16	250	1	0	16	125	1.5	0
2400	16	125	2.5	0	9	463	1	0.01	11	303	1	0.01	10	250	1	0
3600	11	202	1.5	0.01	11	101	2.5	0.01	11	202	1	0.01	11	101	1.5	0.01
4800	10	250	1	0	7	119	2.5	0.04	11	101	1.5	0.01	10	125	1	0
7200	11	101	1.5	0.01	10	139	1	0.08	11	101	1	0.01	14	17	3.5	0.04
9600	10	125	1	0	7	149	1	0.13	14	17	3.5	0.04	10	25	2.5	0
14400	14	17	3.5	0.04	14	33	1.5	0.21	15	37	1	0.1	7	17	3.5	0.04
19200	10	25	2.5	0	16	13	2.5	0.16	7	17	3.5	0.04	16	13	1.5	0.16
38400	16	13	1.5	0.16	8	13	2.5	0.16	16	13	1	0.16	8	13	1.5	0.16
56000	13	11	1.5	0.1	7	17	1.5	0.04	13	11	1	0.1	9	12	1	0.79
57600	8	2	13	0.16	12	1	14.5	0.22	9	1	15.5	0.44	8	1	13	0.16
115200	13	8	1	0.16	7	5	2.5	0.79	10	7	1	0.79	13	4	1	0.16
128000	11	1	8.5	0.27	12	1	6.5	0.16	9	7	1	0.79	16	3	1	2.34
230400	13	4	1	0.16	11	4	1	1.36	10	1	3.5	0.79	13	2	1	0.16
345600	10	1	3.5	0.79					15	1	1.5	2.88	7	1	2.5	0.79
460800	13	2	1	0.16	11	2	1	1.36	7	1	2.5	0.79	13	1	1	0.16
576000	14	1	1.5	0.79	7	1	2.5	0.79	7	2	1	0.79	7	1	1.5	0.79
691200	7	1	2.5	0.79												
806400	10	1	1.5	0.79												
921600	13	1	1	0.16												
1105920					9	1	1	0.47								

Table 27-5. Baud Rate Programming

Baud Rate	PCLK = 5 MHz				PCLK = 4 MHz				PCLK = 3 MHz				PCLK = 2 MHz			
	O	N	P	%err	O	N	P	%err	O	N	P	%err	O	N	P	%err
300	11	202	7.5	0.01	12	202	5.5	0.01	16	250	2.5	0	12	101	5.5	0.01
600	11	101	7.5	0.01	12	101	5.5	0.01	16	125	2.5	0	11	202	1.5	0.01
1200	10	119	3.5	0.04	11	202	1.5	0.01	10	250	1	0	11	101	1.5	0.01
1800	11	101	2.5	0.01	11	202	1	0.01	11	101	1.5	0.01	11	101	1	0.01
2000	10	250	1	0	16	125	1	0	15	100	1	0	16	25	2.5	0
2400	7	119	2.5	0.04	11	101	1.5	0.01	10	125	1	0	14	17	3.5	0.04
3600	10	139	1	0.08	11	101	1	0.01	14	17	3.5	0.04	15	37	1	0.1
4800	7	149	1	0.13	14	17	3.5	0.04	10	25	2.5	0	7	17	3.5	0.04
7200	14	33	1.5	0.21	15	37	1	0.1	7	17	3.5	0.04	9	31	1	0.44
9600	16	13	2.5	0.16	7	17	3.5	0.04	16	13	1.5	0.16	16	13	1	0.16
14400	7	33	1.5	0.21	9	31	1	0.44	13	16	1	0.16	9	1	15.5	0.44
19200	8	13	2.5	0.16	16	13	1	0.16	8	13	1.5	0.16	16	1	6.5	0.16
38400	13	10	1	0.16	16	1	6.5	0.16	13	6	1	0.16	8	1	6.5	0.16
56000	15	6	1	0.79	13	1	5.5	0.1	9	6	1	0.79	9	4	1	0.79
57600	7	1	12.5	0.79	7	1	10	0.79	8	1	6.5	0.16	7	1	5	0.79
115200	11	4	1	1.36	10	1	3.5	0.79	13	2	1	0.16	7	1	2.5	0.79
128000	13	3	1	0.16	9	1	3.5	0.79	16	1	1.5	2.34	8	2	1	2.34
230400	11	2	1	1.36	7	1	2.5	0.79	13	1	1	0.16				

Baud Rate	PCLK = 1 MHz				PCLK = 500 kHz			
	O	N	P	%err	O	N	P	%err
300	11	202	1.5	0.01	11	101	1.5	0.01
600	11	101	1.5	0.01	14	17	3.5	0.04
1200	14	17	3.5	0.04	7	17	3.5	0.04
1800	15	37	1	0.1	9	31	1	0.44
2000	10	50	1	0	10	25	1	0
2400	7	17	3.5	0.04	16	13	1	0.16
3600	9	31	1	0.44	9	1	15.5	0.44
4800	16	13	1	0.16	16	1	6.5	0.16
7200	9	1	15.5	0.44	10	7	1	0.79
9600	16	1	6.5	0.16	8	1	6.5	0.16
14400	10	7	1	0.79	10	1	3.5	0.79
19200	8	1	6.5	0.16	13	2	1	0.16
38400	13	2	1	0.16	13	1	1	0.16
56000	9	2	1	0.79				
57600	7	1	2.5	0.79				

28 Dual Microwire/SPI Interfaces

Microwire/Plus is a synchronous serial communications protocol, originally implemented in National Semiconduc- tor's COPS® and HPC families of microcontrollers to mini- mize the number of connections, and therefore the cost, of communicating with peripherals.

The CP3SP33 has an Microwire/SPI interface module (MWSPI) that can communicate with all peripherals that conform to Microwire/Plus or Serial Peripheral Interface (SPI) specifications. This Microwire interface is capable of operating as either a master or slave and in 8- or 16-bit mode. Figure 101 shows a typical Microwire interface appli- cation.

Two modules are provided: module 0 on the CPU APB bus and module 1 on the shared audio peripheral APB bus. Module 1 can be accessed by either the CPU or DSP, so its interrupt and DMA request signals are routed through the ASC module.

The Microwire interface module includes the following fea- tures:

- Programmable operation as a master or slave
- Programmable shift-clock frequency (master only)
- Programmable 8- or 16-bit mode of operation
- 8- or 16-bit serial I/O data shift register
- Two modes of clocking data
- Serial clock can be low or high when idle
- 16-bit read buffer
- Busy bit, Read Buffer Full bit, and Overrun bit for polling and as interrupt sources
- Supports multiple masters
- DMA capability
- Maximum bus clock frequency of 12 MHz running from a 48 MHz PCLK Clock
- Supports very low-end slaves with the Slave Ready output
- Echo back enable/disable (slave only)

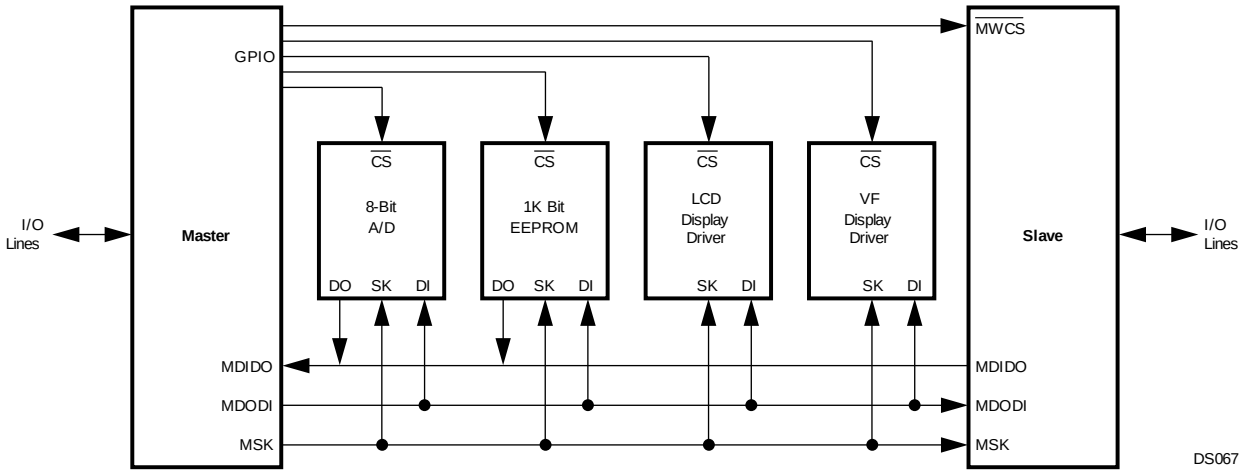


Figure 28-1. Microwire Interface

28.1 Microwire Operation

The Microwire interface allows several devices to be connected on one three-wire system. At any given time, one of these devices operates as the master while all other devices operate as slaves. The Microwire interface allows the device to operate either as a master or slave transferring 8-bits or 16-bits of data.

The master device supplies the synchronous clock (MSK) for the serial interface and initiates the data transfer. The slave devices respond by sending (or receiving) the requested data. Each slave device uses the master's clock for serially shifting data out (or in), while the master shifts the data in (or out).

The three-wire system includes: the serial data in signal (MDIDO for master mode, MDODI for slave mode), the serial data out signal (MDODI for master mode, MDIDO for slave mode), and the serial clock (MSK).

In slave mode, an optional fourth signal (MWCS) may be used to enable the slave transmit. At any given time, only one slave can respond to the master. Each slave device has its own chip select signal (MWCS) for this purpose.

Figure 28-2 shows a block diagram of the Microwire serial interface in the device.

28.1.1 Shifting

The Microwire interface is a full duplex transmitter/receiver. A 16-bit shift register, which can be split into a low and high byte, is used for both transmitting and receiving. In 8-bit mode, only the lower 8-bits are used to transfer data. The transmit data is shifted out through MDODIn pin (master mode) or MDIDOn pin (slave mode), starting with the MSB. At the same time, the receive data is shifted in through MDIDOn pin (master mode) or MDODIn pin (slave mode), also starting with the MSB first.

The shift in and shift out are controlled by the MSK clock. In each clock cycle of MSK, one bit of data is transmitted/received. The 16-bit shift register is accessible as the MWnDAT register. Reading the MWnDAT register returns the value in the receive buffer. Writing to the MWnDAT register updates the 16-bit transmit buffer.

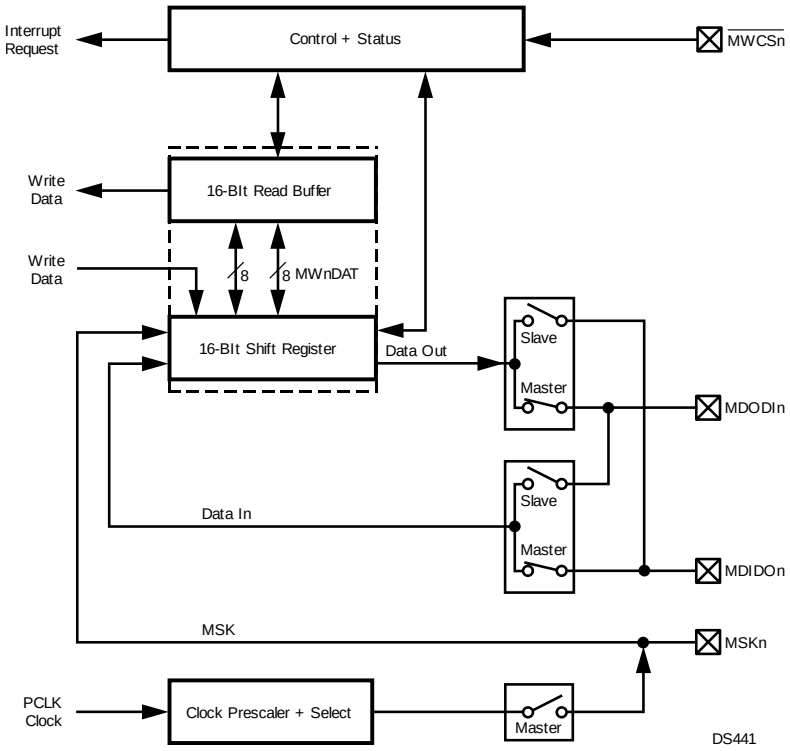


Figure 28-2. Microwire Block Diagram

28.1.2 Reading

The Microwire interface implements a double buffer on read. As illustrated in Figure 28-2, the double read buffer consists of the 16-bit shift register and a read buffer.

The 16-bit shift register loads the read buffer with new data when the data transfer sequence is completed and previous data in the read buffer has been read. In master mode, an overrun error occurs when the read buffer is full, the 16-bit shift register is full and a new data transfer sequence starts.

When 8-bit mode is selected, the lower byte of the shift register is loaded into the lower byte of the read buffer and the read buffer's higher byte remains unchanged.

The RBF bit indicates if the MWnDAT register holds valid data. The OVR bit indicates that an overrun condition has occurred.

28.1.3 Writing

The BSY bit indicates whether the MWnDAT register can be written. All write operations to the MWnDAT register update the shift register while the data contained in the read buffer is not affected. Undefined results will occur if the MWnDAT register is written while the BSY bit is set.

28.1.4 Clocking Modes

Two clocking modes are supported:

- *Normal mode*—output data transmitted on the MDODIn pin (master mode) or the MDIDOn pin (slave mode) is clocked out on the falling edge of the shift clock MSKn. Input data, which is received via the MDIDOn pin (master mode) or the MDODIn pin (slave mode), is sampled on the rising edge of MSKn.
- *Alternate mode*—output data is shifted out on the rising edge of MSKn on the MDODIn pin (master mode) or MDIDOn pin (slave mode). Input data, which is received via MDIDOn pin (master mode) or MDODIn pin (slave mode), is sampled on the falling edge of MSKn.

The clocking mode is selected with the SCM bit. The SCIDL bit allows selection of the value of MSKn when it is idle (when there is no data being transferred). In master mode, the MSKn clock frequency can be programmed in the SCDV field of the MWnCTL1 register. Figure 28-3, Figure 28-4, Figure 28-5, and Figure 28-6 show the data transfer timing for the normal and the alternate modes with the SCIDL bit clear and set.

Note that when data is shifted out on MDODIn (master mode) or MDIDOn (slave mode) on the leading edge of the MSKn clock, bit 15 (16-bit mode) is shifted out on the second leading edge of the MSKn clock. When data are shifted out on MDODIn (master mode) or MDIDOn (slave mode) on the trailing edge of MSKn, bit 15 (16-bit mode) is shifted out on the first trailing edge of MSKn.

28.2 Master Mode

In master mode, the MSKn pin is an output for the shift clock, MSKn. When data is written to the MWnDAT register, eight or sixteen MSKn clocks, depending on the mode selected, are generated to shift the 8 or 16 bits of data, and then MSKn goes idle again. The MSKn idle state can be either high or low, depending on the SCIDL bit.

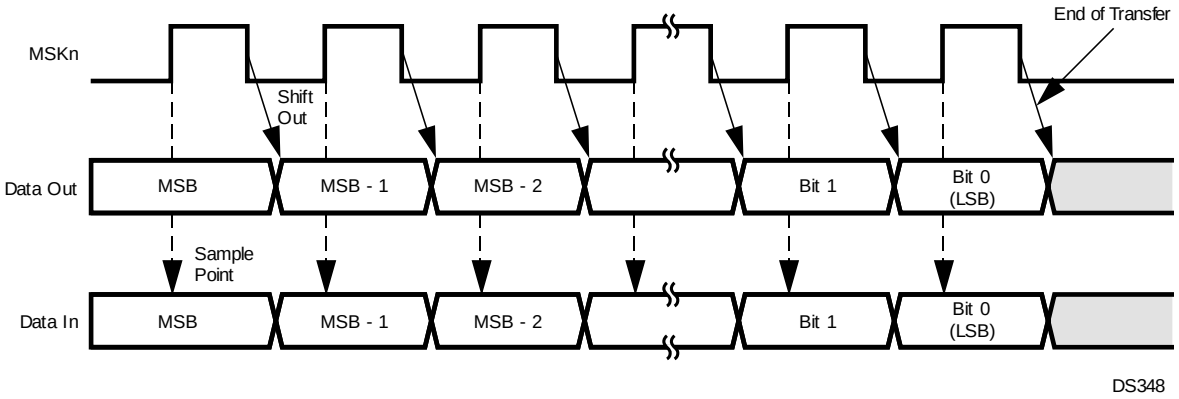


Figure 28-3. Normal Mode (SCIDL = 0)

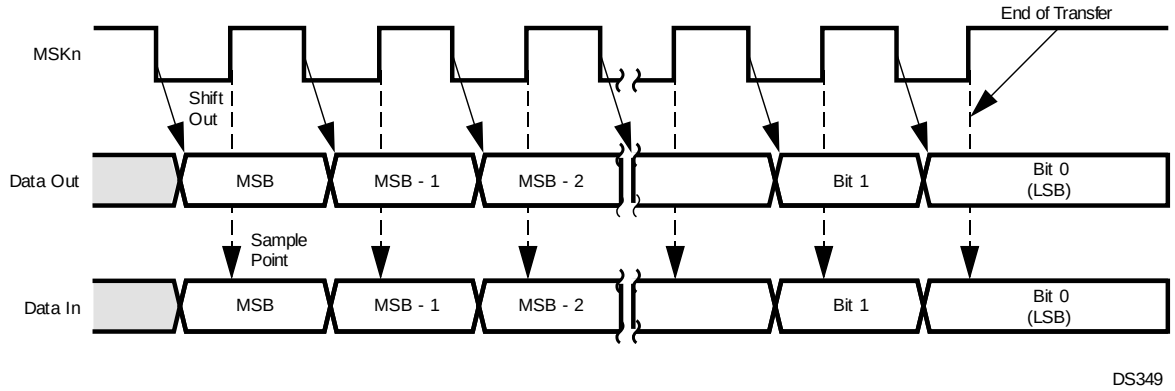


Figure 28-4. Normal Mode (SCIDL = 1)

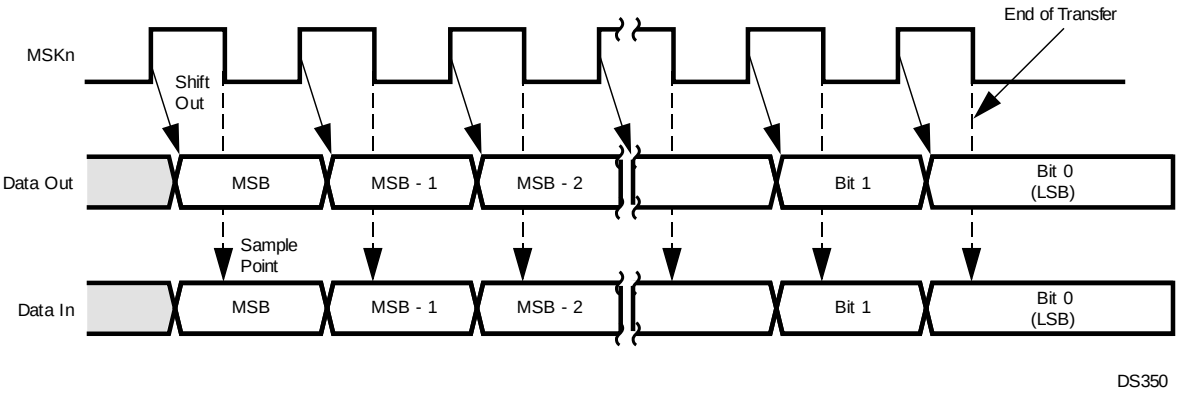


Figure 28-5. Alternate Mode (SCIDL = 0)

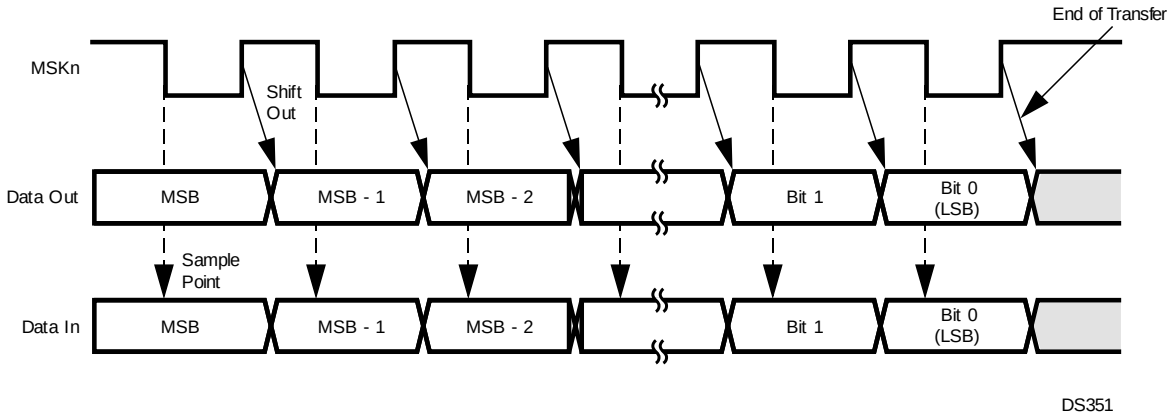


Figure 28-6. Alternate Mode (SCIDL = 1)

28.3 Slave Mode

In slave mode, the MSKn pin is an input for the shift clock MSKn. MDIDOn is put in TRI-STATE mode when $\overline{\text{MWCSn}}$ is inactive. Data transfer is enabled when $\overline{\text{MWCSn}}$ is active.

The slave starts driving MDIDOn when $\overline{\text{MWCSn}}$ is active. The most significant bit (lower byte in 8-bit mode or upper byte in 16-bit mode) is output onto the MDIDOn pin first. After eight or sixteen clocks (depending on the selected mode), the data transfer is completed.

If a new shift process starts before MWnDAT was written, i.e., while MWnDAT does not contain any valid data, and the ECHO bit is set, the data received from MDODIn is transmitted on MDIDOn in addition to being shifted to MWnDAT. If the ECHO bit is clear, the data transmitted on MDIDOn is the data held in the MWnDAT register, regardless of its validity. The master may negate the $\overline{\text{MWCSn}}$ signal to synchronize the bit count between the master and the slave. If the slave is the only slave in the system, $\overline{\text{MWCSn}}$ can be tied to ground.

28.4 Interrupt Support

Interrupts may be enabled for any of the conditions shown in Table 28-1.

Table 28-1. Microwire Interrupt Trigger Condition

Condition	Status Bit in the MWnSTAT Register	Interrupt Enable Bit in the MWnCTRL1 Register	Description
Not Busy	BSY	EIW	The shift register is ready for the next data transfer sequence.
Read Buffer Full	RBF	EIR	The read buffer is full and waiting to be unloaded.
Overrun	OVF	EIO	A new data transfer sequence started while both the shift register and the read buffer were full.

Figure 28-7 illustrates the interrupt generation logic of this module.

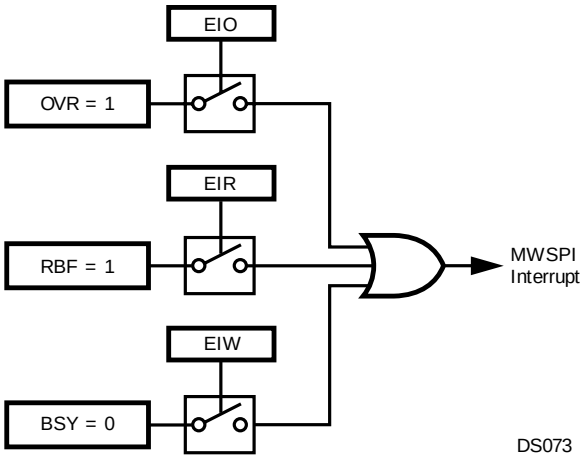


Figure 28-7. MWSPI Interrupts

Module 0 uses IRQ29, and module 1 uses IRQ10.

Because the interrupt request for module 1 can be handled by either the CPU or DSP interrupt controllers, the interrupt request is passed through the Audio Subsystem Controller (ASC), which is programmed to route it to the interrupt controller. To use the interrupt, it must be enabled at three levels:

- *Microwire module*—the interrupt must be enabled in the MW1CTL1 register.
- *ASC module*—the interrupt is assigned to ASC interrupt channel 0, so bit 0 in the ASCINTSEL register must be clear to select the CPU interrupt controller or set to select the DSP interrupt controller.
- *Interrupt controller*—the interrupt request must be enabled in the interrupt controller. For the CPU interrupt controller, this is interrupt request IRQ10.

28.5 DMA Support

The Microwire module may be operated with either one or two DMA channels. Two DMA channels are required for processor-independent full-duplex operation. Both receive and transmit DMA can be enabled individually. If transmit DMA is enabled (EDW = 1), a DMA request is asserted every time the BSY flag is cleared. Enabling receive DMA (EDR = 1) asserts a DMA request every time the Receive Buffer Full flag (RBF) is set. The Enable Interrupt on Read (EIR) bit must be clear when the EDR bit is set to avoid an interrupt request while using receive DMA. The Enable Interrupt on Write (EIW) bit must be clear when the EDW bit is set. However, a data overrun condition may occur, so the Enable Interrupt on Overrun (EIO) bit should be set.

Module 0 uses DMA requests 33 (read) and 34 (write), and module 1 uses DMA requests 18 (read) and 19 (write).

Because module 1 DMA requests can be handled by either the CPU or DSP DMA controllers, the DMA requests are passed through the Audio Subsystem Controller (ASC), which is programmed to route them to the DMA controller. To use the DMA requests, they must be enabled at three levels:

- *Microwire module*—the DMA requests must be enabled using the EDR and EDW bits in the MW1CTL2 register.
- *ASC module*—the DMA requests are assigned to ASC DMA channels 18 (read) and 19 (write). These channels are controlled by bits 2 and 3 in the ASCDMASEL1 register. The bits must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be enabled in the ASCDDMASELn registers.
- *DMA controller*—the DMA requests must be enabled in the DMA controller. For the CPU DMA controller, these are DMA requests 18 (read) and 19 (write).

28.6 Freeze Mode

While Freeze mode is asserted, the module will exhibit the following behavior:

In master mode, the transaction will be stalled at the bit that which is being currently transferred. The MSKn and MDOD- In signals will retain the current level. Once Freeze mode is de-asserted, the module will complete the current transaction.

In slave mode, the module will continue to operate normally. Data will be shifted at the rate of the MSK clock.

Reading the MWnDAT register will not cause the RBF flag to be cleared, and it will not update the read buffer if new data is ready into the shift register.

28.7 Microwire Interface Registers

Software interacts with the Microwire interface by accessing the Microwire registers. There are three such registers:

Table 28-2. Microwire Interface Registers

NAME	ADDRESS	DESCRIPTION
MW0CTL1	FF 8404h	Microwire Module 0 Control Register 1
MW1CTL1	FF 5804h	Microwire Module 1 Control Register 1
MW0CTL2	FF 840Ah	Microwire Module 0 Control Register 2
MW1CTL2	FF 580Ah	Microwire Module 1 Control Register 2
MW0STAT	FF 8408h	Microwire Module 0 Status Register
MW1STAT	FF 5808h	Microwire Module 1 Status Register
MW0DAT	FF 8400h	Microwire Module 0 Data Register
MW1DAT	FF 5800h	Microwire Module 1 Data Register

28.7.1 Microwire Module n Control Register 1 (MWnCTL1)

The MWnCTL1 registers are 16-bit, read/write registers used to control the Microwire module. To avoid clock glitches, the MWEN bit must be clear while changing the states of any other bits in the register. At reset, all non-reserved bits are cleared. The register format is shown below.

15 9	8	7	6	5	4	3	2	1	0
SCDV	SCIDL	SCM	EIW	EIR	EIO	ECHO	MOD	MNS	MWEN

MWEN MWEN The Microwire Enable bit controls whether the Microwire interface module is enabled.
0 – Microwire module disabled.
1 – Microwire module enabled.
Clearing this bit disables the module, clears the status bits in the Microwire status register (the BSY, RBF, and OVR bits in MWnSTAT), and places the Microwire interface pins in the states described below.

PIN	STATE WHEN DISABLED
MSKn	Master – SCIDL Bit Slave – Input
MDIDOn	Master – Input Slave – TRI-STATE
MDODIn	Master – Known value Slave – Input

MNS The Master/Slave Select bit controls whether the CP3SP33 is a master or slave. When clear, the device operates as a slave. When set, the device operates as the master.
0 – CP3SP33 is slave.
1 – CP3SP33 is master.

MOD The Mode Select bit controls whether 8- or 16-bit mode is used. When clear, the device operates in 8-bit mode. When set, the device operates in 16-bit mode. This bit must only be changed when the module is disabled or idle (MWnSTAT.BSY = 0).
0 – 8-bit mode.
1 – 16-bit mode.

ECHO The Echo Back bit controls whether the echo back function is enabled in slave mode. This bit must be written only when the Microwire interface is idle (MWnSTAT.BSY=0). The ECHO bit is ignored in master mode. The MWnDAT register is valid from the time the register has been written until the end of the transfer. In the echo back mode, MDODIn is transmitted (echoed back) on MDIDOn if the MWnDAT register does not contain any valid data. With the echo back function disabled, the data held in the MWnDAT register is transmitted on MDIDOn, whether or not the data is valid.
0 – Echo back disabled.
1 – Echo back enabled.

EIO The Enable Interrupt on Overrun bit enables or disables the overrun interrupt. When set, an interrupt is asserted when the Receive Overrun bit (MWnSTAT.OVR) is set. Otherwise, no interrupt is asserted when an overrun occurs. This bit must only be enabled in master mode.
0 – Disable overrun interrupts.
1 – Enable overrun interrupts.

EIR The Enable Interrupt for Read bit controls whether an interrupt is asserted when the read buffer becomes full. When set, an interrupt is asserted when the Read Buffer Full bit (MWnSTAT.RBF) is set. Otherwise, no interrupt is asserted when the read buffer is full.
0 – No read buffer full interrupt.
1 – Interrupt when read buffer becomes full.

EIW The Enable Interrupt for Write bit controls whether an interrupt is asserted when the Busy bit (MWnSTAT.BSY) is cleared, which indicates that a data transfer sequence has been completed and the read buffer is ready to receive the new data. Otherwise, no interrupt is asserted when the Busy bit is cleared.
0 – No interrupt on data transfer complete.
1 – Interrupt on data transfer complete.

SCM The Shift Clock Mode bit selects between the normal clocking mode and the alternate clocking mode. In the normal mode, the output data is clocked out on the falling edge of MSKn and the input data is sampled on the rising edge of MSKn. In the alternate mode, the output data is clocked out on the rising edge of MSKn and the input data is sampled on the falling edge of MSKn.
0 – Normal clocking mode.
1 – Alternate clocking mode.

SCIDL The Shift Clock Idle bit controls the value of the MSKn output when the Microwire module is idle. This bit must be changed only when the Microwire module is disabled (MEN = 0) or when no bus transaction is in progress (MWnSTAT.BSY = 0).
0 – MSKn is low when idle.
1 – MSKn is high when idle

SCDV The Shift Clock Divider Value field specifies the divisor used for generating the MSKn shift clock from the PCLK Clock. The divisor is 2 × (SCDV6:0 + 1). Valid values are 0000001b to 1111111b, so the division ratio may range from 4 to 256. This field is ignored in slave mode (MWnCTL1.MNS = 0).

28.7.2 Microwire Module n Control Register 2 (MWnCTL2)

The MWnCTL2 registers are 16-bit, read/write registers used to enable DMA requests. At reset, all non-reserved bits are cleared. The register format is shown below.

7	2	1	0
Reserved		EDW	EDR

EDR The Enable DMA Read bit controls whether a DMA request is enabled when the RBF bit in the MWnSTAT register is set.
0 – DMA request disabled.
1 – DMA request enabled.

EDW The Enable DMA Read bit controls whether a DMA request is enabled when the BSY bit in the MWnSTAT register is cleared.
0 – DMA request disabled.
1 – DMA request enabled.

28.7.3 Microwire Module n Status Register (MWnSTAT)

The MWnSTAT registers are 16-bit, read-only registers that shows the current status of the Microwire interface module. At reset, all non-reserved bits are clear. The register format is shown below.

15	3	2	1	0
Reserved		OVR	RBF	BSY

BSY The Busy bit, when set, indicates that the Microwire shift register is busy. In master mode, the BSY bit is set when the MWnDAT register is written. In slave mode, the bit is set on the first leading edge of MSKn when MWCSn is asserted or when the MWnDAT register is written, whichever occurs first. In both master and slave modes, this bit is cleared when the Microwire data transfer sequence is completed and the read buffer is ready to receive the new data; in other words, when the previous data held in the read buffer has already been read. If the previous data in the read buffer has not been read and new data has been received into the shift register, the BSY bit will not be cleared, as the transfer could not be completed because the contents of the shift register could not be transferred into the read buffer.
0 – Shift register is not busy.
1 – Shift register is busy.

RBF The Read Buffer Full bit, when set, indicates that the read buffer is full and ready to be read by software. It is set when the shift register loads the read buffer, which occurs upon completion of a transfer sequence if the read buffer is empty. The RBF bit is updated when the MWnDAT register is read. At that time, the RBF bit is cleared if the shift register does not contain any new data (in other words, the shift register is not receiving data or has not yet received a full byte of data). The RBF bit remains set if the shift register already holds new data at the time that MWnDAT is read. In that case, MWnDAT is immediately reloaded with the new data and is ready to be read by software.
0 – Read buffer is not full.
1 – Read buffer is full.

OVR The Receive Overrun bit, when set in master mode, indicates that a receive overrun has occurred. An overrun occurs when the read buffer is full, the 8-bit shift register is full, and a new data transfer sequence starts. This bit is undefined in slave mode. The OVR bit, once set, remains set until cleared by software. Software clears this bit by writing a 1 to its bit position. Writing a 0 to this bit position has no effect. No other bits in the MWnSTAT register are affected by writes to the register.
0 – No receive overrun has occurred.
1 – Receive overrun has occurred.

28.7.4 Microwire Module n Data Register (MWnDAT)

The MWnDAT registers are 16-bit, read/write registers used to transmit and receive data through the MDODIn and MDIDOn pins.

7								0
MWDAT								

Figure 28-8 shows the hardware structure of the register.

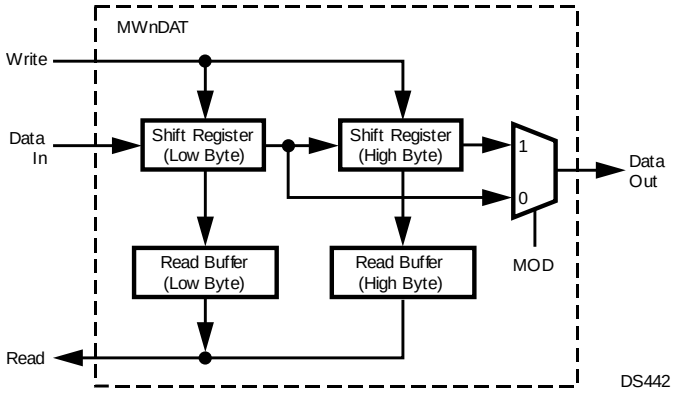


Figure 28-8. MWnDAT Register

29 Dual ACCESS.bus Interfaces

The ACCESS.bus interface modules (ACB) provide a two- wire serial interface compatible with the ACCESS.bus physical layer. It permits easy interfacing to a wide range of low-cost memories and I/O devices, including: EEPROMs, RAMs, timers, A/D converters, D/A converters, clock chips, and peripheral drivers. It is compatible with Intel's SMBus and Philips' I2C bus. The ACB modules can be configured as a bus master or slave, and can maintain bidirectional communications with both multiple master and slave devices

Two modules are provided, module 0 on the CPU APB bus and module 1 on the shared audio peripheral APB bus. Module 1 can be accessed by either the CPU or DSP, so its interrupt and DMA request signals are routed through the ASC module.

This section presents an overview of the bus protocol, and its implementation by the ACB module.

- ACCESS.bus master and slave
- Supports polling and interrupt-controlled operation
- Generate a wake-up signal on detection of a Start Condition, while in power-down mode
- Optional internal pullup on SDA and SCL pins
- Interrupt and DMA capability
- Up to 400 kHz bus clock frequency (Fast-mode)

29.1 ACCESS.bus Protocol Overview

The ACCESS.bus protocol uses a two-wire interface for bi- directional communication between the devices connected to the bus. The two interface signals are the Serial Data Line (SDA) and the Serial Clock Line (SCL). These signals should be connected to the positive supply, through pullup resistors, to keep the signals high when the bus is idle.

The ACCESS.bus protocol supports multiple master and slave transmitters and receivers. Each bus device has a unique address and can operate as a transmitter or a receiver (though some peripherals are only receivers).

During data transactions, the master device initiates the transaction, generates the clock signal, and terminates the transaction. For example, when the ACB initiates a data transaction with an ACCESS.bus peripheral, the ACB becomes the master. When the peripheral responds and transmits data to the ACB, their master/slave (data transaction initiator and clock generator) relationship is unchanged, even though their transmitter/receiver functions are reversed.

29.1.1 Data Transactions

One data bit is transferred during each clock period. Data is sampled during the high phase of the serial clock (SCL). Consequently, throughout the clock high phase, the data must remain stable (see Figure 29-1). Any change on the SDA signal during the high phase of the SCL clock and in the middle of a transaction aborts the current transaction. New data must be driven during the low phase of the SCL clock. This protocol permits a single data line to transfer both command/control information and data using the synchronous serial clock.

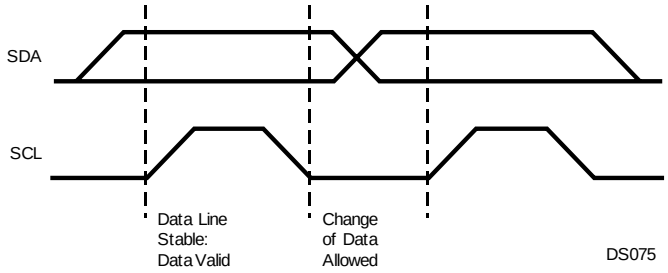


Figure 29-1. Bit Transfer

Each data transaction is composed of a Start Condition, a number of byte transfers (programmed by software), and a Stop Condition to terminate the transaction. Each byte is transferred with the most significant bit first, and after each byte, an Acknowledge signal must follow.

At each clock cycle, the slave can stall the master while it handles the previous data, or prepares new data. This can be performed for each bit transferred or on a byte boundary by the slave holding SCL low to extend the clock-low period. Typically, slaves extend the first clock cycle of a transfer if a byte read has not yet been stored, or if the next byte to be transmitted is not yet ready. Some microcontrollers with limited hardware support for ACCESS.bus extend the access after each bit, to allow software time to handle this bit.

29.1.1.1 Start and Stop

The ACCESS.bus master generates Start and Stop Conditions (control codes). After a Start Condition is generated, the bus is considered busy and it retains this status until a certain time after a Stop Condition is generated. A high-to-low transition of the data line (SDA) while the clock (SCL) is high indicates a Start Condition. A low-to-high transition of the SDA line while the SCL is high indicates a Stop Condition (see Figure 29-2).

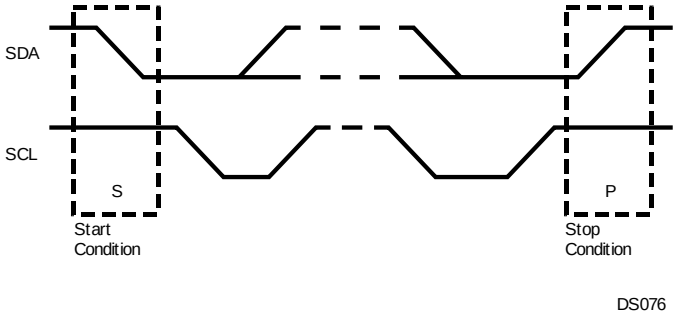


Figure 29-2. Start and Stop Conditions

In addition to the first Start Condition, a repeated Start Condition can be generated in the middle of a transaction. This allows another device to be accessed, or a change in the direction of the data transfer.

29.1.1.2 Acknowledge Cycle

The Acknowledge Cycle consists of two signals: the ac- knowledge clock pulse the master sends with each byte transferred, and the acknowledge signal sent by the receiving device (Figure 29-3).

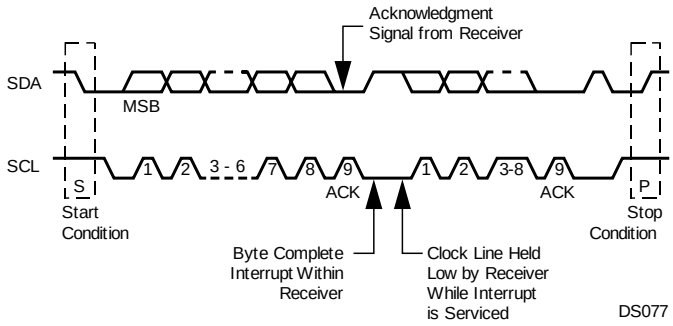


Figure 29-3. ACCESS.bus Data Transaction

The master generates the acknowledge clock pulse on the ninth clock pulse of the byte transfer. The transmitter releases the SDA line (permits it to go high) to allow the receiver to send the acknowledge signal. The receiver must pull down the SDA line during the acknowledge clock pulse, which signals the correct reception of the last data byte, and its readiness to receive the next byte. Figure 29-4 illustrates the acknowledge cycle.

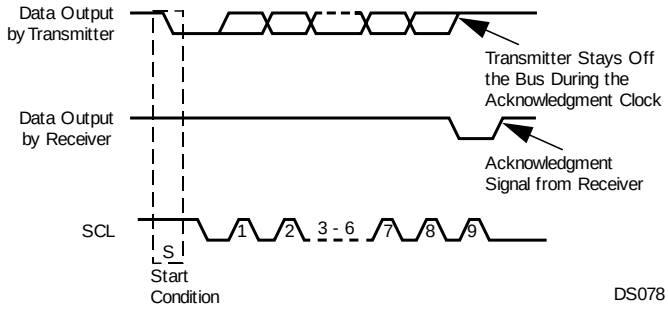


Figure 29-4. ACCESS.bus Acknowledge Cycle

The master generates an acknowledge clock pulse after each byte transfer. The receiver sends an acknowledge signal after every byte received. There are two exceptions to the “acknowledge after every byte” rule.

- When the master is the receiver, it must indicate to the transmitter an end-of-data condition by not-acknowledging (“negative acknowledge”) the last byte clocked out of the slave. This “negative acknowledge” still includes the acknowledge clock pulse (generated by the master), but the SDA line is not pulled down.
- When the receiver is full, otherwise occupied, or a problem has occurred, it sends a negative acknowledge to indicate that it cannot accept additional data bytes.

29.1.1.3 Addressing Transfer Formats

Each device on the bus has a unique address. Before any data is transmitted, the master transmits the address of the slave being addressed. The slave device should send an acknowledge signal on the SDA signal, once it recognizes its address.

The address is the first seven bits after a Start Condition. The direction of the data transfer (R/W) depends on the bit sent after the address (the eighth bit). A low-to-high transition during a SCL high period indicates the Stop Condition, and ends the transaction (Figure 29-5).

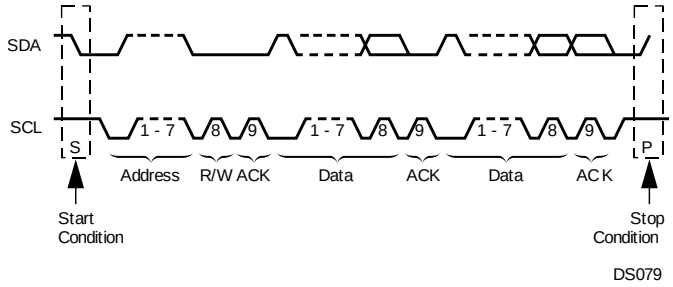


Figure 29-5. A Complete ACCESS.bus Data Transaction

When the address is sent, each device in the system compares this address with its own. If there is a match, the device considers itself addressed and sends an acknowledge signal. Depending upon the state of the R/W bit (1 = read, 0 = write), the device acts as a transmitter or a receiver.

The ACCESS.bus protocol allows sending a general call address to all slaves connected to the bus. The first byte sent specifies the general call address (00h) and the second byte specifies the meaning of the general call (for example, “Write slave address by software only”). Those slaves that require the data acknowledge the call and become slave receivers; the other slaves ignore the call.

29.1.1.4 Arbitration on the Bus

Arbitration is required when multiple master devices attempt to gain control of the bus simultaneously. Control of the bus is initially determined according to address bits and clock cycle. If the masters are trying to address the same bus device, data comparisons determine the outcome of this arbitration. In master mode, the device immediately aborts a transaction if the value sampled on the SDA lines differs from the value driven by the device. (Exceptions to this rule are SDA while receiving data; in these cases the lines may be driven low by the slave without causing an abort.)

The SCL signal is monitored for clock synchronization and allows the slave to stall the bus. The actual clock period will be the one set by the master with the longest clock period or by the slave stall period. The clock high period is determined by the master with the shortest clock high period.

When an abort occurs during the address transmission, the master that identifies the conflict should give up the bus, switch to slave mode, and continue to sample SDA to see if it is being addressed by the winning master on the ACCESS.bus.

29.2 ACB Functional Description

The ACB modules provides the physical layer for ACCESS.bus-compliant serial interfaces. The modules are configurable as either master or slave devices. As slaves, the ACB modules may issue a request to become the bus master.

29.2.1 Master Mode

An ACCESS.bus transaction starts with a master device requesting bus mastership. It sends a Start Condition, followed by the address of the device it wants to access. If this transaction is successfully completed, software can assume that the device has become the bus master.

For a device to become the bus master, software should perform the following steps:

1. Set the ACBnCTL1.START bit, and configure the ACBnCTL1.INTEN bit to the desired operation mode (Polling or Interrupt). This causes the ACB to issue a Start Condition on the ACCESS.bus, as soon as the ACCESS.bus is free (ACBnCST.BB=0). It then stalls the bus by holding SCLn low.
2. If a bus conflict is detected, (i.e., some other device pulls down the SCLn signal before this device

does), the ACBnST.BER bit is set.

3. If there is no bus conflict, the ACBnST.MASTER and ACBnST.SDAST bits are set.
4. If the ACBnCTL1.INTEN bit is set, and either the ACBnST.BER bit or the ACBnST.SDAST bit is set, an interrupt is asserted.

29.2.1.1 Sending the Address Byte

Once this device is the active master of the ACCESS.bus (ACBnST.MASTER = 1), it can send the address on the bus. The address must not be this device's own address as specified in the ACBnADDR.ADDR field if the ACBnAD- DR.SAEN bit is set or the ACBnADDR2.ADDR field if the ACBnADDR2.SAEN bit is set. The address also must not be the global call address if the ACBnST.GCMTCH bit is set or the ARP address if the ACBnST.ARPMATCH bit is set.

To send the address byte, use the following sequence:

1. Load the ACBnCTL1.INTEN and ACBnCTL1.DMAEN bits for the desired operation mode. For a receive transaction where software wants only one byte of data, it should set the ACBnCTL1.ACK bit. If only an address needs to be sent, set the ACBnCTL1.STASTRE bit.
2. Write the address byte (7-bit target device address), and the direction bit, to the ACBnSDA register. This causes the module to generate a transaction. At the end of this transaction, the acknowledge bit received is copied to the ACBnST.NEGACK bit. During the transaction, the SDAn and SCLn signals are continuously checked for conflicts with other devices. If a conflict is detected, the transaction is aborted, the ACBnST.BER bit is set, and the ACBnST.MASTER bit is cleared.

3. If the ACBnCTL1.STASTRE bit is set, and the transaction was successfully completed (i.e., both the ACBnST.BER and ACBnST.NEGACK bits are cleared), the ACBnST.STASTR bit is set. In this case, the ACB stalls any further ACCESS.bus operations (i.e., holds SCLn low). If the ACBnCTL1.INTE bit is set, an interrupt is asserted.
4. If the requested direction is transmit, and the start transaction was completed successfully (i.e., neither the ACBnST.NEGACK nor ACBnST.BER bit is set, and no other master has accessed the device), the ACBnST.SDAST bit is set to indicate that the module is waiting for service.
5. If the requested direction is receive, the start transaction was completed successfully, and the ACBnCTL1.STASTRE bit is clear, the module starts receiving the first byte automatically.
6. Check that both the ACBnST.BER and ACBnST.NEGACK bits are clear. If either the ACBnCTL1.INTEN or ACBnCTL1.DMAEN bit is set, an interrupt is asserted when either the ACBnST.BER or ACBnST.NEGACK bit is set.

29.2.1.2 Master Transmit

After becoming the bus master, the device can start transmitting data on the ACCESS.bus. To transmit a byte using interrupts or polling, software must:

1. Check that the BER and NEGACK bits in the ACBnST register are clear and the ACBnST.SDAST bit is set. Also, if the ACBnCTL1.STASTRE bit is set, check that the ACBnST.STASTR bit is clear.
2. Write the data byte to be transmitted to the ACBnSDA register.

To transmit a byte using DMA, software must:

1. If the ACBnCTL1.DMAEN bit was set before the start transaction, a DMA request is generated automatically at the end of the address transaction and each following transaction unless for some reason (e.g., ACBCST, MATCH, or BER were set) an interrupt was asserted
2. When the ACBnST.NEGACK or ACBnST.BER bits are set, an interrupt is asserted, and the module stops sending DMA requests.

When the slave responds with a negative acknowledge, the ACBnST.NEGACK bit is set and the ACBnST.SDAST bit re- mains clear. In this case, if the ACBnCTL1.INTEN or ACBnCTL1.DMAEN bit is set, an interrupt is asserted.

29.2.1.3 Master Receive

After becoming the bus master, the device can start receiving data on the ACCESS.bus. To receive a byte using interrupts or polling, software must:

1. Check that the ACBnST.SDAST bit is set and the ACBnST.BER bit is clear. Also, if the ACBnCTL1.STASTRE bit is set, check that the ACBnST.STASTR bit is clear.
2. Set the ACBnCTL1.ACK bit, if the next byte is the last byte that should be read. This causes a negative acknowledge to be sent.
3. Read the data byte from the ACBnSDA register.

To receive a byte using DMA, software must:

1. The DMA request becomes active after the module receives a byte of data. If an error occurs during the transaction (e.g., ACBCST.NMATCH or ACBST.BER is set), an interrupt is asserted and the DMA operation is stalled.
2. Before receiving the last byte of data, set the ACBnCTL1.ACK bit. This should be done by programming the DMA to interrupt the CPU one byte before the end of the transmission and letting the software set the ACBnCTL1.ACK bit.

29.2.1.4 Master Stop

A Stop Condition may be issued only when this device is the active bus master (ACBnST.MASTRER = 1). To end a transaction, set the ACBnCTL1.STOP bit before clearing the current stall bit (i.e., the ACBnST.SDAST, ACBnST.NEGACK, or ACBnST.STASTR bit). This causes the module to send a Stop Condition immediately, and clear the ACBnCTL1.STOP bit.

29.2.1.5 Master Bus Stall

The ACB module can stall the ACCESS.bus between trans- fers while waiting for the APB bus master (CPU, DSP, or DMA controller) response. The ACCESS.bus is stalled by holding the SCLn signal low after the acknowledge cycle. Note that this is interpreted as the beginning of the following bus operation. Software must make sure that the next operation is prepared before the bit that causes the bus stall is cleared.

The bits that can cause a stall in master mode are:

- Negative acknowledge after sending a byte (ACBnST.NEGACK = 1).
- ACBnST.SDAST bit is set.
- If the ACBnCTL1.STASTRE bit is set, after a successful start (ACBnST.STASTR = 1).

29.2.1.6 Repeated Start

A repeated start is performed when this device is already the bus master (ACBnST.MASTER = 1). In this case, the ACCESS.bus is stalled and the ACB waits for software intervention to handle the condition: negative acknowledge (ACBnST.NEGACK = 1), empty buffer (ACBnST.SDAST = 1), or a stop-after-start (ACBnST.STASTR = 1).

For a repeated start:

1. Set the ACBnCTL1.START bit.
2. In master receive mode, read the last data item from the ACBnSDA register.
3. Follow the address send sequence, as described in “Sending the Address Byte” on page 225
4. If the ACB was waiting for handling due to ACBnST.STASTR = 1, clear it only after writing the requested address and direction to the ACBnSDA register.

29.2.1.7 Master Error Detections

The ACB detects illegal Start or Stop Conditions (i.e., a Start or Stop Condition within the data transfer or the acknowledge cycle) and a conflict on the data lines of the ACCESS.bus. If an illegal action is detected, the BER bit is set, and the MASTER mode is exited (the MASTER bit is cleared).

29.2.1.8 Bus Idle Error Recovery

When a request to become the active bus master or a restart operation fails, the ACBnST.BER bit is set to indicate the error. In some cases, both this device and the other device may identify the failure and leave the bus idle. In this case, the start sequence may not be completed and the ACCESS.bus may remain deadlocked.

To recover from deadlock, use the following sequence:

1. Clear the ACBnST.BER and ACBnCST.BB bits.
2. Wait for a timeout period to check that there is no other active master on the bus (i.e., the ACBnCST.BB bit remains clear).
3. Disable, and re-enable the ACB to put it in the non-ad- dressed slave mode.
4. At this point, some of the slaves may not identify the bus error. To recover, the ACB becomes the bus master by issuing a Start Condition and sends an address field. Then, it issues a Stop Condition to synchronize all the slaves.

29.2.2 Slave Node

A slave device waits in Idle mode for a master to initiate a bus transaction. Whenever the ACB is enabled with a clear ACBnST.MASTER bit, it acts as a slave device.

Once a Start Condition on the bus is detected, this device checks whether the address sent by the current master matches either:

- ACBnADDR.ADDR, if the ACBnADDR.SAEN bit is set.
- ACBnADDR2.ADDR, if the ACBnADDR2.SAEN bit is set.
- The general call address, if the ACBnCTL1.GCM bit is set.
- The global ARP address, if the ACBnCTL3.ARPMEN bit is set.

This match is checked even when the ACBnST.MASTER bit is set. If a bus conflict (on SDAn or SCLn) is detected, the ACBnST.BER bit is set, the ACBnST.MASTER bit is cleared, and this device continues to search the received message for a match. If an address match, ARP match, or a global match, is detected:

1. This device asserts its data pin during the acknowledge cycle.
2. The ACBnCST.MATCH, ACBnCST.MATCHAF (or ACBnCST.GCMTCH if it is a global call address match, or ACBnCST.ARPMATCH if it is an ARP address), and ACBnST.NMATCH in the ACBnCST register are set. If the ACBnST.XMIT bit is set (i.e., slave transmit mode), the ACBnST.SDAST bit is set to indicate that the buffer is empty.
3. If the ACBnCTL1.INTEN bit is set, an interrupt is asserted if both the INTEN and NMINTE bits in the ACBnCTL1 register are set.
4. Software then reads the ACBnST.XMIT bit to identify the direction requested by the master device. It clears the ACBnST.NMATCH bit so future byte transfers are identified as data bytes.

29.2.2.1 Slave Receive and Transmit

Slave Receive and Transmit are performed after a match is detected and the data transfer direction is identified. After a byte transfer, the ACB extends the acknowledge clock until software reads or writes the ACBnSDA register. The receive and transmit sequence are identical to those used in the master routine.

29.2.2.2 Slave Bus Stall

When operating as a slave, this device stalls the ACCESS.bus by extending the first clock cycle of a transaction in the following cases:

1. The ACBnST.SDAST bit is set.
2. The ACBnST.NMATCH, and ACBnCTL1.NMINTE bits are set.

29.2.2.3 Slave Error Detections

The ACB detects illegal Start and Stop Conditions on the ACCESS.bus (i.e., a Start or Stop Condition within the data transfer or the acknowledge cycle). When an illegal Start or Stop Condition is detected, the BER bit is set and the MATCH and GMATCH bits are cleared, causing the module to be an unaddressed slave.

29.2.2.4 Power Down

When this device is in Power Save, Idle, or Halt mode, the ACB module is not active but retains its status. If the ACB is enabled (ACBnCTL2.ENABLE = 1) on detection of a Start Condition, a wake-up signal is issued to the MIWU module. Use this signal to switch this device to Active mode.

The ACB module cannot check the address byte for a match following the start condition that caused the wake-up event for this device. The ACB responds with a negative acknowledge, and the device should resend both the Start Condition and the address after this device has had time to wake up.

Check that the ACBnCST.BUSY bit is inactive before entering Power Save, Idle, or Halt mode. This assures that the device does not acknowledge an address sent and stop responding later.

29.2.3 SDAn nand SCLn Pin Configuration

The SDA and SCL pins are driven as open-drain signals. Internal pullups may be enabled for either pin. For more information, see the I/O configuration section.

29.2.4 ACB Clock Frequency Configuration

The ACB module permits software to select the frequency used for the ACCESS.bus clock. The clock period is set by the ACBCTL2.SCLFRQ6:0 and ACBCTL3.SCLFRQ8:7 fields. Together, they form a 9-bit value that specifies the SCL clock period. This clock low period may be extended by stall periods initiated by the ACB module or by another ACCESS.bus device. In case of a conflict with another bus master, a shorter clock high period may be forced by the other bus master until the conflict is resolved.

29.3 Interrupt Support

ACB module 0 uses IRQ23, and module 1 uses IRQ11.

Because the interrupt request for module 1 can be handled by either the CPU or DSP interrupt controllers, the interrupt request is passed through the Audio Subsystem Controller (ASC), which is programmed to route it to the interrupt controller. To use the interrupt, it must be enabled at three levels:

- *ACB module*—the interrupt must be enabled in the ACBnCTL1 register.
- *ASC module*—the interrupt is assigned to ASC interrupt channel 1, so bit 1 in the ASCINTSEL register must be clear to select the CPU interrupt controller or set to select the DSP interrupt controller.
- *Interrupt controller*—the interrupt request must be enabled in the interrupt controller. For the CPU interrupt controller, this is interrupt request IRQ11.

29.4 SMA Support

ACB module 0 uses DMA request 35, and module 1 uses DMA request 20.

Because the module 1 DMA request can be handled by either the CPU or DSP DMA controllers, the DMA request is passed through the Audio Subsystem Controller (ASC), which must be programmed to route it to the DMA controller.

To use the DMA request, it must be enabled at three levels:

- *ACB module*—the DMA request must be enabled.
- *ASC module*—the DMA request is assigned to ASC DMA channel 20. This channel is controlled by bit 4 in the ASCDMASEL1 register. The bit must be clear to select the CPU DMA controller or set to select the DSP DMA controller. If the DSP DMA controller is selected, the DSP DMA channel must be enabled in the ASCDDMASELn registers.
- *DMA controller*—the DMA request must be enabled in the DMA controller. For the CPU DMA controller, this is DMA request 20.

29.5 ACCESS.bus Interface Registers

The ACCESS.bus interface uses the registers listed in Table 29-1.

Table 29-1. ACCESS.bus Interface Registers

NAME	ADDRESS	DESCRIPTION
ACB0SDA	FF 8000h	ACB0 Serial Data Register
ACB1SDA	FF 5400h	ACB1 Serial Data Register
ACB0ST	FF 8004h	ACB0 Status Register
ACB1ST	FF 5404h	ACB1 Status Register

Table 29-1. ACCESS.bus Interface Registers (continued)

NAME	ADDRESS	DESCRIPTION
ACB0CST	FF 8008h	ACB0 Control Status Register
ACB1CST	FF 5408h	ACB1 Control Status Register
ACB0CTL1	FF 800Ch	ACB0 Control Register 1
ACB1CTL1	FF 540Ch	ACB1 Control Register 1
ACB0CTL2	FF 8014h	ACB0 Control Register 2
ACB1CTL2	FF 5414h	ACB1 Control Register 2
ACB0CTL3	FF 801Ch	ACB0 Control Register 3
ACB1CTL3	FF 541Ch	ACB1 Control Register 3
ACB0ADDR1	FF 8010h	ACB0 Own Address Register 1
ACB1ADDR1	FF 5410h	ACB1 Own Address Register 1
ACB0ADDR2	FF 8018h	ACB0 Own Address Register 2
ACB1ADDR2	FF 5418h	ACB1 Own Address Register 2

29.5.1 ACBn Serial Data Register (ACBnSDA)

The ACBnSDA registers are 8-bit, read/write shift registers used to transmit and receive data. The most significant bit is transmitted (received) first and the least significant bit is transmitted (received) last. Reading or writing the ACBnSDA registers is allowed when ACBnST.SDAST is set; or for repeated starts after setting the START bit. An attempt to access the register in other cases produces unpredictable results.

7							0
DATA							

29.5.2 ACBn Status Register (ACBnST)

The ACBnST registers are 8-bit, read/write registers that indicate the ACB status. The NMATCH, STASTR, NEGACK, BER, and SLVSTP bits may be cleared by writing 1 to them. Writing 0 has no effect on these bits. Following reset, Idle mode, Halt mode, and disabling the module, these registers are clear.

7	6	5	4	3	2	1	0
SLVSTP	SDAST	BER	NEGACK	STASTR	NMATCH	MASTER	XMIT

XMIT — The Direction Bit bit is set when the ACB module is currently in master/slave transmit mode. Otherwise it is cleared.
0 – Receive mode.
1 – Transmit mode.

MASTER — The Master bit indicates that the module is currently in master mode. It is set when a request for bus mastership succeeds. It is cleared upon arbitration loss (BER is set) or the recognition of a Stop Condition.
0 – Slave mode.
1 – Master mode.

NMATCH — The New match bit is set when the address byte following a Start Condition, or repeated starts, causes a match or a global-call match. The NMATCH bit is cleared when written with 1. Writing 0 to NMATCH is ignored. If the ACBnCTL1.INTEN bit is set, an interrupt is sent when this bit is set. When NMATCH is set DMA requests are not asserted.
0 – No match.
1 – Match or global-call match.

STASTR The Stall After Start bit is set by the successful completion of an address sending (i.e., a Start Condition sent without a bus error, or negative acknowledge), if the ACBnCTL1.STASTRE bit is set. This bit is ignored in slave mode. When the STASTR bit is set, it stalls the bus by pulling down the SCLn line, and suspends any other action on the bus (e.g., receives first byte in master receive mode). In addition, if the ACBnCTL1.INTEN bit is set, it also sends an interrupt to the core. Writing 1 to the STASTR bit clears it. It is also cleared when the module is disabled. Writing 0 to the STASTR bit has no effect. 0 – No stall after start condition. 1 – Stall after successful start.

NEGACK The Negative Acknowledge bit is set by hardware when a transmission is not acknowledged on the ninth clock. (In this case, the SDAST bit is not set.) Writing 1 to NEGACK clears it. It is also cleared when the module is disabled. Writing 0 to the NEGACK bit is ignored.
0 – No transmission not acknowledged condition.
1 – Transmission not acknowledged.

BER The Bus Error bit is set by the hardware when a Start or Stop Condition is detected during data transfer (i.e., Start or Stop Condition during the transfer of bits 2 through 8 and acknowledge cycle), or when an arbitration problem is detected. Writing 1 to the BER bit clears it. It is also cleared when the module is disabled. Writing 0 to the BER bit is ignored.
0 – No bus error occurred.
1 – Bus error occurred.

SDST The SDA Status bit indicates that the SDA data register is waiting for data (transmit, as master or slave) or holds data that should be read (receive, as master or slave). This bit is cleared when reading from the ACBnSDA register during a receive, or when written to during a transmit. When the ACBnCTL1.START bit is set, reading the ACBnSDA register does not clear the SDAST bit. This enables the ACB to send a repeated start in master receive mode.
0 – ACB module is not waiting for data transfer.
1 – ACB module is waiting for data to be loaded or unloaded.

SLVSTP The Slave Stop bit indicates that a Stop Condition was detected after a slave transfer (i.e., after a slave transfer in which MATCH or GCMATCH is set). Writing 1 to SLVSTP clears it. It is also cleared when the module is disabled. Writing 0 to SLVSTP is ignored.
0 – No stop condition after slave transfer occurred.
1 – Stop condition after slave transfer occurred.

29.5.3 ACBn Control Status Register (ACBnCST)

The ACBnCST registers are 8-bit, read/write registers that maintain the current ACB status. Following reset, Idle mode, Halt mode, and disabling the module, the non-reserved bits of the ACBnCST registers are cleared.

7	6	5	4	3	2	1	0
ARPMATCH	MATCHAF	TGSCL	TSDA	GCMTCH	MATCH	BB	BUSY

BUSY The BUSY bit indicates that the ACB module is:

- Generating a Start Condition
- In Master mode (ACBnST.MASTER is set)
- In Slave mode (ACBnCST.MATCH or ACBnCST.GCMTCH is set)
- In the period between detecting a Start and completing the reception of the address byte. After this, the ACB either becomes not busy or enters slave mode.

The BUSY bit is cleared by the completion of any of the above states, and by disabling the module. BUSY is a read only bit. It must always be written with 0.
0 – ACB module is not busy.
1 – ACB module is busy.

BB	The Bus Busy bit indicates the bus is busy. It is set when the bus is active (i.e., a low level on either SDAn or SCLn) or by a Start Condition. It is cleared when the module is disabled, on detection of a Stop Condition, or when writing 1 to this bit. See section Section 29.6 Usage Notes for a description of the use of this bit. 0 – Bus is not busy. 1 – Bus is busy.
MATCH	The Address Match bit indicates in slave mode when ACBnADDR.SAEN is set and the first seven bits of the address byte (the first byte transferred after a Start Condition) matches the 7-bit address in the ACBnADDR register, or when ACBnADDR2.SAEN is set and the first seven bits of the address byte matches the 7-bit address in the ACBnADDR2 register. It is cleared by Start Condition or repeated Start and Stop Condition (including illegal Start or Stop Condition). 0 – No address match occurred. 1 – Address match occurred.
GCMTCH	The Global Call Match bit is set in slave mode when the ACBnCTL1.GCMEN bit is set and the address byte (the first byte transferred after a Start Condition) is 00h. It is cleared by a Start Condition or repeated Start and Stop Condition (including illegal Start or Stop Condition). 0 – No global call match occurred. 1 – Global call match occurred.
TSDA	The Test SDA bit samples the state of the SDAn signal. This bit can be used while recovering from an error condition in which the SDAn signal is constantly pulled low by a slave that went out of sync. This bit is a read-only bit. Data written to it is ignored.
TGSCL	The Toggle SCL bit enables toggling the SCLn signal during error recovery. When the SDAn signal is low, writing 1 to this bit drives the SCn signal high for one cycle. Writing 1 to TGSCL when the SDAn signal is high is ignored. The bit is cleared when the clock toggle is completed. 0 – Writing 0 has no effect. 1 – Writing 1 toggles the SDAn signal high for one cycle.
MATCHAF	The Match Address Field bit indicates which of two address fields matched the slave address. If both fields match, the MATCHAF bit is cleared. The MATCHAF bit is cleared by a Start condition, repeated Start condition, or Stop condition. Illegal Start and Stop conditions also clear the MATCHAF bit. 0 – ACBnADDR.ADDR field matched the slave address. 1 – ACBnADDR2.ADDR field matched the slave address.
ARPMATCH	The ARP Address Match bit indicates when an ARP match occurs. This bit is set in slave mode when ACBnCTL3.ARPEN is set and the address byte (first byte transferred after a Start condition) is 110 0001b. The ARPMATCH bit is cleared by a Start condition, repeated Start condition, or Stop condition. Illegal Start and Stop conditions also clear the ARPMATCH bit. 0 – Slave address was not 110 0001b. 1 – ARP address match enabled, and slave address was 110 0001b.

29.5.4 ACBn Control Register 1 (ACBnCTL1)

The ACBnCTL1 registers are 8-bit, read/write registers that configure and control the ACB modules. Following reset, Idle mode, Halt mode, and disabling the module, the ACBnCTL1 register is cleared.

7	6	5	4	3	2	1	0
STASTRE	NMINTE	GCMEN	ACK	DMAEN	INTEN	STOP	START

START The Start bit is set to generate a Start Condition on the ACCESS.bus. The START bit is cleared when the Start Condition is sent, or upon detection of a Bus Error (ACBnST.BER = 1). This bit should be set only when in Master mode, or when requesting Master mode. If this device is not the active master of the bus (ACBnST.MASTER = 0), setting the START bit generates a Start Condition as soon as the ACCESS.bus is free (ACBnCST.BB = 0). An address send sequence should then be performed. If this device is the active master of the bus (ACBnST.MASTER = 1), when the START bit is set, a write to the ACBnSDA register generates a Start Condition, then the ACBnSDA data is transmitted as the slave's address and the requested transfer direction. This case is a repeated Start Condition. It may be used to switch the direction of the data flow between the master and the slave, or to choose another slave device without using a Stop Condition in between.
0 – Writing 0 has no effect.
1 – Writing 1 generates a Start condition.

STOP The Stop bit in master mode generates a Stop Condition that completes or aborts the current message transfer. This bit clears itself after the Stop condition is issued.
0 – Writing 0 has no effect.
1 – Writing 1 generates a Stop condition.

INTEN The Interrupt Enable bit controls asserting ACB interrupts. When the INTEN bit is cleared, interrupts are disabled. When the INTEN bit is set, interrupts are enabled.
0 – Interrupts disabled.
1 – Interrupts enabled.
An interrupt is asserted on any of the following events:

- An address MATCH is detected (ACBnST.NMATCH = 1) and the NMINTE bit is set.
- A Bus Error occurs (ACBnST.BERR = 1).
- Negative acknowledge after sending a byte (ACBnST.NEGACK = 1).
- If DMA is not enabled, an interrupt is asserted on acknowledgement of each transaction (same as hardware setting the ACBnST.SDAST bit).
- If ACBnCTL1.STASTRE = 1, in master mode after a successful start (ACBnST.STASTR = 1).
- Detection of a Stop Condition while in slave receive mode (ACBnST.SLVSTP = 1).

DMAEN The DMA Enable bit controls whether DMA requests are enabled. A DMA request is asserted at the end of any data transaction (ACBnST.SDAST = 1). If INTEN is set, interrupts are generated upon occurrence of any error or a new match).
0 – DMA requests disabled.
1 – DMA requests enabled.

ACK The Acknowledge bit holds the value this device sends in master or slave mode during the next acknowledge cycle. Setting this bit to 1 instructs the transmitting device to stop sending data, since the receiver either does not need, or cannot receive, any more data. This bit is cleared after the first acknowledge cycle. This bit is ignored when in transmit mode.

GCMEN The Global Call Match Enable bit enables the match of an incoming address byte to the general call address (Start Condition followed by address byte of 00h) while the ACB is in slave mode. When cleared, the ACB does not respond to a global call.
0 – Global call matching disabled.
1 – Global call matching enabled.

NMINTE The New Match Interrupt Enable controls whether ACB interrupts are generated on new matches. Set the NMINTE bit to enable the interrupt on a new match (i.e., when ACBnST.NMATCH is set). The interrupt is issued only if the ACBnCTL1.INTEN bit is set. This bit must be set when using DMA for the data transfer.
0 – New match interrupts disabled.
1 – New match interrupts enabled.

STASTRE The Stall After Start Enable bit enables the stall after start mechanism. When enabled, the ACB is stalled after the address byte. When the STASTRE bit is clear, the ACBnST.STASTR bit is always clear.
0 – No stall after start.
1 – Stall-after-start enabled.

29.5.5 ACBn Control Register 2 (ACBnCTL2)

The ACBnCTL2 registers are 8-bit, read/write registers that control the module and select the ACB clock rate. At reset, the ACBnCTL2 register is cleared.

7	1	0
SCLFRQ6:0		ENABLE

ENABLE The Enable bit controls the ACB module. When this bit is set, the ACB module is enabled. When the Enable bit is clear, the ACB module is disabled, the ACBnCTL1, ACBnST, and ACBnCST registers are cleared, and the clocks are halted.
0 – ACB module disabled.
1 – ACB module enabled.

SCLFRQ6:0 The SCL Frequency field specifies the low 7 bits of the SCLn period (low time plus high time) in master mode. The ACBnCTL3 register holds the high 2 bits. The clock low time and high time are defined as follows:
$t_{SCLl} = t_{SCLh} = 2 \times SCLFRQ8:0 \times t_{CLK}$ in which t_{CLK} is the PCLK Clock period. The SCLFRQ8:0 field may have values in the range of 008h through 1FFh. Other values are reserved.

29.5.6 ACBn Control Register 3 (ACBnCTL3)

The ACBnCTL3 registers are 8-bit, read/write registers that expand the clock prescaler field and enable ARP matches. At reset, the ACBnCTL3 registers are cleared.

7	3	2	1	0
Reserved		ARPMEN	SCLFRQ8:7	

ARPMEN The ARP Match Enable bit enables the matching of an incoming address byte to the SMBus ARP address 110 0001b general call address (Start condition followed by address byte of 00h), while the ACB is in slave mode.
0 – ACB does not respond to ARP addresses.
1 – ARP address matching enabled.

SCLFRQ The SCL Frequency field specifies the SCL period (low time plus high time) in master mode. The ACBnCTL3 register provides a 2-bit expansion of this field, with the remaining 7 bits being held in the ACBnCTL2 register.

29.5.7 ACBn Own Address Register 1 (ACBnADDR1)

The ACBnADDR1 registers are 8-bit, read/write registers that hold the module's first ACCESS.bus address. After reset, their values are undefined.

7	6	0
SAEN	ADDR	

ADDR The Own Address field holds the first 7-bit ACCESS.bus address of this device. When in slave mode, the first 7 bits received after a Start Condition are compared to this field (first bit received to bit 6, and the last to bit 0). If the address field matches the received data and the SAEN bit is set, a match is detected.

SAEN The Slave Address Enable bit controls whether address matching is performed in slave mode. When set, the SAEN bit indicates that the ADDR field holds a valid address and enables the match of ADDR to an incoming address byte. When cleared, the ACB does not check for an address match.
0 – Address matching disabled.
1 – Address matching enabled.

29.5.8 ACBn Own Address Register 2 (ACBnADDR2)

The ACBnADDR2 registers are 8-bit, read/write registers that hold the module's second ACCESS.bus address. After reset, their values are undefined.

7	6	0
SAEN	ADDR	

ADDR The Own Address field holds the second 7-bit ACCESS.bus address of this device. When in slave mode, the first 7 bits received after a Start Condition are compared to this field (first bit received to bit 6, and the last to bit 0). If the address field matches the received data and the SAEN bit is set, a match is detected.

SAEN The Slave Address Enable bit controls wheth- er address matching is performed in slave mode. When set, the SAEN bit indicates that the ADDR field holds a valid address and enables the match of ADDR to an incoming address byte. When cleared, the ACB does not check for an address match.
0 – Address matching disabled.
1 – Address matching enabled.

29.6 Usage Notes

- When the ACB module is disabled, the ACBnCST.BB bit is cleared. After enabling the ACB (ACBnCTL2.ENABLE = 1) in systems with more than one master, the bus may be in the middle of a transaction with another device, which is not reflected in the BB bit. There is a need to allow the ACB to synchronize to the bus activity status before issuing a request to become the bus master, to prevent bus errors. Therefore, before issuing a request to become the bus master for the first time, software should check that there is no activity on the bus by checking the BB bit after the bus allowed timeout period.
- When waking up from power down, before checking the ACBnCST.MATCH bit, test the ACBnCST.BUSY bit to make sure that the address transaction has finished.
- The BB bit is intended to solve a deadlock in which two, or more, devices detect a usage conflict on the bus and both devices cease being bus masters at the same time. In this situation, the BB bits of both devices are active (because each deduces that there is another master currently performing a transaction, while in fact no device is executing a transaction), and the bus would stay locked until some device sends a ACBnCTL1.STOP condition. The ACBnCST.BB bit allows software to monitor bus usage, so it can avoid sending a STOP signal in the middle of the transaction of some other device on the bus. This bit detects whether the bus remains unused over a certain period, while the BB bit is set.

- In some cases, the bus may get stuck with the SCLn or SDAn lines active. A possible cause is an erroneous Start or Stop Condition that occurs in the middle of a slave receive session. When the SCLn signal is stuck active, there is nothing that can be done, and it is the responsibility of the module that holds the bus to release it. When the SDAn signal is stuck active, the ACB module enables the release of the bus by using the following sequence. Note that in normal cases, the SCLn signal may be toggled only by the bus master. This protocol is a recovery scheme which is an exception that should be used only in the case when there is no other master on the bus. The recovery procedure is as follows:
  1. Disable and re-enable the module to set it into the not addressed slave mode
  2. Set the ACBnCTL1.START bit to make an attempt to issue a Start Condition.
  3. Check if the SDAn signal is active (low) by reading ACBnCST.TSDA bit. If it is active, issue a single SCLn cycle by writing 1 to ACBnCST.TGSCL bit. If the SDAn line is not active, continue from step 5.
  4. Check if the ACBnST.MASTER bit is set, which indicates that the Start Condition was sent. If not, repeat step 3 and 4 until the SDAn signal is released.
  5. Clear the BB bit. This enables the START bit to be executed. Continue according to section Section 29.2.1.8 Bus Idle Error Recovery

30 Real Time Clock

The Real Time Clock module (RTC) implements a 32-bit counter with a 16-bit programmable prescaler, driven by the 32.768 kHz Slow Clock through a 5-bit clock divider, as shown in Figure 30-1. This provides a time base with a long wraparound period (over 136 years) and the ability to schedule long-duration time delays and periodic interrupts. Three compare registers are implemented: one register for asserting an interrupt on wraparound of the programmable prescaler and two registers for asserting interrupts when the 32-bit counter reaches specified values.

The RTC is typically programmed to divide a 32.768 kHz Slow Clock by 32, which provides a 1024 Hz clock input to the 16-bit prescaler. The prescaler is typically programmed for division by 1024, resulting in a 1 Hz clock input to the 32-bit counter. However, the RTC can operate with a different Slow Clock frequency, an input clock divisor of 1, 2, 4, 8, 16, or 32, and any integer prescaler divisor between 1 and 65,536 (decimal).

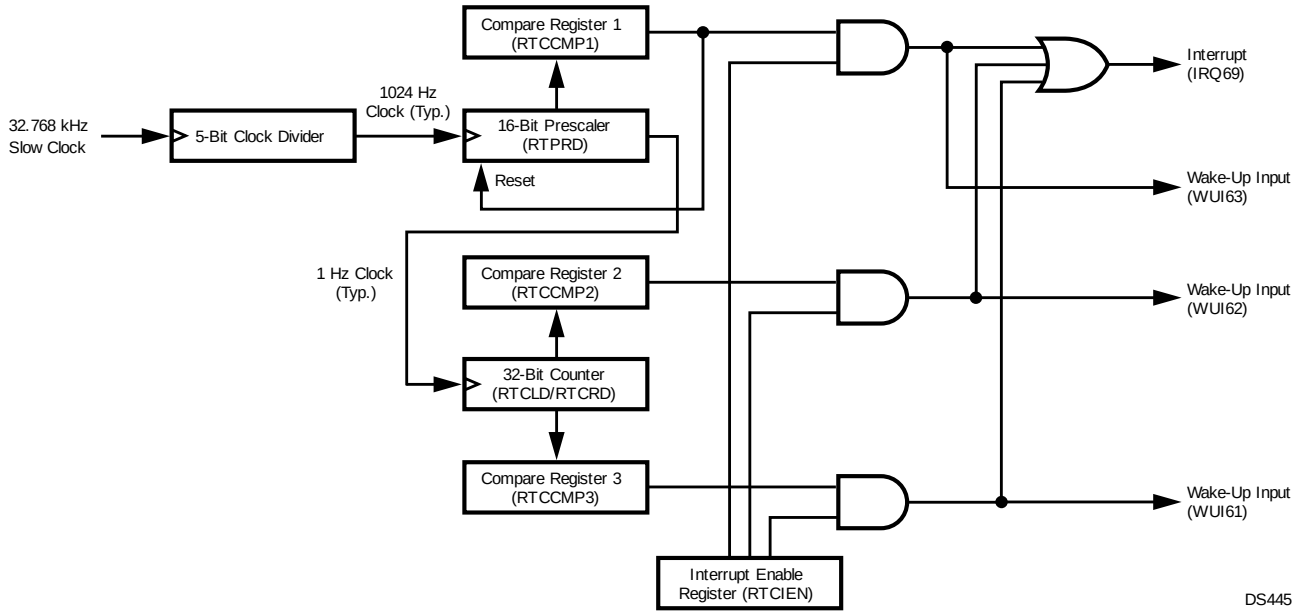


Figure 30-1. Real Time Clock

30.1 Programming

When software changes any of the timing parameters, the change is delayed until it can be accepted synchronously to the operation of the RTC. Software can poll register bits which indicate when the change has been accepted.

- *Clock divider*—when the RTDIV field of the RTCCST register is written, the new divisor is not accepted until the end of the divider period using the old divisor. While the change is pending but not yet accepted, the RTUDIV bit in the RTUDST register will be set.
- *Prescaler*—the prescaler divisor is controlled by the RTCCMP1 register. When this register is loaded, the RTUCP1 bit in the RTUDIV register is set, and it remains set until the new value is accepted.
- *Counter*—the counter is loaded by writing the RTCLD register. When this register is written, the RTURTC bit in the RTUDST register is set, and it remains set until the new value is accepted, which occurs at the next rising edge of the counter input clock (prescaler output).

When software reads the current prescaler value in the RTPRD register or the current counter value in the RTCRD register, the data is delayed by four PCLK Clock cycles for synchronization to the bus. Reading these registers within one second after wake-up from Idle mode may return incorrect values. Raising the clock input to the 32-bit counter above 1 Hz provides a proportional reduction of the time during which incorrect values may be returned. Timekeeping is not affected by these incorrect values.

30.2 Interrupt

The events enabled in the RTCIEN register are ORed together to generate the interrupt request signal IRQ69. The three individual events RTCEVT1, RTCEVT2, and RTCEVT3 are mapped to MIWU inputs WUI63, WUI62, and WUI61, respectively.

30.3 Reset

After software starts the RTC by setting the RTSTRT bit in the RTCCST register, software cannot stop the RTC. The RTC can only be stopped by a power-on reset.

A system reset (except a power-on reset) does not affect the contents of the timing parameters, which allows the RTC to continue functioning through system reset events. Only a power-on reset stops counting and resets the RTC registers to their default values. If a system reset occurs while changes to the timing parameters are pending, those changes will be discarded without being accepted.

The prescaler can be reset by setting the RTPRST bit in the RTCCST register. The reset will occur on the next rising edge of the prescaler input clock (divider output). After soft-ware sets the RTPRST bit, the bit remains set until the prescaler is reset.

30.4 Real-Time Clock Interface Registers

The Real-Time Clock interface uses the registers listed in Table 30-1.

Table 30-1. Real-Time Clock Interface Registers

NAME	ADDRESS	DESCRIPTION
RTCCST	FF A800h	RTC Control and Status Register
RTUDST	FF A804h	RTC Update Status Register
RTCEIST	FF A808h	RTC Event and Interrupt Status Register
RTCIEN	FF A80Ch	RTC Interrupt Enable Register
RTPRD	FF A810h	RTC Prescaler Read Register
RTCRD	FF A814h	RTC Counter Read Register
RTCLD	FF A818h	RTC Counter Load Register
RTCCMP1	FF A81Ch	RTC Compare Register 1
RTCCMP2	FF A820h	RTC Compare Register 2
RTCCMP3	FF A824h	RTC Compare Register 3

30.4.1 RTC Control and Status Register (RTCCST)

The RTCCST register is an 8-bit, read/write register that provides control and status for the RTC. At system reset, the RTPRST bit is cleared, but the other bits in this register retain their values. At power-on reset, this register is cleared to 00h.

7	5	4	3	2	0
Reserved		RTSTRT	RTPRST	RTDIV	

RTDIV The RTC Divider field selects the divisor for the 5-bit clock divider. Values above 101b are reserved.
000 – ÷1
001 – ÷2
010 – ÷4
011 – ÷8
100 – ÷16
101 – ÷32

RTPRST The RTC Prescaler Reset bit is used to clear the 16-bit prescaler to 0000h. The reset will occur on the next rising edge of the prescaler input clock (divider output). After software writes 1 to the bit to reset the prescaler, the bit remains set until the prescaler is cleared.
0 – Normal operation.
1 – Prescaler reset pending.

RTSTRT The RTC Start bit enables the RTC to count. This bit is can only be cleared by a power-on reset.
0 – RTC suspended.
1 – RTC counting.

30.4.2 RTC Update Status Register (RTUDST)

The RTUDST register is an 8-bit, read-only register that indicates pending changes to RTC registers. At system reset or power-on reset, this register is cleared to 00h.

7	5	4	3	2	1	0
Reserved		RTUCP3	RTUCP2	RTUCP1	RTURPTC	RTUDIV

RTUDIV The RTC Update Clock Divisor bit indicates a new divisor for the 5-bit clock divider has been loaded, but not yet accepted.
0 – No new divisor loaded.
1 – Pending change to the clock divisor.

RTURTC The RTC Update Counter bit indicates a new value for the 32-bit counter has been loaded, but not yet accepted.
0 – No new counter value loaded.
1 – Pending change to the counter value.

RTUCP1 The RTC Update Compare Register 1 bit indicates a new terminal count value for the 16-bit prescaler has been loaded into the RTCCMP1 register, but not yet accepted.
0 – No new prescaler terminal count loaded.
1 – Pending change to the terminal count.

RTUCP2 The RTC Update Compare Register 2 bit indicates a new interrupt trigger value for the 32-bit counter has been loaded into the RTCCMP2 register, but not yet accepted.
0 – No new trigger value loaded.
1 – Pending change to the trigger value.

RTUCP3 The RTC Update Compare Register 3 bit indicates a new interrupt trigger value for the 32-bit counter has been loaded into the RTCCMP3 register, but not yet accepted.
0 – No new trigger value loaded.
1 – Pending change to the trigger value.

30.4.3 RTC Event and Interrupt Status Register (RTCEIST)

The RTCEIST register is an 8-bit, read/write register that indicates the status of match events from the compare registers. Individual bits in this register can be cleared by writing them with 1. At system reset or power-on reset, this register is cleared to 00h.

7	3	2	1	0
Reserved		RTCEVT3	RTCEVT2	RTCEVT1

RTCEVT1 The RTC Event 1 bit indicates a match occurred between the RTCCMP1 register and the 16-bit prescaler.
0 – No match occurred since this bit was last cleared.
1 – A match occurred.

RTCEVT2 The RTC Event 2 bit indicates a match occurred between the RTCCMP2 register and the 32-bit counter.
0 – No match occurred since this bit was last cleared.
1 – A match occurred.

RTCEVT3 The RTC Event 3 bit indicates a match occurred between the RTCCMP3 register and the 32-bit counter.
0 – No match occurred since this bit was last cleared.
1 – A match occurred.

30.4.4 RTC Interrupt Enable Register (RTCIEN)

The RTCIEN register is an 8-bit, read/write register that enables interrupts from the compare registers. At system reset or power-on reset, this register is cleared to 00h.

7	3	2	1	0
Reserved		RTCIEN3	RTCIEN2	RTCIEN1

RTCIEN1 The RTC Interrupt Enable 1 bit is used to en- able an interrupt when a match occurs between the 16-bit prescalar and the RTCCMP1 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RTCIEN2 The RTC Interrupt Enable 2 bit is used to en- able an interrupt when a match occurs between the 32-bit counter and the RTCCMP2 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

RTCIEN3 The RTC Interrupt Enable 3 bit is used to en- able an interrupt when a match occurs between the 32-bit counter and the RTCCMP3 register.
0 – Interrupt disabled.
1 – Interrupt enabled.

30.4.5 RTC Prescaler Read Register (RTPRD)

The RTPRD register is a 16-bit, read-only register used to read the current value of the 16-bit prescaler. System reset does not affect this register. At power-on reset, this register is cleared to 0000h.

15	0
RTPRD	

30.4.6 RTC Counter Read Register (RTCRD)

The RTCRD register is a 32-bit, read-only register used to read the current value of the 32-bit counter. System reset does not affect this register. At power-on reset, this register is initialized to 0000 0001h.

15															0
RTCRD															

30.4.7 RTC Counter Load Register (RTCLD)

The RTCLD register is a 32-bit, read/write register used to load a new value into the 32-bit counter. The counter will be loaded on the next rising edge of its input clock (prescaler output). When the RTCLD register is written, the RTURTC bit in the RTUDST register is set. This bit is cleared when the value in the RTCLD register is loaded into the counter. When the RTCLD register is read, it returns the last value that was written to it. At system reset or power-on reset, this register is cleared to 0000 0000h.

15															0
RTCLD															

30.4.8 RTC Compare Register 1 (RTCCMP1)

The RTCCMP register is a 16-bit, read/write register which holds a value compared with the contents of the 16-bit prescaler. When this register is loaded, the RTUCP1 bit in the RTUDST register is set, and the bit remains set until the up- dated RTCCMP1 register value takes effect. When the contents of the RTCCMP1 register matches the contents of the prescaler, the prescaler will be cleared on the next rising edge of its input clock (divider output), and the RTCEVT1 bit in the RTCEIST register is set. If the RTCIEN1 bit in the RTCIEN register is set, an interrupt is asserted. System reset does not affect this register. At power-on reset, this register is initialized to 7FFFh.

15															0
RTCCMP1															

30.4.9 RTC Compare Register 2 (RTCCMP2)

The RTCCMP2 register is a 32-bit, read/write register which holds a value compared with the contents of the 32-bit counter. When this register is loaded, the RTUCP2 bit in the RTUDST register is set, and the bit remains set until the up- dated RTCCMP2 register value takes effect. When the contents of the RTCCMP2 register matches the contents of the counter, the RTCEVT2 bit in the RTCEIST register is set. If the RTCIEN2 bit in the RTCIEN register is set, an interrupt is asserted. System reset does not affect this register. At power-on reset, this register is initialized to FFFF FFFFh.

31															0
RTCCMP2															

30.4.10 RTC Compare Register 3 (RTCCMP3)

The RTCCMP3 register is a 32-bit, read/write register which holds a value compared with the contents of the 32-bit counter. When this register is loaded, the RTUCP3 bit in the RTUDST register is set, and the bit remains set until the updated RTCCMP3 register value takes effect. When the contents of the RTCCMP3 register matches the contents of the counter, the RTCEVT3 bit in the RTCEIST register is set. If the RTCIEN3 bit in the RTCIEN register is set, an interrupt is asserted. System reset does not affect this register. At power-on reset, this register is initialized to FFFF FFFFh.

31															0
RTCCMP3															

31 Timing and Watchdog Module

The Timing and Watchdog Module (TWM) generates the clocks and interrupts used for timing periodic functions in the system; it also provides Watchdog protection over soft- ware execution.

The TWM is designed to provide flexibility in system design by configuring various clock ratios and by selecting the Watchdog clock source. After setting the TWM configuration, software can lock it for a higher level of protection against erroneous software action. Once the TWM is locked, only reset can release it.

31.1 TWM Structure

Figure 115 is a block diagram showing the internal structure of the Timing and Watchdog module. There are two main sections: the Real-Time Timer (T0) section at the top and the Watchdog section on the bottom.

All counting activities of the module are based on the Slow Clock (SLCLK). A prescaler counter divides this clock to make a slower clock. The prescaler factor is defined by a 3- bit field in the Timer and Watchdog Prescaler register, which selects either 1, 2, 4, 8, 16, or 32 as the divisor. Therefore, the prescaled clock period can be 2, 4, 8, 16, or 32 times the Slow Clock period. The prescaled clock signal is called T0IN.

31.2 Timer to Operation

Timer T0 is a programmable 16-bit down counter that can be used as the time base for real-time operations such as a periodic audible tick. It can also be used to drive the Watch- dog circuit.

The timer starts counting from the value loaded into the TWMT0 register and counts down on each rising edge of T0IN. When the timer reaches zero, it is automatically reloaded from the TWMT0 register and continues counting down from that value. Therefore, the frequency of the timer is:

$$f_{TIMER} = \frac{f_{SLCLK}}{(TWTM0 + 1) \times prescaler} \quad (12)$$

When an external crystal oscillator is used as the SLCLK source or when the fast clock is divided accordingly, f_{SLCLK} is 32.768 kHz.

The value stored in TWMT0 can range from 0001h to FFFFh.

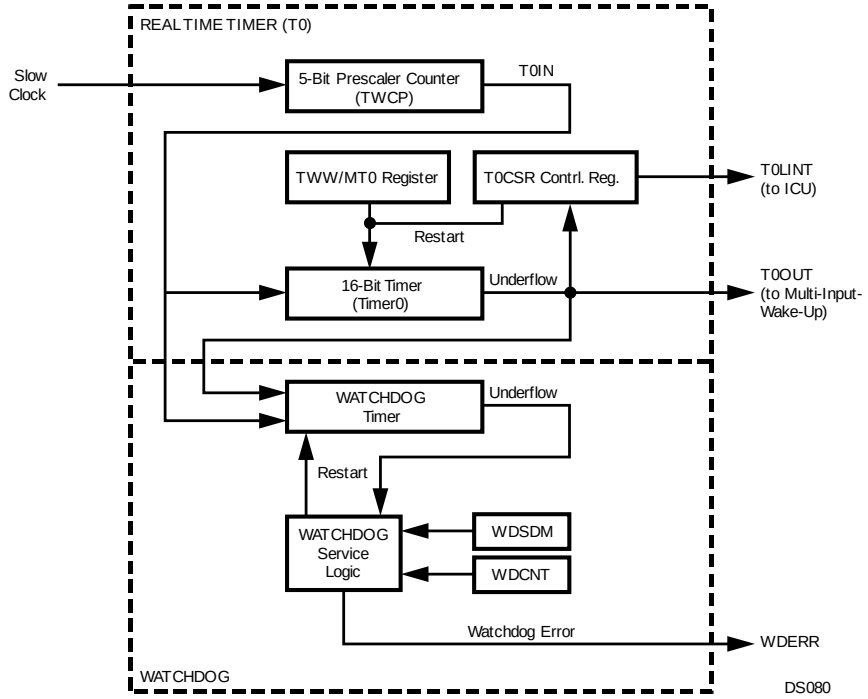


Figure 31-1. Timing and Watchdog Module Block Diagram

When the counter reaches zero, an internal timer signal called T0OUT is set for one T0IN clock cycle. This signal sets the TC bit in the TWMT0 Control and Status Register (T0CSR). It also asserts an interrupt (IRQ70), when enabled by the T0CSR.T0INTE bit. T0OUT is also an input to the MIWU (see Section 17), so an edge-triggered interrupt is also available through this alternative mechanism.

If software loads the TWMT0 register with a new value, the timer uses that value the next time that it reloads the 16-bit timer register (in other words, after reaching zero). Software can restart the timer at any time (on the very next edge of the T0IN clock) by setting the Restart (RST) bit in the T0CSR register. The T0CSR.RST bit is cleared automatically upon restart of the 16-bit timer.

NOTE

To enter Power Save or Idle mode after setting the T0CSR.RST bit, software must wait for the reset operation to complete before performing the switch.

31.3 Watchdog Operation

The Watchdog is an 8-bit down counter that operates on the rising edge of a specified clock source. At reset, the Watchdog is disabled; it does not count and no Watchdog signal is generated. A write to either the Watchdog Count (WDCNT) register or the Watchdog Service Data Match (WDSDM) register starts the counter. The Watchdog counter counts down from the value programmed in the WDCNT register. Once started, only a reset can stop the Watchdog from operating.

The Watchdog can be programmed to use either T0OUT or T0IN as its clock source (the output and input of Timer T0, respectively). The TWCFG.WDCT0I bit selects the clock source.

Software must periodically service the Watchdog. There are two ways to service the Watchdog, the choice depending on the programmed value of the WDSDME bit in the Timer and Watchdog Configuration (TWCFG) register.

If the TWCFG.WDSDME bit is clear, the Watchdog is serviced by writing a value to the WDCNT register. The value written to the register is reloaded into the Watchdog counter. The counter then continues counting down from that value.

If the TWCFG.WDSDME bit is set, the Watchdog is serviced by writing the value 5Ch to the Watchdog Service Data Match (WDSDM) register. This reloads the Watchdog counter with the value previously programmed into the WDCNT register. The counter then continues counting down from that value.

A Watchdog error signal is generated by any of the following events:

- The Watchdog serviced too late.
- The Watchdog serviced too often.
- The WDSDM register is written with a value other than 5Ch when WDSDM type servicing is enabled (TWCFG.WDSDME = 1).

A Watchdog error condition resets the device.

31.3.1 Register Locking

The Timer and Watchdog Configuration (TWCFG) register is used to set the Watchdog configuration. It controls the Watchdog clock source (T0IN or T0OUT), the type of Watchdog servicing (using WDCNT or WDSDM), and the locking state of the TWCFG, TWCPR, TIMER0, T0CSR, and WDCNT registers. A register that is locked cannot be read or written. A write operation is ignored and a read operation returns unpredictable results.

If the TWCFG register is itself locked, it remains locked until the device is reset. Any other locked registers also remain locked until the device is reset. This feature prevents a run- away program from tampering with the programmed Watchdog function.

31.3.2 Power Save Mode Operation

The Timer and Watchdog Module is active in both the Power Save and Idle modes. The clocks and counters continue to operate normally in these modes. The WDSDM register is accessible in the Power Save and Idle modes, but the other TWM registers are accessible only in the Active mode. Therefore, Watchdog servicing must be carried out using the WDSDM register in the Power Save or Idle mode.

In the Halt mode, the entire device is frozen, including the Timer and Watchdog Module. On return to Active mode, operation of the module resumes at the point at which it was stopped.

NOTE

After a restart or Watchdog service through WDCNT, do not enter Power Save mode for a period equivalent to 5 Slow Clock cycles.

31.4 TWM Registers

The TWM registers controls the operation of the Timing and Watchdog Module. There are six such registers:

Table 31-1. TWM Registers

NAME	ADDRESS	DESCRIPTION
TWCFG	FF A000h	Timer and Watchdog Configuration Register
TWCP	FF A004h	Timer and Watchdog Clock Prescaler Register
TWMT0	FF A008h	TWM Timer 0 Register
T0CSR	FF A00Ch	TWMT0 Control and Status Register
WDCNT	FF A010h	Watchdog Count Register
WDSDM	FF A014h	Watchdog Service Data Match Register

The WDSDM register is accessible in both Active and Power Save mode. The other TWM registers are accessible only in Active mode.

31.4.1 Timer and Watchdog Configuration Register (TWCFG)

The TWCFG register is a byte-wide, read/write register that selects the Watchdog clock input and service method, and also allows the Watchdog registers to be selectively locked. A locked register cannot be read or written; a read operation returns unpredictable values and a write operation is ignored. Once a lock bit is set, it cannot be cleared until the device is reset. At reset, the non-reserved bits of the register are cleared. The register format is shown below.

7	6	5	4	3	2	1	0
Reserved		WDSDME	WDCT0I	LWDCNT	LTWMT0	LTWCP	LTWCFG

LTWCFG The Lock TWCFG Register bit controls access to the TWCFG register. When clear, access to the TWCFG register is allowed. When set, the TWCFG register is locked.
0 – TWCFG register unlocked.
1 – TWCFG register locked.

LTWCP The Lock TWCP Register bit controls access to the TWCP register. When clear, access to the TWCP register is allowed. When set, the TWCP register is locked.
0 – TWCP register unlocked.
1 – TWCP register locked.

LTWMT0 The Lock TWMT0 Register bit controls access to the TWMT0 register. When clear, access to the TWMT0 and T0CSR registers are allowed. When set, the TWMT0 and T0CSR registers are locked.
0 – TWMT0 register unlocked.
1 – TWMT0 register locked.

LWDCNT The Lock LDWCNT Register bit controls access to the LDWCNT register. When clear, access to the LDWCNT register is allowed. When set, the LDWCNT register is locked.
0 – LDWCNT register unlocked.
1 – LDWCNT register locked.

WDCT0I The Watchdog Clock from T0IN bit selects the clock source for the Watchdog timer. When clear, the T0OUT signal (the output of Timer T0) is used as the Watchdog clock. When set, the T0IN signal (the prescaled Slow Clock) is used as the Watchdog clock.
0 – Watchdog timer is clocked by T0OUT.
1 – Watchdog timer is clocked by T0IN.

WDSDME The Watchdog Service Data Match Enable bit controls which method is used to service the Watchdog timer. When clear, Watchdog servicing is accomplished by writing a count value to the WDCNT register; write operations to the Watchdog Service Data Match (WDSDM) register are ignored. When set, Watchdog servicing is accomplished by writing the value 5Ch to the WDSDM register.
0 – Write a count value to the WDCNT register to service the Watchdog timer.
1 – Write 5Ch to the WDSDM register to service the Watchdog timer.

31.4.2 Timer and Watchdog Clock Prescaler Register (TWCP)

The TWCP register is a byte-wide, read/write register that specifies the prescaler value used for dividing the low-frequency clock to generate the T0IN clock. At reset, the non-reserved bits of the register are cleared. The register format is shown below.

7	3	2	0
Reserved		MDIV	

MDIV Main Clock Divide. This 3-bit field defines the prescaler factor used for dividing the low speed device clock to create the T0IN clock. The allowed 3-bit values and the corresponding clock divisors and clock rates are listed below.

MDIV	Clock Divisor (f_{SCLK} = 32.768 kHz)	T0IN Frequency
000	1	32.768 kHz
001	2	16.384 kHz
010	4	8.192 kHz
011	8	4.096 kHz
100	16	2.056 kHz
101	32	1.024 kHz
Other	Reserved	N/A

31.4.3 TWM Timer 0 Register (TWMT0)

The TWMT0 register is a 16-bit, read/write register that de- fines the T0OUT interrupt rate. At reset, TWMT0 register is initialized to FFFFh. The register format is shown below.

15	0
PRESET	

PRESET The Timer T0 Preset field holds the value used to reload Timer T0 on each underflow. Therefore, the frequency of the Timer T0 interrupt is the frequency of T0IN divided by (PRESET+1). The allowed values of PRESET are 0001h through FFFFh.

31.4.4 TWMT0 Control and Status Register (T0CSR)

The T0CSR register is a byte-wide, read/write register that controls Timer T0 and shows its current status. At reset, the non-reserved bits of the register are cleared. The register format is shown below.

7	5	4	3	2	1	0
Reserved		FRZT0E	WDLTD	T0INTE	TC	RST

RST The Restart bit is used to reset Timer T0. When this bit is set, it forces the timer to reload the value in the TWMT0 register on the next rising edge of the selected input clock. The RST bit is reset automatically by the hardware on the same rising edge of the selected input clock. Writing a 0 to this bit position has no effect. At reset, the non-reserved bits of the register are cleared.
0 – Writing 0 has no effect.
1 – Writing 1 resets Timer T0.

TC The Terminal Count bit is set by hardware when the Timer T0 count reaches zero and is cleared when software reads the T0CSR register. It is a read-only bit. Any data written to this bit position is ignored.
0 – Timer T0 did not count down to 0.
1 – Timer T0 counted down to 0.

T0INTE The Timer T0 Interrupt Enable bit enables an interrupt to the CPU each time the Timer T0 count reaches zero. When this bit is clear, Timer T0 interrupts are disabled.
0 – Timer T0 interrupts disabled.
1 – Timer T0 interrupts enabled.

WDLTD The Watchdog Last Touch Delay bit is set when either WDCNT or WDSDM is written and the data transfer to the Watchdog is in progress (see WDCNT and WDSDM register description). When clear, it is safe to switch to Power Save mode.
0 – No data transfer to the Watchdog is in progress, safe to enter Power Save mode.
1 – Data transfer to the Watchdog in progress.

FRZT0E The Freeze Timer 0 Enable bit controls whether Timer 0 is stopped in Freeze mode. If this bit is set, the Timer 0 is frozen (stopped) when the Freeze input to the TWM is asserted. If the FRZT0E bit is clear, only the Watchdog timer is frozen by asserting the Freeze input signal. After reset, this bit is clear.
0 – Timer T0 unaffected by Freeze mode.
1 – Timer T0 stopped in Freeze mode.

31.4.5 Watchdog Count Register (WDCNT)

The WDCNT register is a byte-wide, write-only register that holds the value that is loaded into the Watchdog counter each time the Watchdog is serviced. The Watchdog is started by the first write to this register. Each successive write to this register restarts the Watchdog count with the written value. At reset, this register is initialized to 0Fh.

7							0
PRESET							

The WDCNT register operates from PCLK Clock, not Slow Clock. Because the rest of the Watchdog service mechanism operates from Slow Clock, there is a small amount of latency between the write to the WDCNT register and the actual service of the Watchdog. Due to this latency it is not recommended to use small PRESET values. In Power Save mode, it is even more important because the latency time (in clock cycles) is even longer than in Active mode. The minimum PRESET value must be large enough to satisfy this latency, which is expressed in this relation:

$$\text{PRESET} > \left(\left(\frac{\text{Peripheral Bus Clock}}{\text{Watchdog Clock}} \right) \times 6 \right) + 2 \tag{13}$$

31.4.6 Watchdog Service Data Match Register (WDSDM)

The WSDSM register is a byte-wide, write-only register used for servicing the Watchdog. When this type of servicing is enabled (TWCFG.WDSDME = 1), the Watchdog is serviced by writing the value 5Ch to the WSDSM register. Each such servicing reloads the Watchdog counter with the value previously written to the WDCNT register. Writing any data other than 5Ch triggers a Watchdog error. Writing to the register more than once in one Watchdog clock cycle also triggers a Watchdog error signal. If this type of servicing is disabled (TWCFG.WDSDME = 0), any write to the WSDSM register is ignored.

7							0
RSTDATA							

31.5 Watchdog Programming Procedure

The highest level of protection against software errors is achieved by programming and then locking the Watchdog registers and using the WDSDM register for servicing. This is the procedure:

1. Write the desired values into the TWM Clock Prescaler register (TWCP) and the TWM Timer 0 register (TWMT0) to control the T0IN and T0OUT clock rates. The frequency of T0IN can be programmed to any of six frequencies ranging from 1/32 × fSLCLK to fSLCLK. The frequency of T0OUT is equal to the

frequency of T0IN divided by (1+ PRESET), in which PRESET is the value written to the TWMT0 register.

2. Configure the Watchdog clock to use either T0IN or T0OUT by setting or clearing the TWCFG.WDCT0I bit.
3. Write the initial value into the WDCNT register. This starts operation of the Watchdog and specifies the maximum allowed number of Watchdog clock cycles between service operations. Starting from this point, the Watchdog must be periodically serviced to prevent a device reset.
4. Set the T0CSR.RST bit to restart the TWMT0 timer.
5. Lock the Watchdog registers and enable the Watchdog Service Data Match Enable function by setting bits 0, 1, 2, 3, and 5 in the TWCFG register.
6. Service the Watchdog by periodically writing the value 5Ch to the WDSDM register at an appropriate rate. Servicing must occur at least once per period programmed into the WDCNT register, but no more than once in a single Watchdog input clock cycle.

32 Dual Multi-Function Timers

The two Multi-Function Timer modules each contain a pair of 16-bit timer/counters. Each timer/counter unit offers a choice of clock sources for operation and can be configured to operate in any of the following modes:

- Processor-Independent Pulse Width Modulation (PWM) mode, which generates pulses of a specified width and duty cycle, and which also provides a general-purpose timer/counter.
- Dual-Input Capture mode, which measures the elapsed time between occurrences of external events, and which also provides a general-purpose timer/counter.
- Dual Independent Timer mode, which generates system timing signals or counts occurrences of external events.
- Single-Input Capture and Single Timer mode, which provides one external event counter and one system timer.

Each timer unit uses two I/O pins, called TAn and TBn. (Only TA0 is available in the FBGA-144 package.)

32.1 Timer Structure

Figure 32-1 is a block diagram showing the internal structure of the MFT. There are two main functional blocks: a Timer/ Counter and Action block and a Clock Source block. The Timer/Counter and Action block contains two separate timer/counter units, called Timer/Counter 1 and Timer/Counter 2.

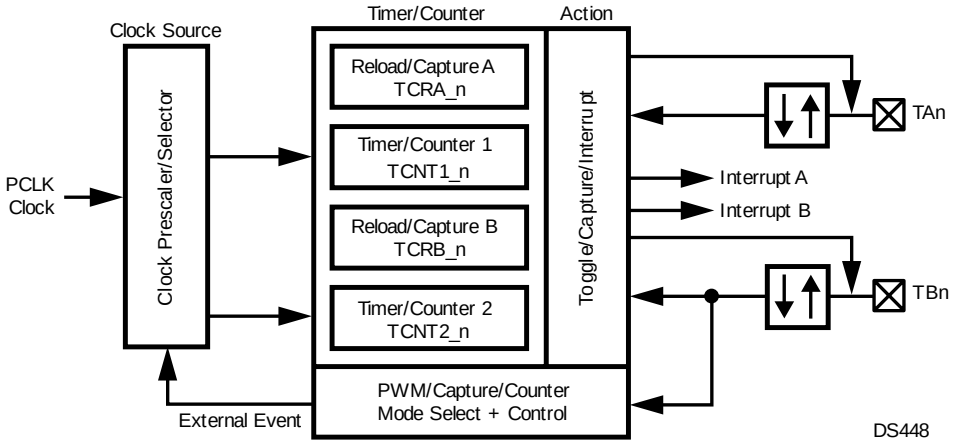


Figure 32-1. Multi-Function Timer Block Diagram

32.1.1 Timer/Counter Block

The Timer/Counter block contains the following functional blocks:

- Two 16-bit counters, Timer/Counter 1 (TCNT1_n) and Timer/Counter 2 (TCNT2_n)
- Two 16-bit reload/capture registers, TCRA_n and TCRB_n
- Control logic necessary to configure the timer to operate in any of the four operating modes
- Interrupt control and I/O control logic

32.1.2 Clock Source Block

The Clock Source block generates the signals used to clock the two timer/counter registers. The internal structure of the Clock Source block is shown in Figure 32-2.

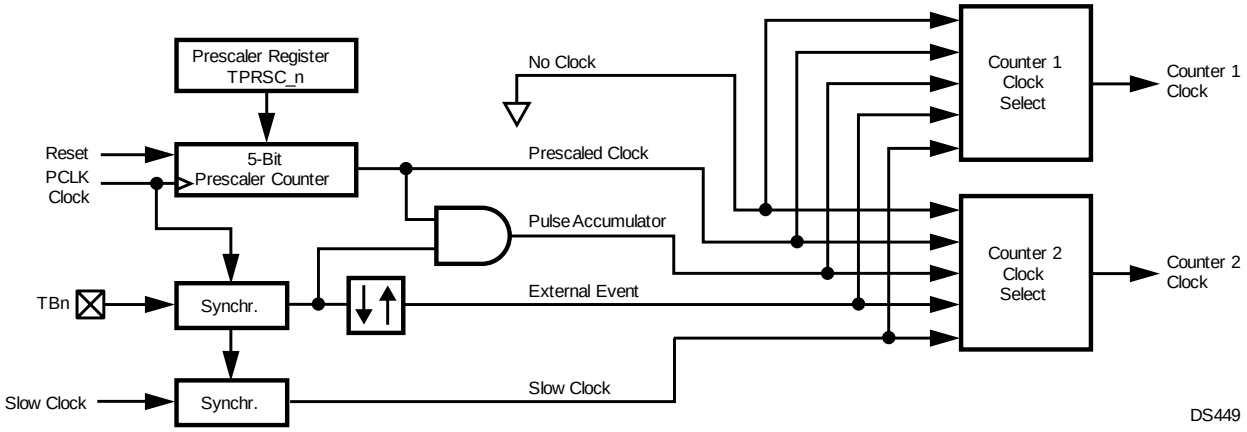


Figure 32-2. Multi-Function Timer Clock Source

32.1.2.1 Counter Clock Source Select

There are two clock source selectors that allow software to independently select the clock source for each of the two 16-bit counters from any of the following sources:

- No clock (which stops the counter)
- Prescaled PCLK Clock
- External event count based on TBn
- Pulse accumulate mode based on TBn
- Slow Clock

32.1.2.2 Prescaler

The 5-bit clock prescaler allows software to run the timer with a prescaled clock signal. The prescaler consists of a 5- bit read/write prescaler register (TPRSC_n) and a 5-bit down counter. The PCLK Clock is divided by the value contained in the prescaler register plus 1. Therefore, the PCLK Clock frequency can be divided by any value from 1 to 32. The prescaler register and down counter are both cleared upon reset.

32.1.2.3 External Event Clock

The TBn pin can be configured to operate as an external event input clock for either of the two 16-bit counters. This input can be programmed to detect either rising or falling edges. The minimum pulse width of the external signal is one PCLK Clock cycle. This means that the maximum frequency at which the counter can run in this mode is one-half of the PCLK Clock frequency. This clock source is not available in the capture modes (modes 2 and 4) because the TBn pin is used as one of the two capture inputs.

32.1.2.4 Pulse Accumulate Mode

The counter can also be configured to count prescaler output clock pulses when the TBn input is high and not count when the TBn input is low, as illustrated in Figure 32-3. The resulting count is an indicator of the cumulative time that the TBn input is high. This is called the "pulse-accumulate" mode. In this mode, an AND gate generates a clock signal for the counter whenever a prescaler clock pulse is generated and the TBn input is high. (The polarity of the TBn signal is programmable, so the counter can count when the TBn input is low rather than high.) The pulse-accumulate mode is not available in the capture modes (modes 2 and 4) because the TBn pin is used as one of the two capture inputs.

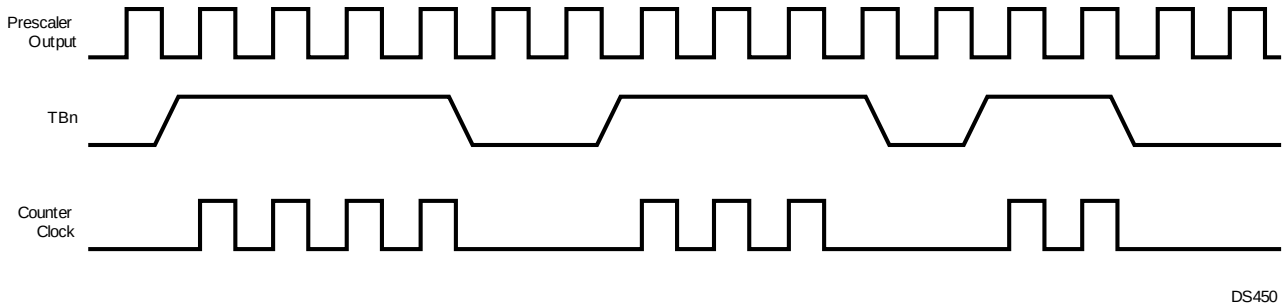


Figure 32-3. Pulse-Accumulate Mode

32.1.2.5 Slow Clock

Slow Clock can be selected as the clock source for the two 16-bit counters. Because Slow Clock can be asynchronous, it must be synchronized to PCLK Clock, therefore the maximum input frequency of Slow Clock is the PCLK Clock fre- quency divided by four.

32.1.2.6 Limitations in Low-Power Modes

The Power Save mode drives Slow Clock onto HCLK Clock. In this mode, Slow Clock cannot be used as a clock source for the timers because it would have the same frequency or a higher frequency than PCLK Clock, and the clock ratio needed for synchronization to PCLK Clock would not be maintained. However, the External Event Clock and Pulse Accumulate Mode will still work, as long as the external event pulses are at least the length of the PCLK Clock period.

Idle and Halt modes stop HCLK Clock and PCLK Clock. If PCLK Clock is stopped, the timer stops counting until PCLK Clock resumes operation.

32.2 Timer Operating Modes

Each timer/counter unit can be configured to operate in the following modes:

- Processor-Independent Pulse Width Modulation (PWM) mode
- Dual-Input Capture mode
- Dual Independent Timer mode
- Single-Input Capture and Single Timer mode

At reset, the timers are disabled. To configure and start the timers, software must write a set of values to the registers that control the timers. The registers are described in Section Section 32.4

32.2.1 Mode 1: Processor-Independent PWM

Mode 1 is the Processor-Independent Pulse Width Modulation (PWM) mode, which generates pulses of a specified width and duty cycle, and which also provides a separate general-purpose timer/counter.

Figure 32-4 is a block diagram of the Multi-Function Timer configured to operate in Mode 1. Timer/Counter 1 (TCNT1_n) functions as the time base for the PWM timer. It counts down at the clock rate selected for the counter. When an underflow occurs, the timer register is reloaded alternately from the TCRA_n and TCRB_n registers, and counting proceeds downward from the loaded value.

On the first underflow, the timer is loaded from the TCRA_n register, then from the TCRB_n register on the next underflow, then from the TCRA_n register again on the next underflow, and so on. Every time the counter is stopped and restarted, it always obtains its first reload value from the TCRA_n register. This is true whether the timer is restarted upon reset, after entering Mode 1 from another mode, or after stopping and restarting the clock with the Timer/Counter 1 clock selector.

The timer can be configured to toggle the TAn output bit on each underflow. This generates a clock signal on the TAn output with the width and duty cycle determined by the values stored in the TCRA_n and TCRB_n registers. This is a “processor-independent” PWM clock because once the timer is set up, no more action is required from the CPU to generate a continuous PWM signal.

The timer can be configured to generate separate interrupts upon reload from the TCRA_n and TCRB_n registers. The interrupts can be enabled or disabled under software control. The CPU can determine the cause of each interrupt by looking at the TAPND and TBPND bits, which are updated by the hardware on each occurrence of a timer reload.

In Mode 1, Timer/Counter 2 (TCNT2_n) can be used either as a simple system timer, an external event counter, or a pulse-accumulate counter. The clock counts down using the clock selected with the Timer/Counter 2 clock selector. It asserts an interrupt upon each underflow if the interrupt is enabled with the TDIEN bit.

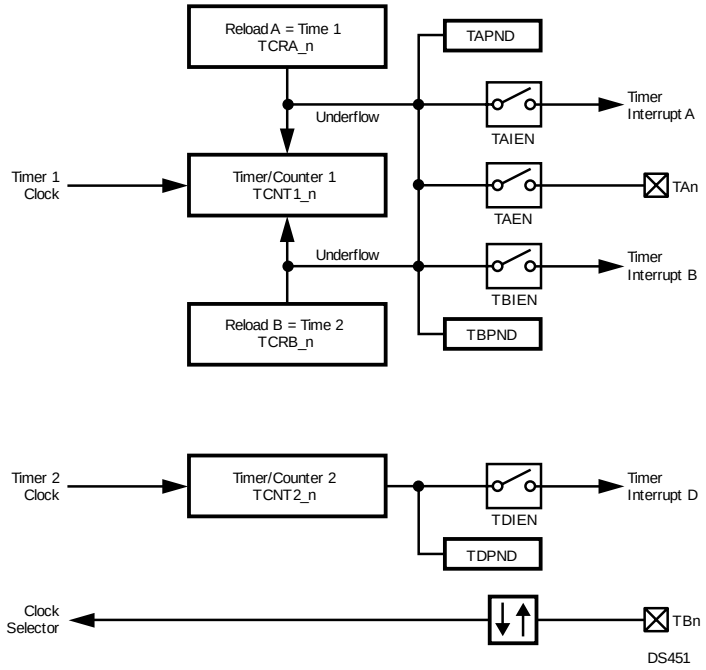


Figure 32-4. Processor-Independent PWM Mode

32.2.2 Mode 2: Dual Input Capture

Mode 2 is the Dual Input Capture mode, which measures the elapsed time between occurrences of external events, and which also provides a separate general-purpose timer/ counter.

Figure 32-5 is a block diagram of the Multi-Function Timer configured to operate in Mode 2. The time base of the capture timer depends on Timer/Counter 1, which counts down using the clock selected with the Timer/Counter 1 clock selector. The TAn and TBn pins function as capture inputs. A transition received on the TAn pin transfers the timer contents to the TCRA_n register. Similarly, a transition received on the TBn pin transfers the timer contents to the TCRB_n register. Each input pin can be configured to sense either rising or falling edges.

The TAn and TBn inputs can be configured to preset the counter to FFFFh on reception of a valid capture event. In this case, the current value of the counter is transferred to the corresponding capture register and then the counter is preset to FFFFh. Using this approach allows software to determine the on-time and off-time and period of an external signal with a minimum of CPU overhead.

The values captured in the TCRA_n register at different times reflect the elapsed time between transitions on the TAn pin. The same is true for the TCRB_n register and the TBn pin. The input signal on the TAn or TBn pin must have a pulse width equal to or greater than one PCLK Clock cycle.

There are three separate interrupts associated with the capture timer, each with its own enable bit and pending bit. The three interrupt events are reception of a transition on the TAn pin, reception of a transition on the TBn pin, and underflow of the TCNT1_n counter. The enable bits for these events are TAIEN, TBIEN, and TCIEN, respectively.

In Mode 2, Timer/Counter 2 (TCNT2_n) can be used as a simple system timer. The clock counts down using the clock selected with the Timer/Counter 2 clock selector. It asserts an interrupt upon each underflow if the interrupt is enabled with the TDIEN bit.

Neither Timer/Counter 1 (TCNT1_n) nor Timer/Counter 2 (TCNT2_n) can be configured to operate as an external event counter or to operate in the pulse-accumulate mode because the TBn input is used as a capture input. Attempting to select one of these configurations will cause one or both counters to stop.

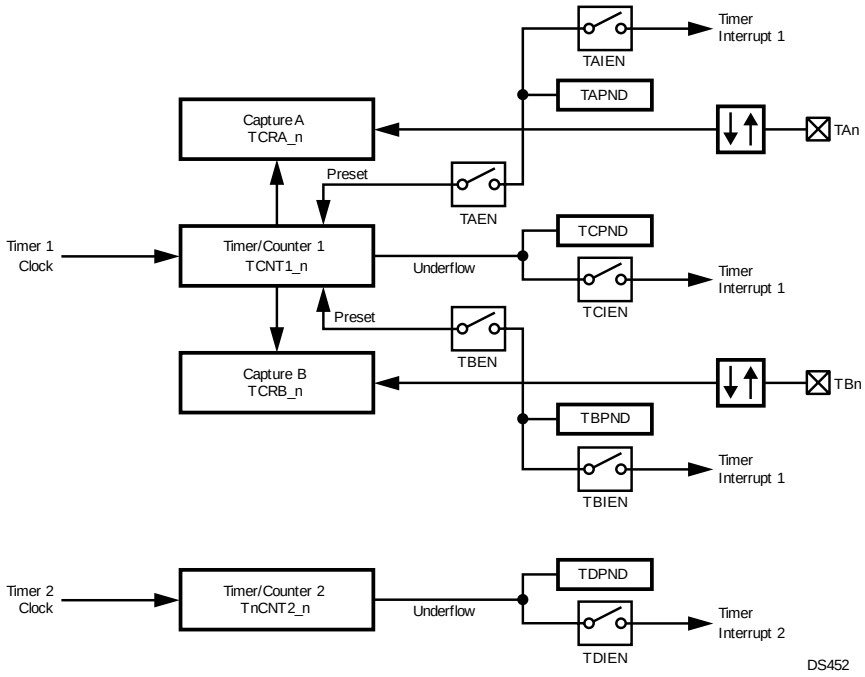


Figure 32-5. Dual-Input Capture Mode

32.2.3 Mode 3: Dual Independent Timer/Counter

Mode 3 is the Dual Independent Timer mode, which generates system timing signals or counts occurrences of exter- nal events.

Figure 32-6 is a block diagram of the Multi-Function Timer configured to operate in Mode 3. The timer is configured to operate as a dual independent system timer or dual external event counter. In addition, Timer/Counter 1 can generate a 50% duty cycle PWM signal on the TAn pin. The TBn pin can be used as an external event input or pulse-accumulate input and can be used as the clock source for either Timer/Counter 1 or Timer/Counter 2. Both counters can also be clocked by the prescaled PCLK Clock.

Timer/Counter 1 (TCNT1_n) counts down at the rate of the selected clock. On underflow, it is reloaded from the TCRA_n register and counting proceeds down from the reloaded value. In addition, the TAn pin is toggled on each underflow if this function is enabled by the TAEN bit. The initial state of the TAn pin is software-programmable. When the TAn pin is toggled from low to high, it sets the TCPND interrupt pending bit and also asserts an interrupt if enabled by the TAIEN bit.

Because the TAn pin toggles on every underflow, a 50% duty cycle PWM signal can be generated on the TAn pin without any further action from the CPU.

Timer/Counter 2 (TCNT2_n) counts down at the rate of the selected clock. On underflow, it is reloaded from the TCRB_n register and counting proceeds down from the reloaded value. In addition, each underflow sets the TDPND interrupt pending bit and asserts an interrupt if the interrupt is enabled by the TDIEN bit.

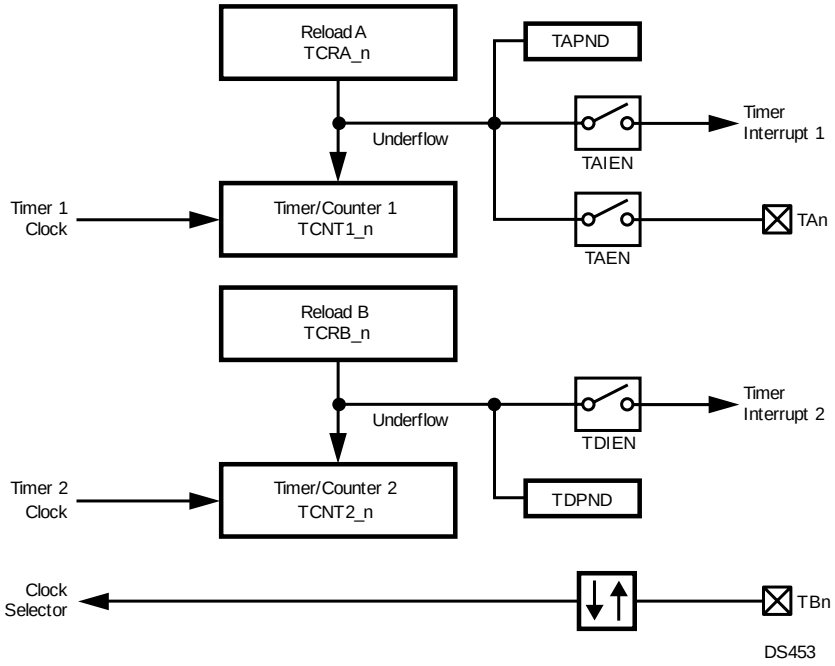


Figure 32-6. Dual-Independent Timer/Counter Mode

32.2.4 Mode 4: Input Capture Plus Timer

Mode 4 is the Single Input Capture and Single Timer mode, which provides one external event counter and one system timer.

Figure 32-7 is a block diagram of the Multi-Function Timer configured to operate in Mode 4. This mode offers a combination of Mode 3 and Mode 2 functions. Timer/Counter 1 is used as a system timer as in Mode 3 and Timer/Counter 2 is used as a capture timer as in Mode 2, but with a single input rather than two inputs.

Timer/Counter 1 (TCNT1_n) operates the same as in Mode 3. It counts down at the rate of the selected clock. On underflow, it is reloaded from the TCRA_n register and counting proceeds down from the reloaded value. The TAn pin is toggled on each underflow, when this function is enabled by the TAEN bit. When the TAn pin is toggled from low to high, it sets the TCPND interrupt pending bit and also asserts an interrupt if the interrupt is enabled by the TAIEN bit. A 50% duty cycle PWM signal can be generated on TAn without any further action from the CPU.

Timer/Counter 2 (TCNT1_n) counts down at the rate of the selected clock. The TBn pin functions as the capture input. A transition received on TBn transfers the timer contents to the TCRB_n register. The input pin can be configured to sense either rising or falling edges.

The TBn input can be configured to preset the counter to FFFFh on reception of a valid capture event. In this case, the current value of the counter is transferred to the capture register and then the counter is preset to FFFFh.

The values captured in the TCRB_n register at different times reflect the elapsed time between transitions on the TBn pin. The input signal on TBn must have a pulse width equal to or greater than one PCLK Clock cycle.

There are two separate interrupts associated with the capture timer, each with its own enable bit and pending bit. The two interrupt events are reception of a transition on TBn and underflow of the TCNT2_n counter. The enable bits for these events are TBIEN and TDIEN, respectively.

Neither Timer/Counter 1 (TCNT1_n) nor Timer/Counter 2 (TCNT2_n) can be configured to operate as an external event counter or to operate in the pulse-accumulate mode because the TBn input is used as a capture input. Attempting to select one of these configurations will cause one or both counters to stop. In this mode, Timer/Counter 2 must be enabled at all times.

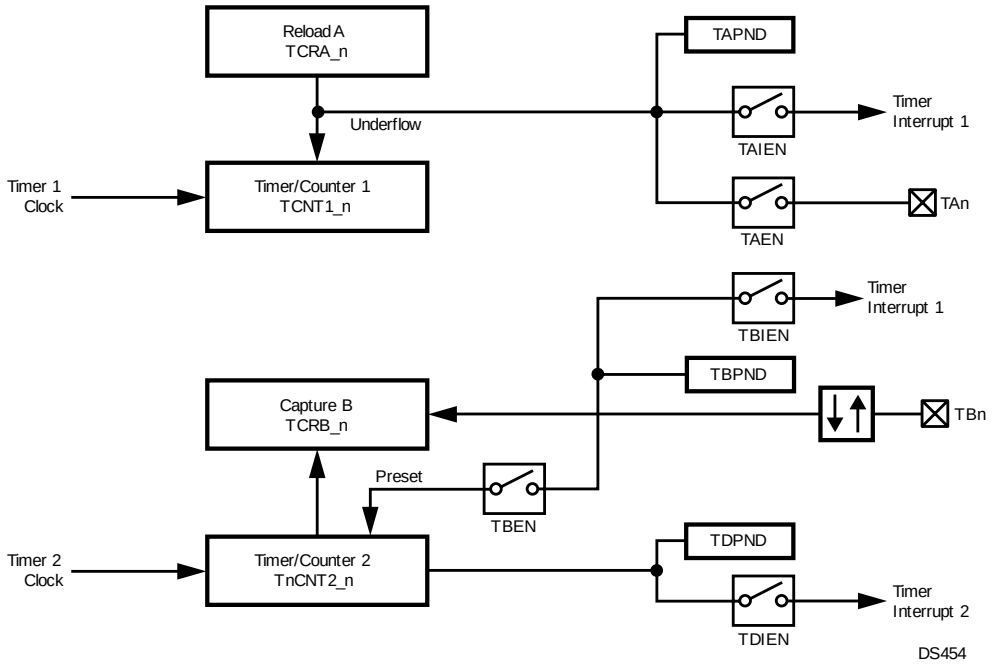


Figure 32-7. Input Capture Plus Timer Mode

32.3 Timer Interrupts

The Multi-Function Timer unit has four interrupt sources, designated A, B, C, and D. Interrupt sources A, B, and C are mapped into a single system interrupt called Timer Interrupt 1, while interrupt source D is mapped into a system interrupt called Timer Interrupt 2. Each of the four interrupt sources has its own enable bit and pending bit. The enable bits are named TAIEN, TBIEN, TCIEN, and TDIEN. The pending bits are named TAPND, TBPND, TCPND, and TDPND.

Timer Interrupts 1 and 2 are system interrupts TAn and TBn, respectively.

Table 32-1 shows the events that trigger interrupts A, B, C, and D in each of the four operating modes. Note that some interrupt sources are not used in some operating modes.

32.4 Timer I/O Functions

The Multi-Function Timer unit uses two I/O pins, called TAn and TBn. The function of each pin depends on the timer operating mode and the TAEN and TBEN enable bits. Table 32-2 shows the functions of the pins in each operating mode, and for each combination of enable bit settings.

When the TAn pin is configured to operate as a PWM output (TAEN = 1), the state of the pin is toggled on each underflow of the TCNT1_n counter. In this case, the initial value on the pin is determined by the TAOUT bit. For example, to start with TAn high, software must set the TAOUT bit before enabling the timer clock. This option is available only when the timer is configured to operate in Mode 1, 3, or 4 (in other words, when TCRA_n is not used in Capture mode).

Table 32-1. Timer Interrupts Overview

Sys. Int.	Interrupt Pending Bit	Mode 1 PWM + Counter	Mode 2 Dual Input Capture + Counter	Mode 3 Dual Counter	Mode 4 Single Capture + Counter
Timer Int. 1 (TAn Int.)	TAPND	TCNT1_n reload from TCRA_n	Input capture on TAn transition	TCNT1_n reload from TCRA_n	TCNT1_n reload from TCRA_n
	TBPND	TCNT1_n reload from TCRB_n	Input Capture on TBn transition	N/A	Input Capture on TBn transition
	TCPND	N/A	TCNT1_n underflow	N/A	N/A

Table 32-1. Timer Interrupts Overview (continued)

Sys. Int.	Interrupt Pending Bit	Mode 1 PWM + Counter	Mode 2 Dual Input Capture + Counter	Mode 3 Dual Counter	Mode 4 Single Capture + Counter
Timer Int. 2 (TBn Int.)	TDPND	TCNT2_n underflow	TCNT2_n underflow	TCNT2_n reload from TCRB_n	TCNT2_n underflow

Table 32-2. Timer I/O Functions

I/O	TAEN TBEN	Mode 1 PWM + Counter	Mode 2 Dual Input Capture + Counter	Mode 3 Dual Counter	Mode 4 Single Capture + Counter
TA	TAEN = 0 TBEN = X	No Output	Capture TCNT1 into TCRA	No Output Toggle	No Output Toggle
	TAEN = 1 TBEN = X	Toggle Output on Underflow of TCNT1	Capture TCNT1 into TCRA and Preset TCNT1	Toggle Output on Underflow of TCNT1	Toggle Output on Underflow of TCNT1
TB	TAEN = X TBEN = 0	Ext. Event or Pulse Accumulate Input	Capture TCNT1 into TCRB	Ext. Event or Pulse Accumulate Input	Capture TCNT2 into TCRB
	TAEN = X TBEN = 1	Ext. Event or Pulse Accumulate Input	Capture TCNT1 into TCRB and Preset TCNT1	Ext. Event or Pulse Accumulate Input	Capture TCNT2 into TCRB and Preset TCNT2

32.5 Timer Registers

Table 32-3 lists the CPU-accessible registers used to control the Multi-Function Timers.

Table 32-3. Multi-Function Timer Registers

NAME	ADDRESS	DESCRIPTION
TPRSC_0	FF 9010h	MFT0 Clock Prescaler Register
TPRSC_1	FF 6010h	RMFT1 Clock Prescale Register
TCKC_0	FF 9014h	MFT0 Clock Unit Control Register
TCKC_1	FF 6014h	MFT1 Clock Unit Control Register
TCNT1_0	FF 9000h	MFT0 Timer/Counter 1 Register
TCNT1_1	FF 6000h	MFT1 Timer/Counter 1 Register
TCNT2_0	FF 900Ch	MFT0 Timer/Counter 2 Register
TCNT2_1	FF 600Ch	MFT1 Timer/Counter 2 Register
TCRA_0	FF 9004h	MFT0 Reload/Capture A Register
TCRA_1	FF 6004h	MFT1 Reload/Capture A Register
TCRB_0	FF 9008h	MFT0 Reload/Capture B Register
TCRB_1	FF 6008h	MFT1 Reload/Capture B Register
TMCTRL_0	FF 9018h	MFT0 Timer Mode Control Register
TMCTRL_1	FF 6018h	MFT1 Timer Mode Control Register
TICTL_0	FF 901Ch	MFT0 Timer Interrupt Control Register
TICTL_1	FF 601Ch	MFT1 Timer Interrupt Control Register
TICLR_0	FF 9020h	MFT0 Timer Interrupt Clear Register
TICLR_1	FF 6020h	MFT1 Timer Interrupt Clear Register

32.5.1 Clock Prescaler Register n (TPRSC_n)

The TPRSC_n register is a byte-wide, read/write register that holds the current value of the 5-bit clock prescaler (CLKPS). This register is cleared on reset. The register format is shown below.

7	5	4	0
Reserved		CLKPS	

CLKPS The Clock Prescaler field specifies the divisor used to generate the Timer Clock from PCLK Clock. When the timer is configured to use the prescaled clock, PCLK Clock is divided by (CLKPS + 1) to produce the timer clock. Therefore, the PCLK Clock divisor can range from 1 to 32.

32.5.2 Clock Unit Control Register n (TCKC_n)

The TCKC_n register is a byte-wide, read/write register that selects the clock source for each timer/counter. Selecting the clock source also starts the counter. This register is cleared on reset, which disables the timer/counters. The register format is shown below.

7	6	5	3	2	0
Reserved		C2CSEL		C1CSEL	

C1CSEL The Counter 1 Clock Select field specifies the clock mode for Timer/Counter 1 as follows:
000 – No clock (Timer/Counter 1 stopped, modes 1, 2, and 3 only).
001 – Prescaled PCLK Clock.
010 – External event on TBn (modes 1 and 3 only).
011 – Pulse-accumulate mode based on TBn (modes 1 and 3 only).
100 – Slow Clock.
101 – Reserved.
110 – Reserved.
111 – Reserved.

C2CSEL The Counter 2 Clock Select field specifies the clock mode for Timer/Counter 2 as follows:
000 – No clock (Timer/Counter 2 stopped, modes 1, 2, and 3 only).
001 – Prescaled PCLK Clock.
010 – External event on TBn (modes 1 and 3 only).
011 – Pulse-accumulate mode based on TBn (modes 1 and 3 only).
100 – Slow Clock.
101 – Reserved.
110 – Reserved.
111 – Reserved.

32.5.3 Timer/Counter 1 Register n (TCNT1_n)

The TCNT1_n register is a 16-bit, read/write register that holds the current count value for Timer/Counter 1. The register contents are not affected by a reset and are unknown after power-up.

15	0
TCNT1	

32.5.4 Timer/Counter 2 Register n (TCNT2_n)

The TCNT2_n register is a 16-bit, read/write register that holds the current count value for Timer/Counter 2. The register contents are not affected by a reset and are unknown after power-up.

15	0
TCNT2	

32.5.5 Reload/Capture A Register n (TCRA_n)

The TCRA_n register is a 16-bit, read/write register that holds the reload or capture value for Timer/Counter 1. The register contents are not affected by a reset and are unknown after power-up.

15	0
TCRA	

32.5.6 Reload/Capture B Register n (TCRB_n)

The TCRB_n register is a 16-bit, read/write register that holds the reload or capture value for Timer/Counter 2. The register contents are not affected by a reset and are unknown after power-up.

15	0
TCRB	

32.5.7 Timer Mode Control Register n (TMCTRL_n)

The TMCTRL_n register is a byte-wide, read/write register that sets the operating mode of the timer/counter and the TAn and TBn pins. This register is cleared at reset. The register format is shown below.

7	6	5	4	3	2	1	0
TEN	TAOUT	TBEN	TAEN	TBEDG	TAEDG	MDSEL	

MDSEL The Mode Select field sets the operating mode of the timer/counter as follows:
00 – Mode 1: PWM plus system timer.
01 – Mode 2: Dual-Input Capture plus system timer.
10 – Mode 3: Dual Timer/Counter.
11 – Mode 4: Single-Input Capture and Single Timer.

TAEDG The TAn Edge Polarity bit selects the polarity of the edges that trigger the TAn input.
0 – TAn input is sensitive to falling edges (high to low transitions).
1 – TAn input is sensitive to rising edges (low to high transitions).

TBEDG The TBn Edge Polarity bit selects the polarity of the edges that trigger the TBn input. In pulse-accumulate mode, when this bit is set, the counter is enabled only when TBn is high; when this bit is clear, the counter is enabled only when TBn is low.
0 – TBn input is sensitive to falling edges (high to low transitions).
1 – TBn input is sensitive to rising edges (low to high transitions).

TAEN The TAn Enable bit controls whether the TAn pin is enabled to operate as a preset input or as a PWM output, depending on the timer operating mode. In Mode 2 (Dual Input Capture), a transition on the TAn pin presets the TCNT1_n counter to FFFFh. In the other modes, TAn functions as a PWM output. When this bit is clear, operation of the pin for the timer/counter is disabled.
0 – TAn input disabled.
1 – TAn input enabled.

TBEN The TBn Enable bit controls whether the TBn pin in enabled to operate in Mode 2 (Dual Input Capture) or Mode 4 (Single Input Capture and Single Timer). A transition on the TBn pin presets the corresponding timer/counter to FFFFh (TCNT1_n in Mode 2 or TCNT2_n in Mode 4). When this bit is clear, operation of the pin for the timer/counter is disabled. This bit setting has no effect in Mode 1 or Mode 3.
0 – TBn input disabled.
1 – TBn input enabled.

TAOUT The TAn Output Data bit indicates the current state of the TAn pin when the pin is used as a PWM output. The hardware sets and clears this bit, but software can also read or write this bit at any time and therefore control the state of the output pin. In case of conflict, a software write has precedence over a hardware update. This bit setting has no effect when the TAn pin is used as an input.
0 – TAn pin is low.
1 – TAn pin is high.

TEN The Timer Enable bit controls whether the Multi-Function Timer is enabled. When the module is disabled all clocks to the counter unit are stopped to minimize power consump- tion. For that reason, the timer/counter registers (TCNT1_n and TCNT2_n), the capture/ reload registers (TCRA_n and TCRB_n), and the interrupt pending bits (TXPND) cannot be written in this mode. Also, the 5-bit clock prescaler and the interrupt pending bits are cleared, and the TAn I/O pin is an input.
0 – Multi-Function Timer is disabled.
1 – Multi-Function Timer is enabled.

32.5.8 Timer Interrupt Control Register n (TICTL_n)

The TICTL_n register is a byte-wide, read/write register that contains the interrupt enable bits and interrupt pending bits for the four timer interrupt sources, designated A, B, C, and D. The condition that causes each type of interrupt depends on the operating mode, as shown in Table 32-1.

This register is cleared upon reset. The register format is shown below.

7	6	5	4	3	2	1	0
TDIEN	TCIEN	TBIEN	TAIEN	TDPND	TCPND	TBPND	TAPND

TAPND The Timer Interrupt Source A Pending bit indicates that timer interrupt condition A has occurred. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1. This bit can be set by hardware or by software. To clear this bit, software must use the Timer Interrupt Clear Register (TICLR_n). Attempting to directly write a 0 to this bit is ignored.
0 – Interrupt source A has not triggered.
1 – Interrupt source A has triggered.

TBPND The Timer Interrupt Source B Pending bit indicates that timer interrupt condition B has occurred. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1. This bit can be set by hardware or by software. To clear this bit, software must use the Timer Interrupt Clear Register (TICLR_n). Attempting to directly write a 0 to this bit is ignored.
0 – Interrupt source B has not triggered.
1 – Interrupt source B has triggered.

TCPND The Timer Interrupt Source C Pending bit indicates that timer interrupt condition C has occurred. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1. This bit can be set by hardware or by software. To clear this bit, software must use the Timer Interrupt Clear Register (TICLR_n). Attempting to directly write a 0 to this bit is ignored.
0 – Interrupt source C has not triggered.
1 – Interrupt source C has triggered.

TDPND The Timer Interrupt Source D Pending bit indicates that timer interrupt condition D has occurred. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1. This bit can be set by hardware or by software. To clear this bit, software must use the Timer Interrupt Clear Register (TICLR_n). Attempting to directly write a 0 to this bit is ignored.
0 – Interrupt source D has not triggered.
1 – Interrupt source D has triggered.

TAIEN The Timer Interrupt A Enable bit controls whether an interrupt is asserted on each occurrence of interrupt condition A. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1.
0 – Condition A interrupts disabled.
1 – Condition A interrupts enabled.

TBIEN The Timer Interrupt B Enable bit controls whether an interrupt is asserted on each occurrence of interrupt condition B. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1.
– Condition B interrupts disabled.
1 – Condition B interrupts enabled.

TCIEN The Timer Interrupt C Enable bit controls whether an interrupt is asserted on each occurrence of interrupt condition C. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1.
0 – Condition C interrupts disabled.
1 – Condition C interrupts enabled.

TDIEN The Timer Interrupt D Enable bit controls whether an interrupt is asserted on each occurrence of interrupt condition D. For an explanation of interrupt conditions A, B, C, and D, see Table 32-1.
0 – Condition D interrupts disabled.
1 – Condition D interrupts enabled.

32.5.9 Timer Interrupt Clear Register n (TICLR_n)

The TICLR_n register is a byte-wide, write-only register that allows software to clear the TAPND, TBPND, TCPND, and TDPND bits in the Timer Interrupt Control (TICTL_n) register. Do not modify this register with instructions that access the register as a read-modify-write operand, such as the bit manipulation instructions. The register reads as FFh. The register format is shown below.

7 4	3	2	1	0
Reserved	TDCLR	TCCLR	TBCLR	TACLR

TACLR The Timer Pending A Clear bit is used to clear the Timer Interrupt Source A Pending bit (TAPND) in the Timer Interrupt Control regi ter (TICTL).
0 – Writing a 0 has no effect.
1 – Writing a 1 clears the TAPND bit.

TBCLR The Timer Pending A Clear bit is used to clear the Timer Interrupt Source B Pending bit (TBPND) in the Timer Interrupt Control register (TICTL).
0 – Writing a 0 has no effect.
1 – Writing a 1 clears the TBPND bit.

TCCLR The Timer Pending C Clear bit is used to clear the Timer Interrupt Source C Pending bit (TCPND) in the Timer Interrupt Control register (TICTL_n).
0 – Writing a 0 has no effect.
1 – Writing a 1 clears the TCPND bit.

TDCLR The Timer Pending D Clear bit is used to clear the Timer Interrupt Source D Pending bit (TDPND) in the Timer Interrupt Control register (TICTL_n).
0 – Writing a 0 has no effect.
1 – Writing a 1 clears the TDPND bit.

33 Dual Versatile Timer Units (VTU)

Each Versatile Timer Unit (VTU) contains four fully independent 16-bit timer subsystems. Each timer subsystem can operate either as dual 8-bit PWM timers, as a single 16-bit PWM timer, or as a 16-bit counter with 2 input capture channels. These timer subsystems offers an 8-bit clock prescaler to accommodate a wide range of system frequencies.

Each of the two VTUs provided on the CP3SP33 offers the following features:

- The VTU can be configured to provide:
  - Eight fully independent 8-bit PWM channels
  - Four fully independent 16-bit PWM channels
  - Eight 16-bit input capture channels
- The VTU consists of four timer subsystems, each of which contains:
  - A 16-bit counter
  - Two 16-bit capture / compare registers
  - An 8-bit fully programmable clock prescaler
- Each of the four timer subsystems can operate in the following modes:
  - Low power mode, i.e., all clocks are stopped
  - Dual 8-bit PWM mode
  - 16-bit PWM mode
  - Dual 16-bit input capture mode
- The VTU controls a total of eight I/O pins, each of which can function as either:
  - PWM output with programmable output polarity
  - Capture input with programmable event detection and timer reset
- A flexible interrupt scheme with
  - Four separate system level interrupt requests
  - A total of 16 interrupt sources each with a separate interrupt pending bit and interrupt enable bit

33.1 VTU Functional Description

Each VTU is comprised of four timer subsystems. Each timer subsystem contains an 8-bit clock prescaler, a 16-bit up-counter, and two 16-bit registers. Each timer subsystem controls two I/O pins which either function as PWM outputs or capture inputs depending on the mode of operation. There are four system-level interrupt requests, one for each timer subsystem. Each system-level interrupt request is controlled by four interrupt pending bits with associated enable/disable bits. All four timer subsystems are fully independent, and each may operate as a dual 8-bit PWM timer, a 16-bit PWM timer, or as a dual 16-bit capture timer. Figure 33-1 shows the main elements of the VTU.

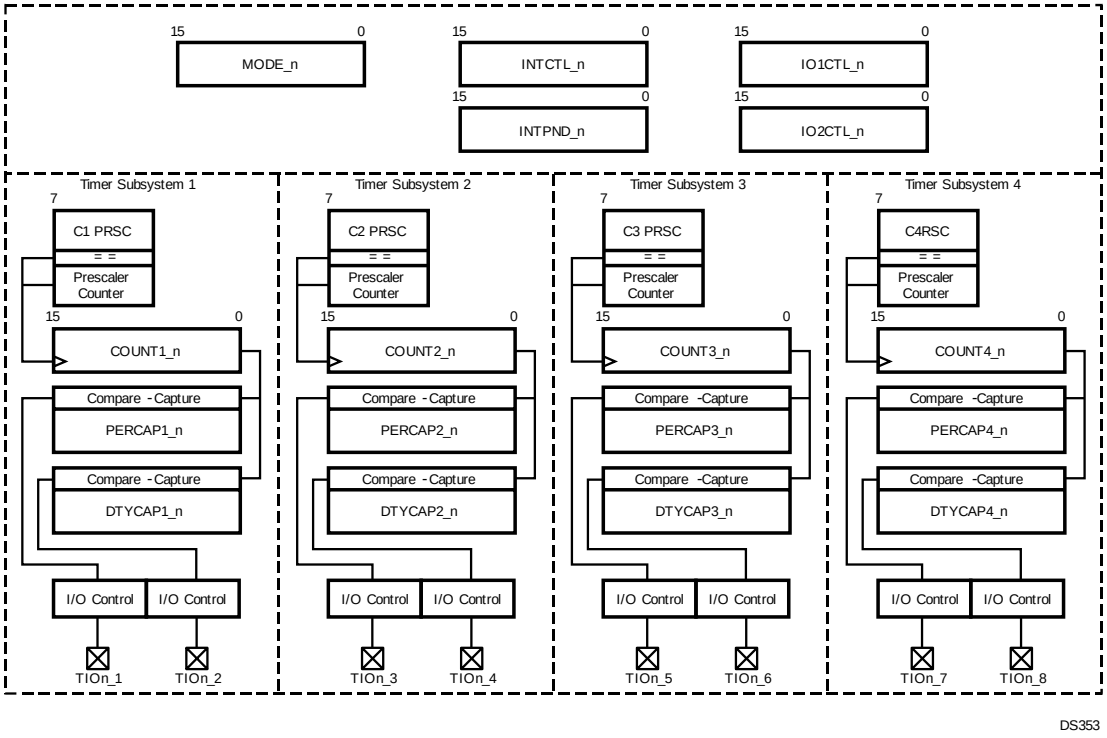


Figure 33-1. Versatile Timer Unit Block Diagram

33.1.1 Dual 8-bit PWM Mode

Each timer subsystem may be configured to generate two fully independent PWM waveforms on the respective TIOx pins. In this mode, the counter COUNTx_n is split and operates as two independent 8-bit counters. (In this section, n is the VTU module number which is either 0 or 1, and x is the VTU subsystem number which may be 1 to 4.) Each counter increments at the rate determined by the clock prescaler.

Each of the two 8-bit counters may be started and stopped separately using the corresponding TxRUN bits. Once either of the two 8-bit timers is running, the clock prescaler starts counting. Once the clock prescaler counter value matches the value of the associated CxPRSC register field, COUNTx_n is incremented.

The period of the PWM output waveform is determined by the value of the PERCAPx_n register. The TIOn_y output starts at the default value as programmed in the IOxCTL_n.PxPOL bit. (In this section, y is the TIOn I/O number, which may be 1 to 8.) Once the counter value reaches the value of the period register PERCAPx_n, the counter is cleared on the next counter increment. On the following increment from 00h to 01h, the TIOn_y output will change to the opposite of the default value.

The duty cycle of the PWM output waveform is controlled by the DTYCAPx_n register value. Once the counter value reaches the value of the duty cycle register DTYCAPx_n, the PWM output TIOn_y changes back to its default value on the next counter increment. Figure 33-2 illustrates this concept.

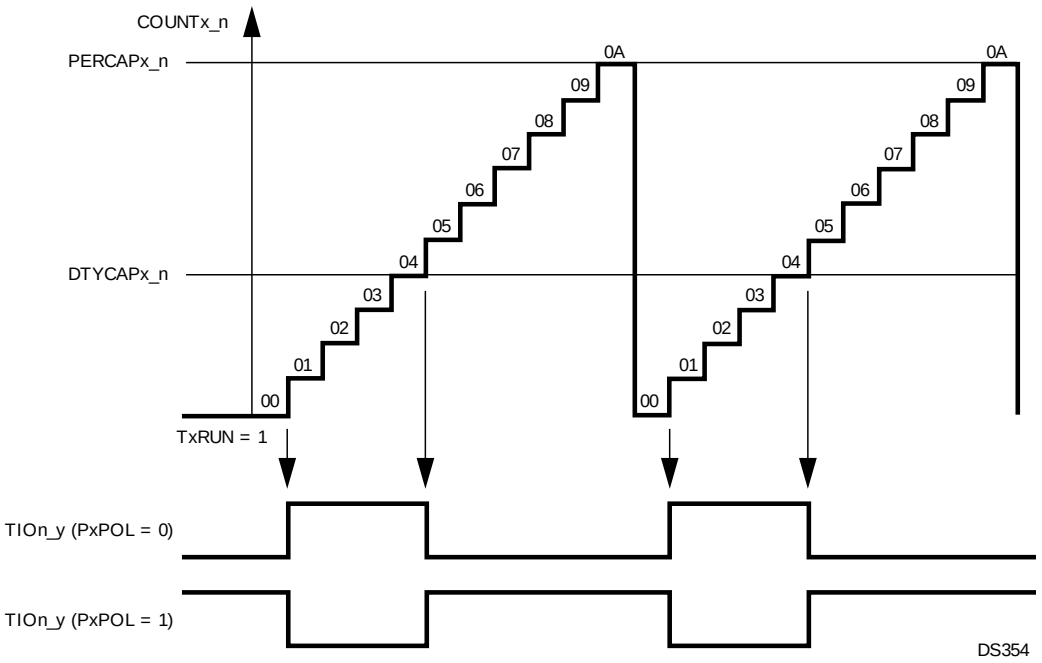


Figure 33-2. VTU PWM Generation

The period time is determined by the following formula:

PWM Period = (PERCAPx_n + 1) × (CxPRSC + 1) × T_{CLK}

The duty cycle in percent is calculated as follows:

Duty Cycle = (DTYCAPx_n / (PERCAPx_n + 1)) × 100

If the duty cycle register (DTYCAPx_n) holds a value which is greater than the value held in the period register (PERCAPx_n) the TIOn_y output will remain at the opposite of its default value which corresponds to a duty cycle of 100%. If the duty cycle register (DTYCAPx_n) register holds a value of 00h, the TIOn_y output will remain at the default value which corresponds to a duty cycle of 0%, in which case the value in the PERCAPx_n register is irrelevant. This scheme allows the duty cycle to be programmed in a range from 0% to 100%.

In order to allow fully synchronized updates of the period and duty cycle compare values, the PERCAPx_n and DTYCAPx_n registers are double buffered when operating in PWM mode. Therefore, if software writes to either the period or duty cycle register while either of the two PWM channels is enabled, the new value will not take effect until the counter value matches the previous period value or the timer is stopped.

Reading the PERCAPx_n or DTYCAPx_n register will always return the most recent value written to it.

The counter registers can be written if both 8-bit counters are stopped. This allows software to preset the counters before starting, which can be used to generate PWM output waveforms with a phase shift relative to each other. If the counter is written with a value other than 00h, it will start incrementing from that value. The TIOn_y output will remain at its default value until the first 00h to 01h transition of the counter value occurs. If the counter is preset to values which are less than or equal to the value held in the period register (PERCAPx_n) the counter will count up until a match between the counter value and the PERCAPx_n register value occurs. The counter will then be cleared and continue counting up. Alternatively, the counter may be written with a value which is greater than the value held in the period register. In that case the counter will count up to FFh, then roll over to 00h. In any case, the TIOn_y pin always changes its state at the 00h to 01h transition of the counter.

Software may only write to the COUNTx_n register if both TxRUN bits of a timer subsystem are clear. Any writes to the counter register while either timer is running will be ignored.

The two I/O pins associated with a timer subsystem function are independent PWM outputs in the dual 8-bit PWM mode. If a PWM timer is stopped using its associated MODE_n.TxRUN bit the following actions result:

- The associated TIOn_y pin will return to its default value as defined by the IOxCTL_n.PxPOL bit.
- The counter will stop and will retain its last value.
- Any pending updates of the PERCAPx_n and DTYCAPx_n register will be completed.
- The prescaler counter will be stopped and reset if both MODE_n.TxRUN bits are cleared.

Figure 33-3 illustrates the configuration of a timer subsystem while operating in dual 8-bit PWM mode. The numbering in Figure 33-3 refers to timer subsystem 1 but equally applies to the other three timer subsystems.

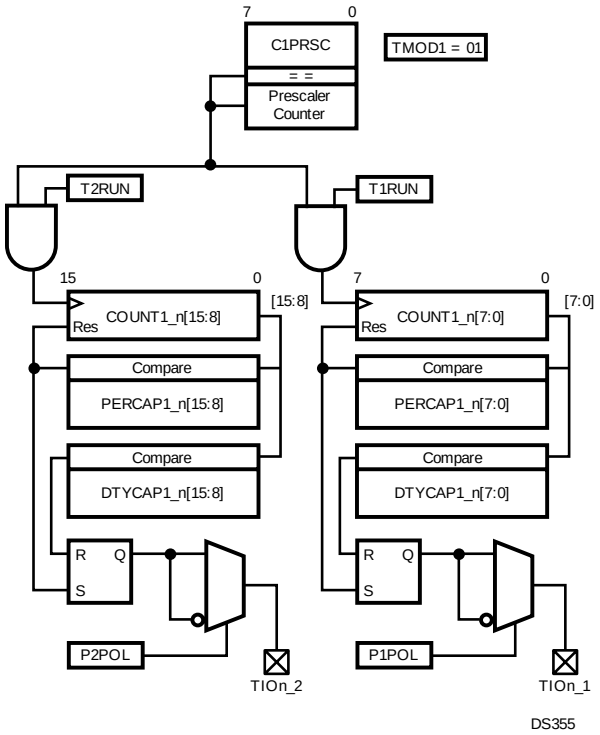


Figure 33-3. VTU Dual 8-Bit PWM Mode

33.1.2 16-Bit PWM Mode

Each of the four timer subsystems may be independently configured to provide a single 16-bit PWM channel. In this case the lower and upper bytes of the counter are concatenated to form a single 16-bit counter.

Operation in 16-bit PWM mode is conceptually identical to the dual 8-bit PWM operation as outlined under Dual 8-bit PWM Mode on page 254. The 16-bit timer may be started or stopped with the lower MODE_n.TxRUN bit, i.e., T1RUN for timer subsystem 1.

The two TIOn_y outputs associated with a timer subsystem can be used to produce either two identical PWM waveforms or two PWM waveforms of opposite polarities. This can be accomplished by setting the two PxPOL bits of the respective timer subsystem to either identical or opposite values.

Figure 33-4 illustrates the configuration of a timer subsystem while operating in 16-bit PWM mode. The numbering in Figure 33-4 refers to timer subsystem 1 but equally applies to the other three timer subsystems.

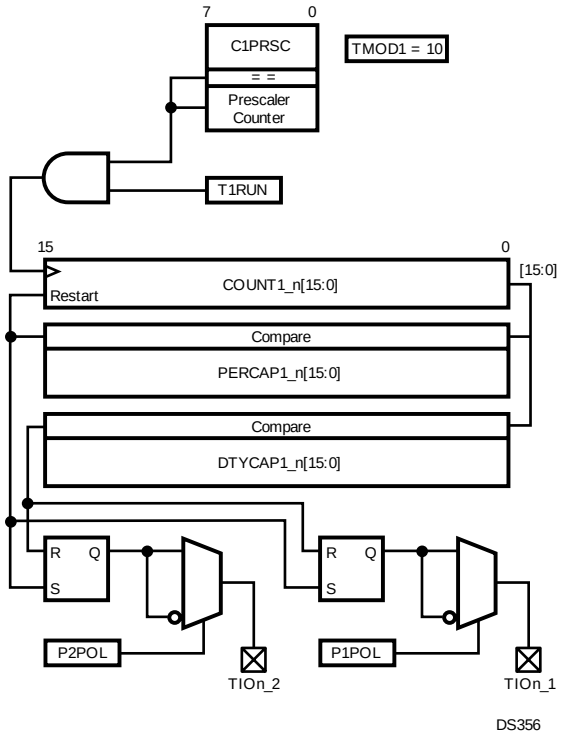


Figure 33-4. VTU 1-bit PWM Mode

33.1.3 Dual 16-Bit Capture Mode

In addition to the two PWM modes, each timer subsystem may be configured to operate in an input capture mode which provides two 16-bit capture channels. The input capture mode can be used to precisely measure the period and duty cycle of external signals.

In capture mode the counter COUNTx_n operates as a 16- bit up-counter while the two TIOn_y pins associated with a timer subsystem operate as capture inputs. A capture event on the TIOn_y pins causes the contents of the counter register (COUNTx_n) to be copied to the PERCAPx_n or DTYCAPx_n registers, respectively.

Starting the counter is identical to the 16-bit PWM mode, i.e., setting the lower of the two MODE_n.TxRUN bits will start the counter and the clock prescaler. In addition, the capture event inputs are enabled once the MODE_n.TxRUN bit is set.

The TIOn_y capture inputs can be independently configured to detect a capture event on either a positive transition, a negative transition or both a positive and a negative transition. In addition, any capture event may be used to reset the counter COUNTx_n and the clock prescaler counter. This avoids the need for software to keep track of timer overflow conditions and greatly simplifies the direct frequency and duty cycle measurement of an external signal.

Figure 33-5 illustrates the configuration of a timer subsystem while operating in capture mode. The numbering in Figure 33-5 refers to timer subsystem 1 but equally applies to the other three timer subsystems.

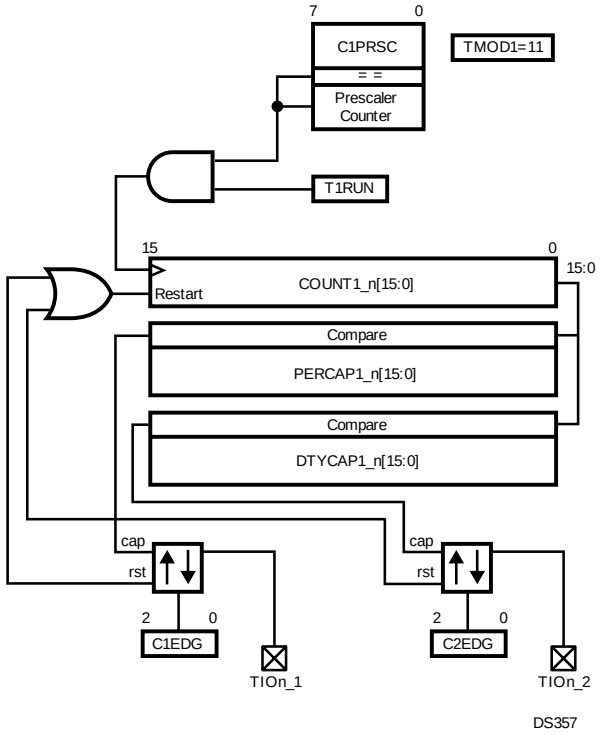


Figure 33-5. VTU Dual 16-Bit Capture Mode

33.1.4 Low Power Mode

If a timer subsystem is not used, software can place it in a low-power mode. All clocks to a timer subsystem are stopped and the counter and prescaler contents are frozen once low-power mode is entered. Software may continue to write to the MODE_n, INTCTL_n, IOxCTL_n, and CLKxPS_n registers. Write operations to the INTPND_n register are allowed; but if a timer subsystem is in low-power mode, its associated interrupt pending bits cannot be cleared. Software cannot write to the COUNTx_n, PERCAPx_n, and DTYCAPx_n registers of a timer subsystem while it is in low-power mode. All registers can be read at any time.

33.1.5 Interrupts

Each VTU has a total of 16 interrupt sources, four for each of the four timer subsystems. All interrupt sources have a pending bit and an enable bit associated with them. All interrupt pending bits are denoted IxAPD through IxDPD where “x” relates to the specific timer subsystem. There is one system level interrupt request for each of the four timer subsystems.

Figure 33-6 illustrates the interrupt structure of the versatile timer module.

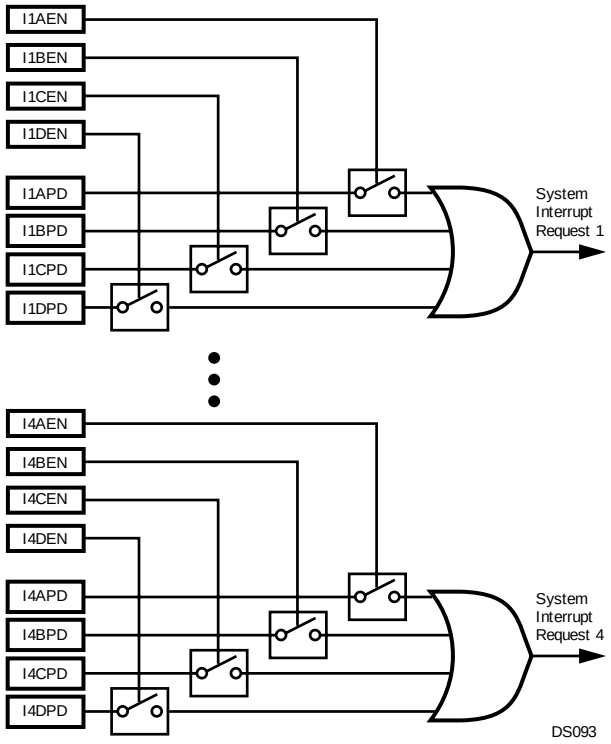


Figure 33-6. VTU Interrupt Request Structure

Each of the timer pending bits, IxAPD through IxDPD, is set by a specific hardware event depending on the mode of operation, i.e., PWM or Capture mode. Table 33-1 outlines the specific hardware events relative to the operation mode which cause an interrupt pending bit to be set.

Table 33-1. VTU Interrupt Sources

PENDING FLAG	DUAL 8-BIT PWM MODE	16-BIT PWM MODE	CAPTURE MODE
IxAPD	Low Byte Duty Cycle match	Duty Cycle match	Capture to PERCAPx_n
IxBPD	Low Byte Period match	Period match	Capture to DTYCAPx_n
IxCPD	High Byte Duty Cycle match	N/A	Counter Overflow
IxDPD	High Byte Period match	N/A	N/A

33.1.6 Freeze Mode

If Freeze is mode is entered, all timer counter clocks will be inhibited and the current value of the timer registers will be frozen; in capture mode, all further capture events are disabled. Once Freeze mode is exited, counting will resume from the previous value and the capture input events are reenabled.

33.2 VTU Registers

Each of the two VTUs has a total of 19 user-accessible registers, as listed in Table 33-2. All registers are word-wide and are initialized to a known value upon reset. All software accesses to the VTU registers must be word accesses.

Table 33-2. VTU Registers

NAME	ADDRESS	DESCRIPTION
MODE_0	FF 8800h	Mode Control Register Module 0
MODE_1	FF 8C00h	Mode Control Register Module 1

Table 33-2. VTU Registers (continued)

NAME	ADDRESS	DESCRIPTION
IO1CTL_0	FF 8804h	I/O Control Register 1 Module 0
IO1CTL_1	FF 8C04h	I/O Control Register 1 Module 1
IO2CTL_0	FF 8808h	I/O Control Register 2 Module 0
IO2CTL_1	FF 8C08h	I/O Control Register 2 Module 1
INTCTL_0	FF 880Ch	Interrupt Control Register Module 0
INTCTL_1	FF 8C0Ch	Interrupt Control Register Module 1
INTPND_0	FF 8810h	Interrupt Pending Register Module 0
INTPND_1	FF 8C10h	Interrupt Pending Register Module 1
CLK1PS_0	FF 8814h	Clock Prescaler Register 1 Module 0
CLK1PS_1	FF 8C14h	Clock Prescaler Register 1 Module 1
CLK2PS_0	FF 8830h	Clock Prescaler Register 2 Module 0
CLK2PS_1	FF 8C30h	Clock Prescaler Register 2 Module 1
COUNT1_0	FF 8818h	Counter 1 Register Module 0
COUNT1_1	FF 8C18h	Counter 1 Register Module 1
PERCAP1_0	FF 881Ch	Period/Capture 1 Register Module 0
PERCAP1_1	FF 8C1Ch	Period/Capture 1 Register Module 1
DTYCAP1_0	FF 8820h	Duty Cycle/Capture 1 Register Module 0
DTYCAP1_1	FF 8C20h	Duty Cycle/Capture 1 Register Module 1
COUNT2_0	FF 8824h	Counter 2 Register Module 0
COUNT2_1	FF 8C24h	Counter 2 Register Module 1
PERCAP2_0	FF 8828h	Period/Capture 2 Register Module 0
PERCAP2_1	FF 8C28h	Period/Capture 2 Register Module 1
DTYCAP2_0	FF 882Ch	Duty Cycle/Capture 2 Register Module 0
DTYCAP2_1	FF 8C2Ch	Duty Cycle/Capture 2 Register Module 1
COUNT3_0	FF 8834h	Counter 3 Register Module 0
COUNT3_1	FF 8C34h	Counter 3 Register Module 1
PERCAP3_0	FF 8838h	Period/Capture 3 Register Module 0
PERCAP3_1	FF 8C38h	Period/Capture 3 Register Module 1
DTYCAP3_0	FF 883Ch	Duty Cycle/Capture 3 Register Module 0
DTYCAP3_1	FF 8C3Ch	Duty Cycle/Capture 3 Register Module 1
COUNT4_0	FF 8840h	Counter 4 Register Module 0
COUNT4_1	FF 8C40h	Counter 4 Register Module 1
PERCAP4_0	FF 8844h	Period/Capture 4 Register Module 0
PERCAP4_1	FF 8C44h	Period/Capture 4 Register Module 1
DTYCAP4_0	FF 8848h	Duty Cycle/Capture 4 Register Module 0
DTYCAP4_1	FF 8C48h	Duty Cycle/Capture 4 Register Module 1

33.2.1 Mode Control Register Module n (MODE_n)

The MODE_n registers are 16-bit, read/write registers which control the mode selection of the four timer subsystems in each module. The registers are clear after reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMOD4		T8RUN	T7RUN	TMOD3		T6RUN	T5RUN	TMOD2		T4RUN	T3RUN	TMOD1		T2RUN	T1RUN

TxRUN The Timer Run bit controls whether the corresponding timer is stopped or running. If set, the associated counter and clock prescaler is started depending on the mode of operation. Once set, the clock to the clock prescaler and the counter are enabled and the counter will increment each time the clock prescaler counter value matches the value defined in the associated clock prescaler field (CxPRSC).

0 – Timer stopped.

1 – Timer running.

TMODx The Timer System Operating Mode field enables or disables the Timer Subsystem and defines its operating mode.

00 – Low-Power Mode. All clocks to the counter subsystem are stopped. The counter is stopped regardless of the value of the TxRUN bits. Read operations to the Timer Subsystem will return the last value; software must not perform any write operations to the Timer Subsystem while it is disabled since those will be ignored.

01 – Dual 8-bit PWM mode. Each 8-bit counter may individually be started or stopped via its associated TxRUN bit. The TIOn_y pins will function as PWM outputs.

10 – 16-bit PWM mode. The two 8-bit counters are concatenated to form a single 16-bit counter. The counter may be started or stopped with the lower of the two TxRUN bits, i.e,. T1RUN, T3RUN, T5RUN, and T7RUN. The TIOn_y pins will function as PWM outputs.

11 – Capture Mode. Both 8-bit counters are concatenated and operate as a single 16-bit counter. The counter may be started or stopped with the lower of the two TxRUN bits, i.e., T1RUN, T3RUN, T5RUN, and T7RUN. The TIOn_y pins will function as capture inputs.

33.2.2 I/O Control Register 1 Module n (IO1CTL_n)

The IO1CTL_n registers are 16-bit, read/write registers. The registers control the I/O pins TIOn_1 through TIOn_4 depending on the selected mode of operation. The registers are clear after reset.

15	14 12	11	10 8	7	6 4	3	2 0
P4POL	C4EDG	P3POL	C3EDG	P2POL	C2EDG	P1POL	C2EDG

CxEDG The Capture Edge Control field specifies the polarity of a capture event and the reset of the counter. The value of this three bit field has no effect while operating in PWM mode.

CxEDG	CAPTURE	COUNTER RESET
000	Rising edge	No
001	Falling edge	No
010	Rising edge	Yes
011	Falling edge	Yes
100	Both edges	No
101	Both edges	Rising edge
110	Both edges	Falling edge
111	Both edges	Both edges

PxPOL The PWM Polarity bit selects the output polarity. While operating in PWM mode the bit specifies the polarity of the corresponding PWM output (TIOn_y). Once a counter is stopped, the output will assume the value of PxPOL, i.e., its initial value. The PxPOL bit has no effect while operating in capture mode.

0 – The PWM output goes high at the 00h to 01h transition of the counter and will go low once the counter value matches the duty cycle value.

1 – The PWM output goes low at the 00h to 01h transition of the counter and will go high once the counter value matches the duty cycle value.

33.2.3 I/O Control Register 2 Module n (IO2CTL_n)

The IO2CTL_n registers are 16-bit, read/write registers. The registers control the I/O pins TIOn_5 through TIOn_8 depending on the selected mode of operation. The registers are cleared at reset.

15	14 12	11	10 8	7	6 4	3	2 0
P6POL	C8EDG	P7POL	C7EDG	P6POL	C6EDG	P5POL	C5EDG

The functionality of the bit fields of the IO2CTL_n register is identical to the ones described in the IO1CTL_n register section.

33.2.4 Interrupt Control Register Module n (INTCTL_n)

The INTCTL_n registers are 16-bit, read/write registers. They contain the interrupt enable bits for the 16 interrupt sources of each VTU. Each interrupt enable bit corresponds to an interrupt pending bit located in the Interrupt Pending Register (INTPND_n). All INTCTL_n register bits are solely under software control. The registers are clear after reset.

7	6	5	4	3	2	1	0
I2DEN	I2CEN	I2BEN	I2AEN	I1DEN	I1CEN	I1BEN	I1AEN

15	14	13	12	11	10	9	8
I4DEN	I4CEN	I4BEN	I4AEN	I3DEN	I3CEN	I3BEN	I3AEN

IxAEN The Timer x Interrupt A Enable bit controls interrupt requests triggered on the corresponding IxAPD bit being set. The associated IxAPD bit will be updated regardless of the value of the IxAEN bit.
0 – Disable system interrupt request for the IxAPD pending bit.
1 – Enable system interrupt request for the IxAPD pending bit.

IxBEN The Timer x Interrupt B Enable bit controls interrupt requests triggered on the corresponding IxBPD bit being set. The associated IxBPD bit will be updated regardless of the value of the IxBEN bit.
0 – Disable system interrupt request for the IxBPD pending bit.
1 – Enable system interrupt request for the IxBPD pending bit.

IxCEN The Timer x Interrupt C Enable bit controls interrupt requests triggered on the corresponding IxCPD bit being set. The associated IxCPD bit will be updated regardless of the value of the IxCEN bit.
0 – Disable system interrupt request for the IxCPD pending bit.
1 – Enable system interrupt request for the IxCPD pending bit.

IxDEN Timer x Interrupt D Enable bit controls interrupt requests triggered on the corresponding IxDPD bit being set. The associated IxDPD bit will be updated regardless of the value of the IxDEN bit.
0 – Disable system interrupt request for the IxDPD pending bit.
1 – Enable system interrupt request for the IxDPD pending bit.

33.2.5 Interrupt Pending Register Module n (INTPND_n)

The INTPND_n registers are 16-bit, read/write registers which contain all 16 interrupt pending bits. There are four interrupt pending bits called IxAPD through IxDPD for each timer subsystem. Each interrupt pending bit is set by a hardware event and can be cleared if software writes a 1 to the bit position. The value will remain unchanged if a 0 is written to the bit position. All interrupt pending bits are cleared at reset.

7	6	5	4	3	2	1	0
I2DPD	I2CPD	I2BPD	I2APD	I1DPD	I1CPD	I1BPD	I1APD

15	14	13	12	11	10	9	8
I4DPD	I4CPD	I4BPD	I4APD	I3DPD	I3CPD	I3BPD	I3APD

IxAPD The Timer x Interrupt A Pending bit indicates that an interrupt condition for the related timer subsystem has occurred. Table 33-1 lists the hardware condition which causes this bit to be set.
0 – No interrupt pending.
1 – Timer interrupt condition occurred.

IxBPD The Timer x Interrupt B Pending bit indicates that an interrupt condition for the related timer subsystem has occurred. Table 33-1 lists the hardware condition which causes this bit to be set.
0 – No interrupt pending.
1 – Timer interrupt condition occurred.

IxCPD The Timer x Interrupt C Pending bit indicates that an interrupt condition for the related timer subsystem has occurred. Table 33-1 lists the hardware condition which causes this bit to be set.
0 – No interrupt pending.
1 – Timer interrupt condition occurred.

IxDPD The Timer x Interrupt D Pending bit indicates that an interrupt condition for the related timer subsystem has occurred. Table 33-1 lists the hardware condition which causes this bit to be set.
0 – No interrupt pending.
1 – Timer interrupt condition occurred.

33.2.6 Clock Prescaler Register 1 Module n (CLK1PS_n)

The CLK1PS_n registers are 16-bit, read/write registers. The registers are split into two 8-bit fields called C1PRSC and C2PRSC. Each field holds the 8-bit clock prescaler compare value for timer subsystems 1 and 2 respectively. The registers are cleared at reset.

15	8	7	0
C2PRSC		C1PRSC	

C1PRSC The Clock Prescaler 1 Compare Value field holds the 8-bit prescaler value for timer subsystem 1. The counter of timer subsystem is incremented each time when the clock prescaler compare value matches the value of the clock prescaler counter. The division ratio is equal to (C1PRSC + 1). For example, 00h is a ratio of 1, and FFh is a ratio of 256.

C2PRSC The Clock Prescaler 2 Compare Value field holds the 8-bit prescaler value for timer subsystem 2. The counter of timer subsystem is incremented each time when the clock prescaler compare value matches the value of the clock prescaler counter. The division ratio is equal to (C2PRSC + 1).

33.2.7 Clock Prescaler Register 2 Module n (CLK2PS_n)

The CLK2PS_n registers are 16-bit, read/write registers. The registers are split into two 8-bit fields called C3PRSC and C4PRSC. Each field holds the 8-bit clock prescaler compare value for timer subsystems 3 and 4 respectively. The registers are cleared at reset.

15	8	7	0
C3PRSC		C4PRSC	

C4PRSC The Clock Prescaler 3 Compare Value field holds the 8-bit prescaler value for timer subsystem 3. The counter of timer subsystem is incremented each time when the clock prescaler compare value matches the value of the clock prescaler counter. The division ratio is equal to (C3PRSC + 1).

C3PRSC The Clock Prescaler 4 Compare Value field holds the 8-bit prescaler value for timer subsystem 4. The counter of timer subsystem is incremented each time when the clock prescaler compare value matches the value of the clock prescaler counter. The division ratio is equal to (C4PRSC + 1).

33.2.8 Counter Register x Module n (COUNTx_n)

The COUNTx_n registers are 16-bit, read/write registers. There are four registers in each module, called COUNT1_n through COUNT4_n, one for each of the four timer subsystems. Software may read the registers at any time. Reading the register will return the current value of the counter. The register may only be written if the counter is stopped (i.e., if both TxRUN bits associated with a timer sub- system are clear). The registers are cleared at reset.

15	0
CNTx	

33.2.9 Period/Capture Register x Module n (PERCAPx_n)

The PERCAPx_n registers are 16-bit, read/write registers. There are four registers called PERCAP1_n through PERCAP4_n, one for each timer subsystem. The registers hold the period compare value in PWM mode of the counter value at the time the last associated capture event occurred. In PWM mode the register is double buffered. If a new period compare value is written while the counter is running, the write will not take effect until counter value matches the previous period compare value or until the counter is stopped. Reading may take place at any time and will return the most recent value which was written. The PERCAPx_n registers are cleared at reset.

15	0
PCAPx	

33.2.10 Duty Cycle/Capture Register x Module n (DTYCAPx_n)

The DTYCAPx_n registers are 16-bit, read/write registers. There are four registers called DTYCAP1_n through DTYCAP4_n, one for each timer subsystem. The registers hold the period compare value in PWM mode or the counter value at the time the last associated capture event occurred. In PWM mode, the register is double buffered. If a new duty cycle compare value is written while the counter is running, the write will not take effect until the counter value matches the previous period compare value or until the counter is stopped. The update takes effect on period boundaries only. Reading may take place at any time and will return the most recent value which was written. The DTYCAPx_n registers are cleared at reset.

15	0
DCAPx	

34 Register Map

Table 34-1 is a detailed memory map showing the specific memory address of the memory, I/O ports, and registers. The table shows the starting address, the size, and a brief description of each memory block and register. For detailed information on using these memory locations, see the applicable sections in the data sheet.

All addresses not listed in the table are reserved and must not be read or written. An attempt to access an unlisted address will have unpredictable results.

Each byte-wide register occupies a single address and can be accessed only in a byte-wide transaction. Each wordwide register occupies two consecutive memory addresses and can be accessed only in a word-wide transaction. Both the byte-wide and word-wide registers reside at word boundaries (even addresses). Therefore, each byte-wide register uses only the lowest eight bits of the internal data bus.

Most device registers are read/write registers. However, some registers are read-only or write-only, as indicated in the table. An attempt to read a write-only register or to write a read-only register will have unpredictable results.

When software writes to a register in which one or more bits are reserved, it must write a zero to each reserved bit unless indicated otherwise in the description of the register. Reading a reserved bit returns an undefined value.

Table 34-1. Detailed Device Mapping

Register Name	Size	Address	Access Type	Value After Reset	Comments
Bluetooth LLC Registers					
PLN	Byte	01 2000h	Write-Only		
WHITENING_CHANNEL_SELECTION	Byte	01 2001h	Write-Only		
SINGLE_FREQUENCY_SELECTION	Byte	01 2002h	Write-Only		
CORRELATOR_CLOCK_SEL	Byte	01 2003h	Write-Only		
LN_BT_CLOCK_0	Byte	01 2018h	Read-Only		
LN_BT_CLOCK_1	Byte	01 2019h	Read-Only		
LN_BT_CLOCK_2	Byte	01 201Ah	Read-Only		
LN_BT_CLOCK_3	Byte	01 201Bh	Read-Only		
RX_CN	Byte	01 201Ch	Read-Only		
TX_CN	Byte	01 201Dh	Read-Only		
AC_ACCEPTLVL	Word	01 201Eh	Write-Only		
LAP_ACCEPTLVL	Byte	01 2020h	Write-Only		
RFSYNCH_DELAY	Byte	01 2021h	Write-Only		
SPI_READ	Word	01 2022h	Read-Only		
SPI_MODE_CONFIG	Byte	01 2024h	Write-Only		
M_COUNTER_0	Byte	01 2026h	Read/Write		
M_COUNTER_1	Byte	01 2027h	Read/Write		
M_COUNTER_2	Byte	01 2028h	Read/Write		
N_COUNTER_0	Byte	01 202Ah	Write-Only		
N_COUNTER_1	Byte	01 202Bh	Write-Only		
BT_CLOCK_WR_0	Byte	01 202Ch	Write-Only		
BT_CLOCK_WR_1	Byte	01 202Dh	Write-Only		
BT_CLOCK_WR_2	Byte	01 202Eh	Write-Only		
BT_CLOCK_WR_3	Byte	01 202Fh	Write-Only		
WTPTC_1SLOT	Word	01 2030h	Write-Only		
WTPTC_3SLOT	Word	01 2032h	Write-Only		
WTPTC_5SLOT	Word	01 2034h	Write-Only		
SEQ_RESET	Byte	01 2036h	Write-Only		
SEQ_CONTINUE	Byte	01 2037h	Write-Only		
RX_STATUS	Byte	01 2038h	Read-Only		
CHIP_ID	Byte	01 203Ah	Read-Only		
INT_VECTOR	Byte	01 203Ch	Read-Only		
SYSTEM_CLK_EN	Byte	01 203Eh	Write-Only		
LINKTIMER_WR_RD	Word	01 2040h	Read-Only		
LINKTIMER_SELECT	Byte	01 2042h	Read-Only		
LINKTIMER_STATUS_EXP_FLAG	Byte	01 2044h	Read-Only		
LINKTIMER_STATUS_RD_WR_FLAG	Byte	01 2045h	Read-Only		
LINKTIMER_ADJUST_PLUS	Byte	01 2046h	Read-Only		
LINKTIMER_ADJUST_MINUS	Byte	01 2047h	Read-Only		
SLOTTIMER_WR_RD	Byte	01 2048h	Read-Only		
RX_CRC	Byte	01 204Ah	Read-Only		
RX_CRC	Byte	01 204Bh	Read-Only		
AFH_PLN_SEL	Byte	01 204Ch	Read/Write		
AFH_SAME_CN_CONFIG	Byte	01 204Dh	Read/Write		
AFH_CN_MAP_0	Byte	01 204Eh	Read/Write		
AFH_CN_MAP_1	Byte	01 204Fh	Read/Write		

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
AFH_CN_MAP_2	Byte	01 2050h	Read/Write		
AFH_CN_MAP_3	Byte	01 2051h	Read/Write		
AFH_CN_MAP_4	Byte	01 2052h	Read/Write		
AFH_CN_MAP_5	Byte	01 2053h	Read/Write		
AFH_CN_MAP_6	Byte	01 2054h	Read/Write		
AFH_CN_MAP_7	Byte	01 2055h	Read/Write		
AFH_CN_MAP_8	Byte	01 2056h	Read/Write		
AFH_CN_MAP_9	Byte	01 2057h	Read/Write		
USB Registers					
FADDR	Byte	FF 0800h	Read/Write	00h	
POWER	Byte	FF 0801h	Read/Write	20h	
INTRTX	Word	FF 0802h	Read-Only	0000h	
INTRRX	Word	FF 0804h	Read-Only	0000h	
INTRTXE	Word	FF 0806h	Read/Write	000Fh	
INTRRXE	Word	FF 0808h	Read/Write	000Eh	
INTRUSB	Byte	FF 080Ah	Read-Only	00h	
INTRUSBE	Byte	FF 080Bh	Read/Write	06h	
FRAME	Word	FF 080Ch	Read-Only	0000h	
INDEX	Byte	FF 080Eh	Read/Write	00h	
INDEXED	8-Word	FF 0810h-FF 081Fh	Read/Write		
EP0FIFO	Dword	FF 0820h	Read/Write		
EP1FIFO	Dword	FF 0824h	Read/Write		
EP2FIFO	Dword	FF 0828h	Read/Write		
EP3FIFO	Dword	FF 082Ch	Read/Write		
DEVCTL	Byte	FF 0860h	Read/Write	00h	
EP0NIND	8-Word	FF 0900h-FF 090Fh	Read/Write		
EP1NIND	8-Word	FF 0910h-FF 091Fh	Read/Write		
EP2NIND	8-Word	FF 0920h-FF 092Fh	Read/Write		
EP3NIND	8-Word	FF 0930h-FF 093Fh	Read/Write		
VCTRL	Word	FF 0C00h	Write-Only	0000h	
VSTATUS	Byte	FF 0C02h	Read-Only	00h	XXX
CAN Module 0 Message Buffers					
C0MB0_CNSTAT	Word	FF B800h	Read/Write	xxxxh	
C0MB0_TSTP	Word	FF B804h	Read/Write	xxxxh	
C0MB0_DATA3	Word	FF B808h	Read/Write	xxxxh	
C0MB0_DATA2	Word	FF B80Ch	Read/Write	xxxxh	
C0MB0_DATA1	Word	FF B810h	Read/Write	xxxxh	
C0MB0_DATA0	Word	FF B814h	Read/Write	xxxxh	
C0MB0_ID0	Word	FF B818h	Read/Write	xxxxh	
C0MB0_ID1	Word	FF B81Ch	Read/Write	xxxxh	
C0MB1	16- word	FF B820h-FF B83Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB2	16- word	FF B840h-FF B85Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB3	16- word	FF B860h-FF B87Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB4	16- word	FF B880h-FF B89Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB5	16- word	FF B8A0h-FF B8BFh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB6	16- word	FF B8C0h-FF B8DFh	Read/Write	xxxxh	Same register layout as C0MB0.

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
C0MB7	16- word	FF B8E0h-FF B8FFh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB8	16- word	FF B900h-FF B91Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB9	16- word	FF B892h-FF B93Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB10	16- word	FF B940h-FF B95Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB11	16- word	FF B960h-FF B97Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB12	16- word	FF B980-FF B99Fh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB13	16- word	FF B9A0h-FF B9BFh	Read/Write	xxxxh	Same register layout as C0MB0.
C0MB14	16- word	FF B9C0h-FF B9DFh	Read/Write	xxxxh	Same register layout as C0MB0.
CAN Module 0 Registers					
CGCR0	Word	FF BA00h	Read/Write	0000h	
CTIM0	Word	FF BA04h	Read/Write	0000h	
GMSKX0	Word	FF BA08h	Read/Write	0000h	
GMSKB0	Word	FF BA0Ch	Read/Write	0000h	
BMSKX0	Word	FF BA10h	Read/Write	0000h	
BMSKB0	Word	FF BA14h	Read/Write	0000h	
CIEN0	Word	FF BA18h	Read/Write	0000h	
CIPND0	Word	FF BA1Ch	Read-Only	0000h	
CICLR0	Word	FF BA20h	Write-Only	0000h	
CICEN0	Word	FF BA24h	Read/Write	0000h	
CSTPND0	Word	FF BA28h	Read-Only	0000h	
CANEC0	Word	FF BA2Ch	Read-Only	0000h	
CEDIAG0	Word	FF BA30h	Read-Only	0000h	
CTMR0	Word	FF BA34h	Read-Only	0000h	
CAN Module 1 Message Buffers					
C1MB0_CNSTAT	Word	FF BC00h	Read/Write	xxxxh	
C1MB0_TSTP	Word	FF BC04h	Read/Write	xxxxh	
C1MB0_DATA3	Word	FF BC08h	Read/Write	xxxxh	
C1MB0_DATA2	Word	FF BC0Ch	Read/Write	xxxxh	
C1MB0_DATA1	Word	FF BC10h	Read/Write	xxxxh	
C1MB0_DATA0	Word	FF BC14h	Read/Write	xxxxh	
C1MB0_ID0	Word	FF BC18h	Read/Write	xxxxh	
C1MB0_ID1	Word	FF BC1Ch	Read/Write	xxxxh	
C1MB1	16- word	FF BC20h - FF BC3Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB2	16- word	FF BC40h - FF BC5Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB3	16- word	FF BC60h - FF BC7Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB4	16- word	FF BC80h - FF BC9Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB5	16- word	FF BCA0h - FF BCBFh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB6	16- word	FF BCC0h - FF BCDFh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB7	16- word	FF BCE0h - FF BCFFh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB8	16- word	FF BD00h - FF BD1Fh	Read/Write	xxxxh	Same register layout as C1MB0.

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
C1MB9	16- word	FF B892h - FF BD3Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB10	16- word	FF BD40h - FF BD5Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB11	16- word	FF BD60h - FF BD7Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB12	16- word	FF BD80h - FF BD9Fh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB13	16- word	FF BDA0h - FF BDBFh	Read/Write	xxxxh	Same register layout as C1MB0.
C1MB14	16- word	FF BDC0h - FF BDDFh	Read/Write	xxxxh	Same register layout as C1MB0.
CAN Module 1 Registers					
CGCR1	Word	FF BE00h	Read/Write	0000h	
CTIM1	Word	FF BE04h	Read/Write	0000h	
GMSKX1	Word	FF BE08h	Read/Write	0000h	
GMSKB1	Word	FF BE0Ch	Read/Write	0000h	
BMSKX1	Word	FF BE10h	Read/Write	0000h	
BMSKB1	Word	FF BE14h	Read/Write	0000h	
CIEN1	Word	FF BE18h	Read/Write	0000h	
CIPND1	Word	FF BE1Ch	Read Only	0000h	
CICLR1	Word	FF BE20h	Write Only	0000h	
CICEN1	Word	FF BE24h	Read/Write	0000h	
CSTPND1	Word	FF BE28h	Read Only	0000h	
CANEC1	Word	FF BE2Ch	Read Only	0000h	
CEDIAG1	Word	FF BE30h	Read Only	0000h	
CTMR1	Word	FF BE34h	Read Only	0000h	
Bus Arbiter					
MASTGP	Dword	FF F000h	Read/Write	0000 0000h	
ARBALGO	Dword	FF F004h	Read/Write	0000 0001h	
DFTMASK	Dword	FF F008h	Read/Write	0000 0002h	
ARBCFGLK	Dword	FF F00Ch	Read/Write	0000 0000h	
DMA Controller					
ADCA0	Dword	FF 0400h	Read/Write	0000 0000h	
ADRA0	Dword	FF 0404h	Read/Write	0000 0000h	
ADCB0	Dword	FF 0408h	Read/Write	0000 0000h	
ADRB0	Dword	FF 040Ch	Read/Write	0000 0000h	
BLTC0	Dword	FF 0410h	Read/Write	0000 0000h	
BLTR0	Dword	FF 0414h	Read/Write	0000 0000h	
RQTR0	Dword	FF 0418h	Read/Write	0000 0000h	
RQTCNT0	Dword	FF 041Ch	Read/Write	0000 0000h	
DMACNT0	Dword	FF 0420h	Read/Write	000C 0000h	
DMASTAT0	Dword	FF 0424h	Read/Write	0000 0000h	
ADCA1	Dword	FF 0440h	Read/Write	0000 0000h	
ADRA1	Dword	FF 0444h	Read/Write	0000 0000h	
ADCB1	Dword	FF 0448h	Read/Write	0000 0000h	
ADRB1	Dword	FF 044Ch	Read/Write	0000 0000h	
BLTC1	Dword	FF 0450h	Read/Write	0000 0000h	
BLTR1	Dword	FF 0454h	Read/Write	0000 0000h	
RQTR1	Dword	FF 0458h	Read/Write	0000 0000h	
RQTCNT1	Dword	FF 045Ch	Read/Write	0000 0000h	
DMACNT1	Dword	FF 0460h	Read/Write	000C 0000h	
DMASTAT1	Dword	FF 0464h	Read/Write	0000 0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
ADCA2	Dword	FF 0480h	Read/Write	0000 0000h	
ADRA2	Dword	FF 0484h	Read/Write	0000 0000h	
ADCB2	Dword	FF 0488h	Read/Write	0000 0000h	
ADRB2	Dword	FF 048Ch	Read/Write	0000 0000h	
BLTC2	Dword	FF 0490h	Read/Write	0000 0000h	
BLTR2	Dword	FF 0494h	Read/Write	0000 0000h	
RQTR2	Dword	FF 0498h	Read/Write	0000 0000h	
RQTCNT2	Dword	FF 049Ch	Read/Write	0000 0000h	
DMACNT2	Dword	FF 04A0h	Read/Write	000C 0000h	
DMASTAT2	Dword	FF 04A4h	Read/Write	0000 0000h	
ADCA3	Dword	FF 04C0h	Read/Write	0000 0000h	
ADRA3	Dword	FF 04C4h	Read/Write	0000 0000h	
ADCB3	Dword	FF 04C8h	Read/Write	0000 0000h	
ADRB3	Dword	FF 04CCh	Read/Write	0000 0000h	
BLTC3	Dword	FF 04D0h	Read/Write	0000 0000h	
BLTR3	Dword	FF 04D4h	Read/Write	0000 0000h	
RQTR3	Dword	FF 04D8h	Read/Write	0000 0000h	
RQTCNT3	Dword	FF 04DCh	Read/Write	0000 0000h	
DMACNT3	Dword	FF 04E0h	Read/Write	000C 0000h	
DMASTAT3	Dword	FF 04E4h	Read/Write	0000 0000h	
ADCA4	Dword	FF 0500h	Read/Write	0000 0000h	
ADRA4	Dword	FF 0504h	Read/Write	0000 0000h	
ADCB4	Dword	FF 0508h	Read/Write	0000 0000h	
ADRB4	Dword	FF 050Ch	Read/Write	0000 0000h	
BLTC4	Dword	FF 0510h	Read/Write	0000 0000h	
BLTR4	Dword	FF 0514h	Read/Write	0000 0000h	
RQTR4	Dword	FF 0518h	Read/Write	0000 0000h	
RQTCNT4	Dword	FF 051Ch	Read/Write	0000 0000h	
DMACNT4	Dword	FF 0520h	Read/Write	000C 0000h	
DMASTAT4	Dword	FF 0524h	Read/Write	0000 0000h	
ADCA5	Dword	FF 0540h	Read/Write	0000 0000h	
ADRA5	Dword	FF 0544h	Read/Write	0000 0000h	
ADCB5	Dword	FF 0548h	Read/Write	0000 0000h	
ADRB5	Dword	FF 054Ch	Read/Write	0000 0000h	
BLTC5	Dword	FF 0550h	Read/Write	0000 0000h	
BLTR5	Dword	FF 0554h	Read/Write	0000 0000h	
RQTR5	Dword	FF 0558h	Read/Write	0000 0000h	
RQTCNT5	Dword	FF 055Ch	Read/Write	0000 0000h	
DMACNT5	Dword	FF 0560h	Read/Write	000C 0000h	
DMASTAT5	Dword	FF 0564h	Read/Write	0000 0000h	
ADCA6	Dword	FF 0580h	Read/Write	0000 0000h	
ADRA6	Dword	FF 0584h	Read/Write	0000 0000h	
ADCB6	Dword	FF 0588h	Read/Write	0000 0000h	
ADRB6	Dword	FF 058Ch	Read/Write	0000 0000h	
BLTC6	Dword	FF 0590h	Read/Write	0000 0000h	
BLTR6	Dword	FF 0594h	Read/Write	0000 0000h	
RQTR6	Dword	FF 0598h	Read/Write	0000 0000h	
RQTCNT6	Dword	FF 059Ch	Read/Write	0000 0000h	
DMACNT6	Dword	FF 05A0h	Read/Write	000C 0000h	
DMASTAT6	Dword	FF 05A4h	Read/Write	0000 0000h	
ADCA7	Dword	FF 05C0h	Read/Write	0000 0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
ADRA7	Dword	FF 05C4h	Read/Write	0000 0000h	
ADCB7	Dword	FF 05C8h	Read/Write	0000 0000h	
ADRB7	Dword	FF 05CCh	Read/Write	0000 0000h	
BLTC7	Dword	FF 05D0h	Read/Write	0000 0000h	
BLTR7	Dword	FF 05D4h	Read/Write	0000 0000h	
RQTR7	Dword	FF 05D8h	Read/Write	0000 0000h	
RQTCNT7	Dword	FF 05DCh	Read/Write	0000 0000h	
DMACNT7	Dword	FF 05E0h	Read/Write	000C 0000h	
DMASTAT7	Dword	FF 05E4h	Read/Write	0000 0000h	
ADCA8	Dword	FF 0600h	Read/Write	0000 0000h	
ADRA8	Dword	FF 0604h	Read/Write	0000 0000h	
ADCB8	Dword	FF 0608h	Read/Write	0000 0000h	
ADRB8	Dword	FF 060Ch	Read/Write	0000 0000h	
BLTC8	Dword	FF 0610h	Read/Write	0000 0000h	
BLTR8	Dword	FF 0614h	Read/Write	0000 0000h	
RQTR8	Dword	FF 0618h	Read/Write	0000 0000h	
RQTCNT8	Dword	FF 061Ch	Read/Write	0000 0000h	
DMACNT8	Dword	FF 0620h	Read/Write	000C 0000h	
DMASTAT8	Dword	FF 0624h	Read/Write	0000 0000h	
ADCA9	Dword	FF 0640h	Read/Write	0000 0000h	
ADRA9	Dword	FF 0644h	Read/Write	0000 0000h	
ADCB9	Dword	FF 0648h	Read/Write	0000 0000h	
ADRB9	Dword	FF 064Ch	Read/Write	0000 0000h	
BLTC9	Dword	FF 0650h	Read/Write	0000 0000h	
BLTR9	Dword	FF 0654h	Read/Write	0000 0000h	
RQTR9	Dword	FF 0658h	Read/Write	0000 0000h	
RQTCNT9	Dword	FF 065Ch	Read/Write	0000 0000h	
DMACNT9	Dword	FF 0660h	Read/Write	000C 0000h	
DMASTAT9	Dword	FF 0664h	Read/Write	0000 0000h	
ADCA10	Dword	FF 0680h	Read/Write	0000 0000h	
ADRA10	Dword	FF 0684h	Read/Write	0000 0000h	
ADCB10	Dword	FF 0688h	Read/Write	0000 0000h	
ADRB10	Dword	FF 068Ch	Read/Write	0000 0000h	
BLTC10	Dword	FF 0690h	Read/Write	0000 0000h	
BLTR10	Dword	FF 0694h	Read/Write	0000 0000h	
RQTR10	Dword	FF 0698h	Read/Write	0000 0000h	
RQTCNT10	Dword	FF 069Ch	Read/Write	0000 0000h	
DMACNT10	Dword	FF 06A0h	Read/Write	000C 0000h	
DMASTAT10	Dword	FF 06A4h	Read/Write	0000 0000h	
ADCA11	Dword	FF 06C0h	Read/Write	0000 0000h	
ADRA11	Dword	FF 06C4h	Read/Write	0000 0000h	
ADCB11	Dword	FF 06C8h	Read/Write	0000 0000h	
ADRB11	Dword	FF 06CCh	Read/Write	0000 0000h	
BLTC11	Dword	FF 06D0h	Read/Write	0000 0000h	
BLTR11	Dword	FF 06D4h	Read/Write	0000 0000h	
RQTR11	Dword	FF 06D8h	Read/Write	0000 0000h	
RQTCNT11	Dword	FF 06DCh	Read/Write	0000 0000h	
DMACNT11	Dword	FF 06E0h	Read/Write	000C 0000h	
DMASTAT11	Dword	FF 06E4h	Read/Write	0000 0000h	
ADCA12	Dword	FF 0700h	Read/Write	0000 0000h	
ADRA12	Dword	FF 0704h	Read/Write	0000 0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
ADCB12	Dword	FF 0708h	Read/Write	0000 0000h	
ADRB12	Dword	FF 070Ch	Read/Write	0000 0000h	
BLTC12	Dword	FF 0710h	Read/Write	0000 0000h	
BLTR12	Dword	FF 0714h	Read/Write	0000 0000h	
RQTR12	Dword	FF 0718h	Read/Write	0000 0000h	
RQTCNT12	Dword	FF 071Ch	Read/Write	0000 0000h	
DMACNT12	Dword	FF 0720h	Read/Write	000C 0000h	
DMASTAT12	Dword	FF 0724h	Read/Write	0000 0000h	
ADCA13	Dword	FF 0740h	Read/Write	0000 0000h	
ADRA13	Dword	FF 0744h	Read/Write	0000 0000h	
ADCB13	Dword	FF 0748h	Read/Write	0000 0000h	
ADRB13	Dword	FF 074Ch	Read/Write	0000 0000h	
BLTC13	Dword	FF 0750h	Read/Write	0000 0000h	
BLTR13	Dword	FF 0754h	Read/Write	0000 0000h	
RQTR13	Dword	FF 0758h	Read/Write	0000 0000h	
RQTCNT13	Dword	FF 075Ch	Read/Write	0000 0000h	
DMACNT13	Dword	FF 0760h	Read/Write	000C 0000h	
DMASTAT13	Dword	FF 0764h	Read/Write	0000 0000h	
ADCA14	Dword	FF 0780h	Read/Write	0000 0000h	
ADRA14	Dword	FF 0784h	Read/Write	0000 0000h	
ADCB14	Dword	FF 0788h	Read/Write	0000 0000h	
ADRB14	Dword	FF 078Ch	Read/Write	0000 0000h	
BLTC14	Dword	FF 0790h	Read/Write	0000 0000h	
BLTR14	Dword	FF 0794h	Read/Write	0000 0000h	
RQTR14	Dword	FF 0798h	Read/Write	0000 0000h	
RQTCNT14	Dword	FF 079Ch	Read/Write	0000 0000h	
DMACNT14	Dword	FF 07A0h	Read/Write	000C 0000h	
DMASTAT14	Dword	FF 07A4h	Read/Write	0000 0000h	
ADCA15	Dword	FF 07C0h	Read/Write	0000 0000h	
ADRA15	Dword	FF 07C4h	Read/Write	0000 0000h	
ADCB15	Dword	FF 07C8h	Read/Write	0000 0000h	
ADRB15	Dword	FF 07CCh	Read/Write	0000 0000h	
BLTC15	Dword	FF 07D0h	Read/Write	0000 0000h	
BLTR15	Dword	FF 07D4h	Read/Write	0000 0000h	
RQTR15	Dword	FF 07D8h	Read/Write	0000 0000h	
RQTCNT15	Dword	FF 07DCh	Read/Write	0000 0000h	
DMACNT15	Dword	FF 07E0h	Read/Write	000C 0000h	
DMASTAT15	Dword	FF 07E4h	Read/Write	0000 0000h	
External Bus Interface Unit					
SMCTLR	Dword	FF 00A4h	Read/Write	0000 0401h	
SCSLR0	Dword	FF 0014h	Read/Write	0040 0000h	
SMSKR0	Dword	FF 0054h	Read/Write	0000 0028h	
SCSLR1	Dword	FF 0018h	Read/Write	0080 0000h	
SMSKR1	Dword	FF 0058h	Read/Write	0000 0128h	
SCSLR2	Dword	FF 001Ch	Read/Write	0100 0000h	
SMSKR2	Dword	FF 005Ch	Read/Write	0000 0229h	
SMTMGR_SET0	Dword	FF 0094h	Read/Write	0002 0D45h	
SMTMGR_SET1	Dword	FF 0098h	Read/Write	0012 0D45h	
SMTMGR_SET2	Dword	FF 009Ch	Read/Write	001A 0D45h	
FLASH_TPRDR	Dword	FF 00A0h	Read/Write	0000 00C8h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
System Configuration					
MCFG	Byte	FF F400h	Read/Write	00h	
MSTAT	Byte	FF F404h	Read-Only	ENV2:0 pins	Bit 6 clear on write
SWRESET	Byte	FF F408h	Write-Only	N/A	
SYSCFG	Byte	FF F40Ch	Read/Write	00h	
CVSD/PCM Converter Module 0					
CVSDIN0	Word	FF 4800h	Write-Only	0000h	
CVSDOUT0	Word	FF 4804h	Read-Only	0000h	
PCMIN0	Word	FF 4808h	Write-Only	0000h	
PCMOUT0	Word	FF 480Ch	Read-Only	0000h	
LOGIN0	Byte	FF 4810h	Write-Only	00h	
LOGOUT0	Byte	FF 4814h	Read-Only	00h	
LINEARIN0	Word	FF 4818h	Write-Only	0000h	
LINEAROUT0	Word	FF 481Ch	Read-Only	0000h	
CVCTRL0	Word	FF 4820h	Read/Write	0000h	
CVSTAT0	Word	FF 4824h	Read-Only	0000h	
CVSD/PCM Converter Module 1					
CVSDIN1	Word	FF 4C00h	Write-Only	0000h	
CVSDOUT1	Word	FF 4C04h	Read-Only	0000h	
PCMIN1	Word	FF 4C08h	Write-Only	0000h	
PCMOUT1	Word	FF 4C0Ch	Read-Only	0000h	
LOGIN1	Byte	FF 4C10h	Write-Only	00h	
LOGOUT1	Byte	FF 4C14h	Read-Only	00h	
LINEARIN1	Word	FF 4C18h	Write-Only	0000h	
LINEAROUT1	Word	FF 4C1Ch	Read-Only	0000h	
CVCTRL1	Word	FF 4C20h	Read/Write	0000h	
CVSTAT1	Word	FF 4C24h	Read-Only	0000h	
Clock Generation and Power Management					
PMMCKCTL	Byte	FF A400h	Read/Write	0001 100Xb	
PMMSTCTL	Byte	FF A404h	Read/Write	00h	
PMMSR	Byte	FF A408h	Read/Write	0000 00XXb	
PMMPRSHC	Word	FF A40Ch	Read/Write	001Fh	
PMMPRSPC	Byte	FF A410h	Read/Write	00h	
PMMPRSSC	Word	FF A414h	Read/Write	02DBh	
PMMPLL1CTL1	Word	FF A420h	Read/Write	0000h	
PMMPLL1CTL2	Word	FF A424h	Read/Write	0000h	
PMMPLL1MDIV	Byte	FF A428h	Read/Write	00h	
PMMPLL1NDIV	Byte	FF A42Ch	Read/Write	00h	
PMMPLL1PDIV	Byte	FF A430h	Read/Write	00h	
PMMPLL1NMOD	Word	FF A434h	Read/Write	0000h	
PMMPLL1STUP	Word	FF A438h	Read/Write	007Fh	
PMMPLL2CTL1	Word	FF A440h	Read/Write	0000h	
PMMPLL2CTL2	Word	FF A444h	Read/Write	0000h	
PMMPLL2MDIV	Byte	FF A448h	Read/Write	00h	
PMMPLL2NDIV	Byte	FF A44Ch	Read/Write	00h	
PMMPLL2PDIV	Byte	FF A450h	Read/Write	00h	
PMMPLL2NMOD	Word	FF A454h	Read/Write	0000h	
PMMPLL2STUP	Word	FF A458h	Read/Write	007Fh	
PMMAUX1CTL	Byte	FF A500h	Read/Write	06h	
PMMAUX1PRSC	Word	FF A504h	Read/Write	00FFh	
PMMAUX2CTL	Byte	FF A510h	Read/Write	06h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
PMMAUX2PRSC	Word	FF A514h	Read/Write	00FFh	
PMMAUX3CTL	Byte	FF A520h	Read/Write	06h	
PMMAUX3PRSC	Word	FF A524h	Read/Write	00FFh	
PMMAUX4CTL	Byte	FF A530h	Read/Write	06h	
PMMAUX4PRSC	Word	FF A534h	Read/Write	00FFh	
PMMAUX5CTL	Byte	FF A540h	Read/Write	06h	
PMMAUX5PRSC	Word	FF A544h	Read/Write	00FFh	
PMMAUX6CTL	Byte	FF A550h	Read/Write	06h	
PMMAUX6PRSC	Word	FF A554h	Read/Write	00FFh	
PMMAUX7CTL	Byte	FF A560h	Read/Write	07h	
PMMAUX7PRSC	Word	FF A564h	Read/Write	0001h	
PMMAUX8CTL	Byte	FF A570h	Read/Write	06h	
PMMAUX8PRSC	Word	FF A574h	Read/Write	00FFh	
Multi-Input Wake-Up					
WKRPND1	Dword	FF C020h	Read/Set	0000 0000h	
WKRPND2	Dword	FF C024h	Read/Set	0000 0000h	
WKFPND1	Dword	FF C030h	Read/Set	0000 0000h	
WKFPND2	Dword	FF C034h	Read/Set	0000 0000h	
WKCLR1	Dword	FF C040h	Write-Only	N/A	
WKCLR2	Dword	FF C044h	Write-Only	N/A	
WKREN1	Dword	FF C000h	Read/Write	0000 0000h	
WKREN2	Dword	FF C004h	Read/Write	0000 0000h	
WKFEN1	Dword	FF C010h	Read/Write	0000 0000h	
WKFEN2	Dword	FF C014h	Read/Write	0000 0000h	
WKRIEN1	Dword	FF C050h	Read/Write	0000 0000h	
WKRIEN2	Dword	FF C054h	Read/Write	0000 0000h	
WKFIEN1	Dword	FF C060h	Read/Write	0000 0000h	
WKFIEN2	Dword	FF C064h	Read/Write	0000 0000h	
WKICTL1	Dword	FF C070h	Read/Write	0000 0000h	
WKICTL2	Dword	FF C074h	Read/Write	0000 0000h	
WKICTL3	Dword	FF C078h	Read/Write	0000 0000h	
WKICTL4	Dword	FF C07Ch	Read/Write	0000 0000h	
WKISTAT	Dword	FF C090h	Read/Write	0000 0000h	
General-Purpose I/O Ports					
PEDIR	Word	FF C404h	Read/Write	0000h	
PEDIN	Word	FF C408h	Read-Only	XXXXh	
PEDOUT	Word	FF C40Ch	Read/Write	XXXXh	
PEIEN	Word	FF C41Ch	Read/Write	0000h	
PEALT	Word	FF C400h	Read/Write	0000h	
PEALTS	Word	FF C418h	Read/Write	0000h	
PEWPU	Word	FF C410h	Read/Write	0000h	
PEPDR	Word	FF C420h	Read/Write	FFFFh	
PFDIR	Word	FF C804h	Read/Write	0000h	
PFDIN	Word	FF C808h	Read-Only	XXXXh	
PFDOUT	Word	FF C80Ch	Read/Write	XXXXh	
PFIEN	Word	FF C81Ch	Read/Write	0000h	
PFALT	Word	FF C800h	Read/Write	0000h	
PFALTS	Word	FF C818h	Read/Write	0000h	
PFWPU	Word	FF C810h	Read/Write	0000h	
PFPDR	Word	FF C820h	Read/Write	FFFFh	
PGDIR	Word	FF 6404h	Read/Write	0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
PGDIN	Word	FF 6408h	Read-Only	XXXXh	
PGDOUT	Word	FF 640Ch	Read/Write	XXXXh	
PGIEN	Word	FF 641Ch	Read/Write	0000h	
PGALT	Word	FF 6400h	Read/Write	0000h	
PGALTS	Word	FF 6418h	Read/Write	0000h	
PGWPU	Word	FF 6410h	Read/Write	0000h	
PGPDR	Word	FF 6420h	Read/Write	FFFFh	
PHDIR	Word	FF CC04h	Read/Write	0000h	
PHDIN	Word	FF CC08h	Read-Only	XXXXh	
PHDOUT	Word	FF CC0Ch	Read/Write	XXXXh	
PHIEN	Word	FF CC1Ch	Read/Write	0000h	
PHALT	Word	FF CC00h	Read/Write	0000h	
PHALTS	Word	FF CC18h	Read/Write	0000h	
PHWPU	Word	FF CC10h	Read/Write	0000h	
PHPDR	Word	FF CC20h	Read/Write	FFFFh	
Advanced Audio Interface					
ARFR	Word	FF 5000h	Read-Only	0000h	
ARDR0	Word	FF 5004h	Read-Only	0000h	
ARDR1	Word	FF 5008h	Read-Only	0000h	
ARDR2	Word	FF 500Ch	Read-Only	0000h	
ARDR3	Word	FF 5010h	Read-Only	0000h	
ATFR	Word	FF 5014h	Write-Only	XXXXh	
ATDR0	Word	FF 5018h	Write-Only	0000h	
ATDR1	Word	FF 501Ch	Write-Only	0000h	
ATDR2	Word	FF 5020h	Write-Only	0000h	
ATDR3	Word	FF 5024h	Write-Only	0000h	
AGCR	Word	FF 5028h	Read/Write	0000h	
AISCR	Word	FF 502Ch	Read/Write	0000h	
ARSCR	Word	FF 5030h	Read/Write	0000h	Becomes 0004h after enabling clock.
ATSCR	Word	FF 5034h	Read/Write	F000h	Becomes F003h after enabling clock.
ACCR	Word	FF 5038h	Read/Write	0000h	
ADMACR	Word	FF 503Ch	Read/Write	0000h	
Interrupt Control Unit					
IVECT	Dword	FF FE00h	Read-Only	0000 0010h	Fixed Addr.
NMISTAT	Dword	FF FE04h	Read-Only	0000 0000h	
EXNMI	Dword	FF FE08h	Read/Write	0000 0000h	
ISTR0	Dword	FF FE10h	Read-Only	0000 0000h	
ISTR1	Dword	FF FE14h	Read-Only	0000 0000h	
ISTR2	Dword	FF FE18h	Read-Only	0000 0000h	
IENR0	Dword	FF FE20h	Read/Write	0000 0000h	
IENR1	Dword	FF FE24h	Read/Write	0000 0000h	
IENR2	Dword	FF FE28h	Read/Write	0000 0000h	
SOFTR0	Dword	FF FE40h	Read/Write	0000 0000h	
SOFTR1	Dword	FF FE44h	Read/Write	0000 0000h	
SOFTR2	Dword	FF FE48h	Read/Write	0000 0000h	
INTGPAR0	Dword	FF FE50h	Read/Write	0000 0000h	
INTGPBR0	Dword	FF FE54h	Read/Write	0000 0000h	
INTGPAR1	Dword	FF FE58h	Read/Write	0000 0000h	
INTGPBR1	Dword	FF FE5Ch	Read/Write	0000 0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
INTGPAR2	Dword	FF FE60h	Read/Write	0000 0000h	
INTGPBR2	Dword	FF FE64h	Read/Write	0000 0000h	
IDBG	Dword	FF FEFCh	Read-Only	0000 0000h	
UART0					
U0TBUF	Byte	FF 9400h	Read/Write	XXh	
U0RBUF	Byte	FF 9404h	Read-Only	XXh	
U0ICTRL	Byte	FF 9408h	Read/Write	01h	Bits 0:1 read only
U0STAT	Byte	FF 940Ch	Read-Only	00h	
U0FRS	Byte	FF 9410h	Read/Write	00h	
U0MDSL1	Byte	FF 9414h	Read/Write	00h	
U0BAUD	Byte	FF 9418h	Read/Write	00h	
U0PSR	Byte	FF 941Ch	Read/Write	00h	
U0OVR	Byte	FF 9420h	Read/Write	00h	
U0MDSL2	Byte	FF 9424h	Read/Write	00h	
U0SPOS	Byte	FF 9428h	Read/Write	06h	
UART1					
U1TBUF	Byte	FF 9800h	Read/Write	XXh	
U1RBUF	Byte	FF 9804h	Read-Only	XXh	
U1ICTRL	Byte	FF 9808h	Read/Write	01h	Bits 0:1 read only
U1STAT	Byte	FF 980Ch	Read-Only	00h	
U1FRS	Byte	FF 9810h	Read/Write	00h	
U1MDSL1	Byte	FF 9814h	Read/Write	00h	
U1BAUD	Byte	FF 9818h	Read/Write	00h	
U1PSR	Byte	FF 981Ch	Read/Write	00h	
U1OVR	Byte	FF 9820h	Read/Write	00h	
U1MDSL2	Byte	FF 9824h	Read/Write	00h	
U1SPOS	Byte	FF 9828h	Read/Write	06h	
UART2					
U2TBUF	Byte	FF 9C00h	Read/Write	XXh	
U2RBUF	Byte	FF 9C04h	Read-Only	XXh	
U2ICTRL	Byte	FF 9C08h	Read/Write	01h	Bits 0:1 read only
U2STAT	Byte	FF 9C0Ch	Read-Only	00h	
U2FRS	Byte	FF 9C10h	Read/Write	00h	
U2MDSL1	Byte	FF 9C14h	Read/Write	00h	
U2BAUD	Byte	FF 9C18h	Read/Write	00h	
U2PSR	Byte	FF 9C1Ch	Read/Write	00h	
U2OVR	Byte	FF 9C20h	Read/Write	00h	
U2MDSL2	Byte	FF 9C24h	Read/Write	00h	
U2SPOS	Byte	FF 9C28h	Read/Write	06h	
UART3					
U3TBUF	Byte	FF 5C00h	Read/Write	XXh	
U3RBUF	Byte	FF 5C04h	Read-Only	XXh	
U3ICTRL	Byte	FF 5C08h	Read/Write	01h	Bits 0:1 read only
U3STAT	Byte	FF 5C0Ch	Read-Only	00h	
U3FRS	Byte	FF 5C10h	Read/Write	00h	
U3MDSL1	Byte	FF 5C14h	Read/Write	00h	
U3BAUD	Byte	FF 5C18h	Read/Write	00h	
U3PSR	Byte	FF 5C1Ch	Read/Write	00h	
U3OVR	Byte	FF 5C20h	Read/Write	00h	
U3MDSL2	Byte	FF 5C24h	Read/Write	00h	
U3SPOS	Byte	FF 5C28h	Read/Write	06h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
Microwire/SPI Interfaces					
MW0DAT	Word	FF 8400h	Read/Write	XXXXh	
MW0CTL1	Word	FF 8404h	Read/Write	0000h	
MW0STAT	Word	FF 8408h	Read-Only	All implemented bits are 0	
MW1DAT	Word	FF 5800h	Read/Write	xxxxh	
MW1CTL1	Word	FF 5804h	Read/Write	0000h	
MW1STAT	Word	FF 5808h	Read-Only	All implemented bits are 0	
ACCESS.bus Module 0					
ACB0SDA	Byte	FF 8000h	Read/Write	XXh	
ACB0ST	Byte	FF 8004h	Read/Write	00h	
ACB0CST	Byte	FF 8008h	Read/Write	00h	
ACB0CTL1	Byte	FF 800Ch	Read/Write	00h	
ACB0ADDR	Byte	FF 8010h	Read/Write	XXh	
ACB0CTL2	Byte	FF 8014h	Read/Write	00h	
ACB0ADDR2	Byte	FF 8018h	Read/Write	XXh	
ACB0CTL3	Byte	FF 801Ch	Read/Write	00h	
ACCESS.bus Module 1					
ACB1SDA	Byte	FF 5400h	Read/Write	XXh	
ACB1ST	Byte	FF 5404h	Read/Write	00h	
ACB1CST	Byte	FF 5408h	Read/Write	00h	
ACB1CTL1	Byte	FF 540Ch	Read/Write	00h	
ACB1ADDR	Byte	FF 5410h	Read/Write	XXh	
ACB1CTL2	Byte	FF 5414h	Read/Write	00h	
ACB1ADDR2	Byte	FF 5418h	Read/Write	XXh	
ACB1CTL3	Byte	FF 541Ch	Read/Write	00h	
Timing and Watchdog Module					
TWCFG	Byte	FF A000h	Read/Write	00h	
TWCP	Byte	FF A004h	Read/Write	00h	
TWMT0	Word	FF A008h	Read/Write	FFFFh	
T0CSR	Byte	FF A00Ch	Read/Write	00h	
WDCNT	Byte	FF A010h	Write-Only	0Fh	
WDSDM	Byte	FF A014h	Write-Only	5Fh	
Multi-Function Timer Module 0					
TCNT1_0	Word	FF 9000h	Read/Write	XXXXh	
TCRA_0	Word	FF 9004h	Read/Write	XXXXh	
TCRB_0	Word	FF 9008h	Read/Write	XXXXh	
TCNT2_0	Word	FF 900Ch	Read/Write	XXXXh	
TPRSC_0	Byte	FF 9010h	Read/Write	00h	
TCKC_0	Byte	FF 9014h	Read/Write	00h	
TMCTRL_0	Byte	FF 9018h	Read/Write	00h	
TICTL_0	Byte	FF 901Ch	Read/Write	00h	
TICLR_0	Byte	FF 9020h	Write-Only	FFh	
Multi-Function Timer Module 1					
TCNT1_1	Word	FF 6000h	Read/Write	XXXXh	
TCRA_1	Word	FF 6004h	Read/Write	XXXXh	
TCRB_1	Word	FF 6008h	Read/Write	XXXXh	
TCNT2_1	Word	FF 600Ch	Read/Write	XXXXh	
TPRSC_1	Byte	FF 6010h	Read/Write	00h	
TCKC_1	Byte	FF 6014h	Read/Write	00h	
TMCTRL_1	Byte	FF 6018h	Read/Write	00h	
TICTL_1	Byte	FF 601Ch	Read/Write	00h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
TICLR_1	Byte	FF 6020h	Write-Only	FFh	
Versatile Timer Unit Module 0					
MODE_0	Word	FF 8800h	Read/Write	0000h	
IO1CTL_0	Word	FF 8804h	Read/Write	0000h	
IO2CTL_0	Word	FF 8808h	Read/Write	0000h	
INTCTL_0	Word	FF 880Ch	Read/Write	0000h	
INTPND_0	Word	FF 8810h	Read/Write	0000h	
CLK1PS_0	Word	FF 8814h	Read/Write	0000h	
COUNT1_0	Word	FF 8818h	Read/Write	0000h	
PERCAP1_0	Word	FF 881Ch	Read/Write	0000h	
DTYCAP1_0	Word	FF 8820h	Read/Write	0000h	
COUNT2_0	Word	FF 8824h	Read/Write	0000h	
PERCAP2_0	Word	FF 8828h	Read/Write	0000h	
DTYCAP2_0	Word	FF 882Ch	Read/Write	0000h	
CLK2PS_0	Word	FF 8830h	Read/Write	0000h	
COUNT3_0	Word	FF 8834h	Read/Write	0000h	
PERCAP3_0	Word	FF 8838h	Read/Write	0000h	
DTYCAP3_0	Word	FF 883Ch	Read/Write	0000h	
COUNT4_0	Word	FF 8840h	Read/Write	0000h	
PERCAP4_0	Word	FF 8844h	Read/Write	0000h	
DTYCAP4_0	Word	FF 8848h	Read/Write	0000h	
Versatile Timer Unit Module 1					
MODE_1	Word	FF 8C00h	Read/Write	0000h	
IO1CTL_1	Word	FF 8C04h	Read/Write	0000h	
IO2CTL_1	Word	FF 8C08h	Read/Write	0000h	
INTCTL_1	Word	FF 8C0Ch	Read/Write	0000h	
INTPND_1	Word	FF 8C10h	Read/Write	0000h	
CLK1PS_1	Word	FF 8C14h	Read/Write	0000h	
COUNT1_1	Word	FF 8C18h	Read/Write	0000h	
PERCAP1_1	Word	FF 8C1Ch	Read/Write	0000h	
DTYCAP1_1	Word	FF 8C20h	Read/Write	0000h	
COUNT2_1	Word	FF 8C24h	Read/Write	0000h	
PERCAP2_1	Word	FF 8C28h	Read/Write	0000h	
DTYCAP2_1	Word	FF 8C2Ch	Read/Write	0000h	
CLK2PS_1	Word	FF 8C30h	Read/Write	0000h	
COUNT3_1	Word	FF 8C34h	Read/Write	0000h	
PERCAP3_1	Word	FF 8C38h	Read/Write	0000h	
DTYCAP3_1	Word	FF 8C3Ch	Read/Write	0000h	
COUNT4_1	Word	FF 8C40h	Read/Write	0000h	
PERCAP4_1	Word	FF 8C44h	Read/Write	0000h	
DTYCAP4_1	Word	FF 8C48h	Read/Write	0000h	
SDC					
ADCGCR	Word	FF AC00h	Read/Write	0000h	
ADCACR	Word	FF AC04h	Read/Write	0000h	
ADCCNTRL	Word	FF AC08h	Read/Write	0000h	
ADCSTART	Word	FF AC0Ch	Write-Only	N/A	
ADCSCDLY	Word	FF AC10h	Read/Write	0000h	
ADCRESLT	Word	FF AC14h	Read-Only	0000h	
I^2S Interface					
I2SCLK	Dword	FF 4000h	Read/Write	0007 0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
I2SRXCTL	Dword	FF 4004h	Read/Write	0000 001Ch	
I2STXCTL	Dword	FF 4008h	Read/Write	0000 001Ch	
I2SSTAT	Dword	FF 401Ch	Read/Write	0000 0808h	
I2STXDATALEFT	Dword	FF 400Ch	Write-Only	0000 0000h	
I2STXDATARIGHT	Dword	FF 4010h	Write-Only	0000 0000h	
I2SRXDATALEFT	Dword	FF 4014h	Read-Only	0000 0000h	
I2SRXDATARIGHT	Dword	FF 4018h	Read-Only	0000 0000h	
Audio Subsystem Controller					
ADMAC	Word	FF 6820h	Read/Write	0000h	
ADMAS	Word	FF 681Ch	Read-Only	0000h	
ASCINTSEL	Word	FF 6818h	Read/Write	0000h	
ASCDMASEL0	Word	FF 6800h	Read/Write	0000h	
ASCDMASEL1	Word	FF 6804h	Read/Write	0000h	
ASCDDMASEL0	Word	FF 6808h	Read/Write	0000h	
ASCDDMASEL1	Word	FF 680Ch	Read/Write	0000h	
ASCDDMASEL2	Word	FF 6810h	Read/Write	0000h	
ASCDDMASEL3	Word	FF 6814h	Read/Write	0000h	
DSP Interface					
PDATA	Word	FF EC00h	Read/Write	0000h	
PADR	Word	FF EC04h	Write-Only	0000h	
PCFG	Word	FF EC08h	Read/Write	0000h	
PSTS	Word	FF EC0Ch	Read-Only	0000h	
PSEM	Word	FF EC10h	Read/Write	0000h	
PMASK	Word	FF EC14h	Read/Write	0000h	
PCLEAR	Word	FF EC18h	Write-Only	0000h	
APBP_SEM	Word	FF EC1Ch	Read-Only	0000h	
APBP_COM0	Word	FF EC20h	Read/Write	0000h	
APBP_REP0	Word	FF EC24h	Read-Only	0000h	
APBP_COM1	Word	FF EC28h	Read/Write	0000h	
APBP_REP1	Word	FF EC2Ch	Read-Only	0000h	
APBP_COM2	Word	FF EC30h	Read/Write	0000h	
APBP_REP2	Word	FF EC34h	Read-Only	0000h	
Telematics Codec					
TCDCBASIC	Word	FF 4400h	Read/Write	0000h	
TCDCDACSTATUS	Word	FF 4404h	Read-Only	1010h	
TCDCADCSTATUS	Word	FF 4408h	Read-Only	1010h	
TCDCDSP	Word	FF 440Ch	Read/Write	0000h	
TCDCADCANA1	Word	FF 4410h	Read/Write	0000h	
TCDCADCANA2	Word	FF 4414h	Read/Write	0000h	
TCDCADC1CLK	Word	FF 4418h	Read/Write	0000h	
TCDCADC2CLK	Word	FF 441Ch	Read/Write	0000h	
TCDCDACCLK	Word	FF 4420h	Read/Write	0000h	
TCDCFIFO	Word	FF 4424h	Read/Write	0000h	
TCDCIRQEN	Word	FF 4428h	Read/Write	0000h	
TCDCIRQPNDCLR	Word	FF 442Ch	Read/Write	0000h	
TCDCCOMPC0	Word	FF 4430h	Read/Write	0000h	
TCDCCOMPC1	Word	FF 4434h	Read/Write	0000h	
TCDCCOMPC2	Word	FF 4438h	Read/Write	0000h	
TCDCDEBUG	Word	FF 443Ch	Read/Write	0000h	
TCDCADC1	Word	FF 4448h	Read-Only	0000h	
TCDCADC2	Word	FF 444Ch	Read-Only	0000h	

Table 34-1. Detailed Device Mapping (continued)

Register Name	Size	Address	Access Type	Value After Reset	Comments
TCDCLEFT	Word	FF 4450h	Read/Write	0000h	
TCDCRIGHT	Word	FF 4454h	Read/Write	0000h	
TCDCMONITOR	Word	FF 4458h	Read/Write	0000h	
Real Time Clock					
RTCCST	Byte	FF A800h	Read/Write	00h	
RTUDST	Byte	FF A804h	Read-Only	00h	
RTCEIST	Byte	FF A8008h	Read/Write	00h	
RTCIEN	Byte	FF A80Ch	Read/Write	00h	
RTPRD	Word	FF A810h	Read-Only	0000h	
RTCRD	Dword	FF A814h	Read-Only	0000 0001h	
RTCLD	Dword	FF A818h	Read/Write	0000 0000h	
RTCCMP1	Word	FF A81Ch	Read/Write	7FFFh	
RTCCMP2	Dword	FF A820h	Read/Write	FFFF FFFFh	
RTCCMP3	Dword	FF A824h	Read/Write	FFFF FFFFh	

35 Register Bit Fields

The following tables show the functions of the bit fields of the device registers. For more information on using these registers, see the detailed description of the applicable function elsewhere in this data sheet.

Bluetooth LLC Registers	7	6	5	4	3	2	1	0
PLN	Reserved					PLN2:0		
WHITENING_ CHANNEL_ SELECTION	Reserved					CHANNEL_ SELECTION1:0		WHITENING
SINGLE_FREQUENCY _SELECTION	Reserved	SINGLE_FREQUENCY_SEL6:0						
CORRELATOR_CLOCK _SEL	Reserved	CDEL			Reserved			CCS
LN_BT_CLOCK_0	LN_BT_CLOCK7:0							
LN_BT_CLOCK_1	LN_BT_CLOCK15:8							
LN_BT_CLOCK_2	LN_BT_CLOCK23:16							
LN_BT_CLOCK_3	Reserved				LN_BT_CLOCK27:23			
RX_CN	Reserved	RX_CN6:0						
TX_CN	Reserved	TX_CN6:0						
AC_ACCEPTLVL[7:0]	AC_ACCEPTLVL7:0							
AC_ACCEPTLVL[15:8]	Reserved						AC_ACCEPTLVL9:8	
LAP_ACCEPTLVL	Reserved		LAP_ACCEPTLVL5:0					
RFSYNCH_DELAY	Reserved		RFSYNCH_DELAY5:0					
SPI_READ[7:0]	SPI_READ7:0							
SPI_READ[15:8]	SPI_READ15:8							
SPI_MODE_CONFIG	Reserved		SPI_CLK_CONF1:0		SPI_LEN_ CONF	SPI_DATA _CONF3	SPI_DATA _CONF2	SPI_DATA_ CONF1
M_COUNTER_0	M_COUNTER7:0							
M_COUNTER_1	M_COUNTER15:8							
M_COUNTER_2	Reserved		FORCE_ WAKEUP	M_COUNTER20:16				
N_COUNTER_0	N_COUNTER7:0							
N_COUNTER_1	Reserved						N_COUNTER9:8	
BT_CLOCK_WR_0	BT_CLOCK_WR7:1							Reserved
BT_CLOCK_WR_1	BT_CLOCK_WR15:8							
BT_CLOCK_WR_2	BT_CLOCK_WR23:16							
BT_CLOCK_WR_3	Reserved				BT_CLOCK_WR27:24			
WTPTC_1SLOT[7:0]	WTPTC_1SLOT7:0							
WTPTC_1SLOT[15:8]	WTPTC_1SLOT15:8							
WTPTC_3SLOT[7:0]	WTPTC_3SLOT7:0							
WTPTC_3SLOT[15:8]	WTPTC_3SLOT15:8							
WTPTC_5SLOT[7:0]	WTPTC_5SLOT7:0							

Bluetooth LLC Registers	7	6	5	4	3	2	1	0
WTPTC_5SLOT[15:8]	WTPTC_5SLOT15:8							
SEQ_RESET	Reserved							SEQ_RESET
SEQ_CONTINUE	Reserved							SEQ_ CONTINUE
RX_STATUS	Reserved	HEC Error	Header Error Correction	AM_ ADDR Error	Payload CRC Error	Payload Error Correction	Payload Length Error	PACKET_ DONE
CHIP_ID	Reserved				CHIP_ID			
INT_VECTOR	INT_VECTOR7:0							
SYSTEM_CLK_EN	Reserved				CLK_EN3	CLK_EN2	INT_SEQ_ EN	CLK_EN1
LINK_TIMER_WR_RD[7:0]	LINKTIMER_WR_RD7:0							
LINK_TIMER_WR_RD[15:8]	LINKTIMER_WR_RD15:8							
LINK_TIMER_SELECT	Reserved					LINKTIMER_SELECT		
LINK_TIMER_STATUS_ EXP_FLAG	LINK_TIMER_STATUS_EXP_FLAG7:0							
LINK_TIMER_STATUS_ RD_WR_FLAG	Reserved						LINK_ TIMER _WRITE_ DONE	LINK_ TIMER_ READ_ VALID
LINK_TIMER_ADJUST_PLUS	LINKTIMER_ADJUST_PLUS7:0							
LINK_TIMER_ADJUST_MINUS	LINKTIMER_ADJUST_MINUS7:0							
SLOTTIMER_WR_RD	Reserved		SLOT_TIMER_WR_RD5:0					
RX_CRC	RX_CRC							
AFH_PLN_SEL	Reserved					AFH_PLN_SEL		
AFH_SAME_CN_CONFIG	Reserved						HOP_ CALC_TX	SCDIS
AFH_CN_MAP_0	CHANNEL_NUMBER7:0							
AFH_CN_MAP_1	CHANNEL_NUMBER15:8							
AFH_CN_MAP_2	CHANNEL_NUMBER23:16							
AFH_CN_MAP_3	CHANNEL_NUMBER31:24							
AFH_CN_MAP_4	CHANNEL_NUMBER39:32							
AFH_CN_MAP_5	CHANNEL_NUMBER47:40							
AFH_CN_MAP_6	CHANNEL_NUMBER55:48							
AFH_CN_MAP_7	CHANNEL_NUMBER63:56							
AFH_CN_MAP_8	CHANNEL_NUMBER71:64							
AFH_CN_MAP_9	AFH_EN	CHANNEL_NUMBER78:72						

USB Controller	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FADDR	Reserved								0	Function Address						
POWER	Reserved								ISUP	SC	Reserved		RST	RESU	SUSP	ENSP
INTRTX	Reserved												EPTX3:1			EP0
INTRRX	Reserved												EPRX3:1			Res.
INTRTXE	Reserved												EPTX3:1			EP0
INTRRXE	Reserved												EPRX3:1			Res.
INTRUSB	Reserved							VBER	SESR	DISC	CON	SOF	RST	BABL	RESU	SUSP
INTRUSBE	Reserved							VBER	SESR	DISC	CON	SOF	RST	BABL	RESU	SUSP
FRAME	Reserved					FRAME10:0										
INDEX	Reserved												EP3:0			
DEVCTL	Reserved								BD	FSD	LSD	VBUS		HM	HRQ	SESS
CSR0	0							FF	SSE	SRP	SDS	SUE	DE	STS	TPR	RPR
TXMAXP	Reserved					MAX_PAYLOAD										
TXCSR	AS	ISO	MODE	DMAEN	FDT	DMAMD	0		ICTX	CDAT	STS	SDS	FF	UR	FNE	TPR
RXMAXP	Reserved					MAX_PAYLOAD										
RXCSR	AS	ISO	DMAEN	DN	DMAMD	0		ICTX	CDAT	STS	SDS	FF	DE	OR	FF	RPR
RXCNT	Reserved			EPRX_CNT11:0												
VCTRL	DWEN	ADDRESS							DATA							
VSTATUS	Reserved												SUSPOTG_CTRL	SUSP_CTRL	FORCE_SUSP	FORCE_SUSPCTRL

CAN Control/ Status	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CGCRn	Reserved				EIT	DIA GEN	INTE RNAL	LOOP BACK	IGN ACK	LO	DD IR	TST PEN	BUFF LOCK	CRX	CTX	CAN EN
CTIMn	PSC6:0							SJW1:0		TSEG1[3:0]				TSEG2[2:0]		
GMSKBn	GM28:18											RTR	IDE	GM17:15		
GMSKXn	GM14:0															XRTR
BMSKBn	BM28:18											RTR	IDE	BM17:15		
BMSKXn	BM14:0															XRTR
CIENn	EIEN	IEN14:0														
CIPNDn	EIPND	IPND14:0														
CICLRn	EICLR	ICLR14:0														
CICENn	EICEN	ICEN14:0														
CSTPNDn	Reserved								NS2:0			IRQ	IST3:0			
CANECn	REC7:0								TEC7:0							
CEDIAGn	Res.	DRIVE	MON	CRC	STUFF	TXE	EBID5:0						EFID3:0			
CTMRn	CTMR15:0															

CAN Memory Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CnMBn.ID1	XI28 ID10	XI27 ID9	XI26 ID8	XI25 ID7	XI24 ID6	XI23 ID5	XI22 ID4	XI21 ID3	XI20 ID2	XI19 ID1	XI18 ID0	SRR RTR	IDE	XI17	XI16	XI15
CnMBn.ID0	XI14	XI13	XI12	XI11	XI10	XI9	XI8	XI7	XI6	XI5	XI4	XI3	XI2	XI1	XI0	RTR
CnMBn. DATA0	Data 1.7	Data 1.6	Data 1.5	Data 1.4	Data 1.3	Data 1.2	Data 1.1	Data 1.0	Data 2.7	Data 2.6	Data 2.5	Data 2.4	Data 2.3	Data 2.2	Data 2.1	Data 2.0
CnMBn. DATA1	Data 3.7	Data 3.6	Data 3.5	Data 3.4	Data 3.3	Data 3.2	Data 3.1	Data 3.0	Data 4.7	Data 4.6	Data 4.5	Data 4.4	Data 4.3	Data 4.2	Data 4.1	Data 4.0
CnMBn. DATA2	Data 5.7	Data 5.6	Data 5.5	Data 5.4	Data 5.3	Data 5.2	Data 5.1	Data 5.0	Data 6.7	Data 6.6	Data 6.5	Data 6.4	Data 6.3	Data 6.2	Data 6.1	Data 6
CnMBn. DATA3	Data 7.7	Data 7.6	Data 7.5	Data 7.4	Data 7.3	Data 7.2	Data 7.1	Data 7	Data 8.7	Data 8.6	Data 8.5	Data 8.4	Data 8.3	Data 8.2	Data 8.1	Data 8.0
CnMBn. TSTP	TSTP 15	TSTP 14	TSTP 13	TSTP 12	TSTP 11	TSTP 10	TSTP 9	TSTP 8	TSTP 7	TSTP 6	TSTP 5	TSTP 4	TSTP 3	TSTP 2	TSTP 1	TSTP 0
CnMBn. CNTSTAT	DLC3	DLC2	DLC1	DLC0	Reserved				PRI3	PRI2	PRI1	PRI0	ST3	ST2	ST1	ST0

EBIU Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCSLRn (low)	Reserved															
SCSLRn (high)	EXT_BASE_ADDRESS31:15															
SMSKRn (low)	Reserved						REG_SEL		MEM_TYPE			MEM_SIZE				
SMSKRn (high)	Reserved															
SMTMGR_SETn (low)	T_WP						T_WR		T_AS			T_RC				
SMTMGR_SETn (high)	Reserved		SMRP		Reserved		PS		PM	T_PRC				T_BTA		
FLASH_TRPDR (low)	Reserved				T_RPD											
FLASH_TRPDR (high)	Reserved															
SMCTLR (low)	SM_DW_S2			SM_DW_S1			SM_DW_S0			Reserved						
SMCTLR (high)	Reserved															

DMAC Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAn (low)	ADCAn15:0															
ADCAn (high)	ADCAn31:16															
ADRAn (low)	ADRAn15:0															
ADRAn (high)	ADRAn31:16															
ADCBn (low)	ADCBn15:0															
ADCBn (high)	ADCBn31:16															
ADRBn (low)	ADRBn15:0															
ADRBn (high)	ADRBn31:16															
BLTCn (low)	BLTCn15:0															
BLTCn (high)	BLTCn31:16															
BLTRn (low)	BLTRn15:0															
BLTRn (high)	BLTRn31:16															
RQTRn (low)	RQTRn15:0															
RQTRn (high)	Reserved															
RQTCNTn (low)	RQTCNTn15:0															
RQTCNTn (high)	Reserved															
DMACNTn (low)	WMO DE	INCB		ADB	INCA		ADA	SWRQ	BPC	OT	DIR	TCS	ETO	EOVR	ETC	CHEN
DMACNTn (high)	Reserved				SRCRQ					TOEN	HPROT				PF	BBE
DMASTATn (low)	Reserved			TO	BLV				Reserved		BNE	ERR	VLD	CHAC	OVR	TC
DMASTATn (high)	Reserved															

Bus Arbiter Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTGP (low)	Reserved										MAS4		MAS3		MAS2	
MASTGP (high)	Reserved															
ARBALGO (low)	Reserved												GRD	GRC	GRB	GRA
ARBALGO (high)	Reserved															
DFTMAST (low)	Reserved												DFTMAST			
DFTMAST (high)	Reserved															
ARBCFGLK (low)	Reserved															LOCK
ARBCFGLK (high)	Reserved															

System Configuration Registers	7	6	5	4	3	2	1	0
MCFG	Reserved	FREEZE	Reserved		ENV1SEL	ENV0SEL	ENV1OE	ENV0OE
MSTAT	ISPRST	WDRST	Reserved				OENV1	OENV0
SWRESET	Key Values							
SYSCFG	XDPUDIS	Reserved			USBIDDIG-PUEN	USBHCLKDIS	BTHCLKDIS	RFCKEN

CVSD/PCM Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CVSDINn	CVSDIN															
CVSDOUTn	CVSDOUT															
PCMINn	PCMIN															
PCMOUTn	PCMOUT															
LOGINn	Reserved								LOGIN							
LOGOUTn	Reserved								LOGOUT							

CVSD/PCM Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LINEARINn	LINEARIN															
LINEAROUTn	LINEAROUT															
CVCTRLn	CVFR	Res.	RESOLUTION		PCMCONV	CVSDCONV		DMAPI	DMAPO	DMACI	DMACO	CVSDERRINT	CVSDINT	PCMINT	CLKEN	CVEN
CVSTATn	Reserved					CVOUTST			CVINST			CVF	CVE	PCMINT	CVNF	CVNE

Clock and PMM Registers	7	6	5	4	3	2	1	0
	15	14	13	12	11	10	9	8
PMMCKCTL	Reserved			FCLKSRC		SCLK	MAIN_DCOE	POR
PMMSTCTL	DSC	WACK	HCCM	DMC	WBPSM	HALT	IDLE	PSM
PMMSR	Reserved						MCLKSTB	SCLKSTB
PMMPRSHC (low)	HPHCLK7:0							
PMMPRSHC (high)	Reserved				HPHCLK11:8			
PMMPRSPC	Reserved						DIVPCLK	
PMMPRSSC (low)	HPSCLK7:0							
PMMPRSSC (high)	Reserved		HPSCLK13:8					
PMMPLLnCTL1 (low)	Reserved			PLLIPSEL		PLLDCOE	PLLEN	
PMMPLLnCTL1 (high)	Reserved							
PMMPLLnCTL2 (low)	Reserved					HCCPLL	DPLLC	PCLKSTB
PMMPLLnCTL2 (high)	Reserved							
PMMPLLnMDIV	MDIV							
PMMPLLnNDIV	NDIV							
PMMPLLnPDIV	PDIV							
PMMPLLnNMOD (low)	NMOD							
PMMPLLnNMOD (high)	NMOD_DITH		Reserved					
PMMPLLnSTUP (low)	CLKCNT7:0							
PMMPLLnSTUP (high)	Reserved						CLKCNT9:8	
PMMAUXnCTL	Reserved					AUXCLKSRC		AUXCLKEN
PMMAUXnPRSC (low)	HPACLK7:0							
PMMAUXnPRSC (high)	Reserved				HPACLK11:8			

MIWU Registers	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKREN1	WKREN31:0																															
WKREN2	WKREN63:32																															
WKFEN1	WKFEN31:0																															
WKFEN2	WKFEN63:32																															
WKRPND1	WKRPD31:0																															
WKRPND2	WKRPD63:32																															
WKFPND1	WKFPD31:0																															
WKFPND2	WKFPD63:32																															
WKCLR1	WKCL31:0																															
WKCLR2	WKCL63:32																															
WKRIEN1	WKRIEN31:0																															
WKRIEN2	WKRIEN63:32																															
WKFIEN1	WKFIEN31:0																															
WKFIEN2	WKFIEN63:32																															
WKICTL1	WKINTR15:0 (2-bit fields)																															
WKICTL2	WKINTR31:16																															

MIWU Registers	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKICTL3	WKINTR47:32																															
WKICTL4	WKINTR63:48																															
WKISTAT	Reserved																								WKISTAT7:0							

GPIO Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxDIR	Px Port Direction															
PxDIN	Px Port Input Data															
PxDOUT	Px Port Output Data															
PxALT	Px Pins Alternate Function Enable															
PxALTS	Px Pins Alternate Function Source Selection															
PxWPU	Px Port Weak Pullup/Pulldown Enable															
PxPDR	Px Port Weak Pullup/Pulldown Direction															
PxIEN	Px Port Interrupt Enable															

AAI Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARFR	ARFH								ARFL							
ARDR0	ARDH								ARDL							
ARDR1	ARDH								ARDL							
ARDR2	ARDH								ARDL							
ARDR3	ARDH								ARDL							
ATFR	ATFH								ATFL							
ATDR0	ATDH								ATDL							
ATDR1	ATDH								ATDL							
ATDR2	ATDH								ATDL							
ATDR3	ATDH								ATDL							
AGCR	CLKEN	AAIEN	IOM2	IFS	FSL		CTF	CRF	IEBC	FSS	IEFS	SCS		LPB	DWL	ASS
AISCR	Reserved				TXEIC	TXIC	RXEIC	RXIC	TXEIP	TXIP	RXEIP	RXIP	TXEIE	TXIE	RXEIE	RXIE
ARSCR	RXFWM				RXDSA				RXSA				RXO	RXE	RXF	
ATSCR	TXFWM				TXDSA				TXSA				TXU	TXF	TXE	TXAE
ACCR	BCPRS								FCPRS							CSS
ADMACR	Reserved			ACO		ACD			TMD				RMD			

ICU Registers	31 . . . 16	15 . . . 12	11 . . . 8	7	6	5	4	3	2	1	0
IVECT	Reserved			INTVECT							
NMISTAT	Reserved										EXT
EXNMI	Reserved								ENLCK	PIN	EN
ISTAT0	IST31:1										Reserved
ISTAT1	IST63:32										
ISTAT2	Reserved			IST70:64							
IENAM0	IENAM31:1										Reserved
IENAM1	IENAM63:32										
IENAM2	Reserved			IENAM70:64							
SOFTINT0	SOFTINT31:1										Reserved
SOFTINT1	SOFTINT63:32										
SOFTINT2	Reserved			SOFTINT70:64							
INTGPAR0	INTGPA31:1										Reserved
INTGPAR1	INTGPA63:32										
INTGPAR2	Reserved			INTGPA70:64							
INTGPBR0	INTGPB31:1										Reserved
INTGPBR1	INTGPB63:32										

ICU Registers	31 . . . 16	15 . . . 12	11 . . . 8	7	6	5	4	3	2	1	0
INTGPBR2	Reserved				INTGPB70:64						
IDBG	Reserved	IRQVECT			INTVECT						

UART Registers	7	6	5	4	3	2	1	0
UnTBUF	UnTBUF							
UnRBUF	URBUF							
UnICTRL	UEEI	UERI	UETI	UEFCI	UCTS	UDCTS	URBF	UTBE
UnSTAT	Reserved	UXMIP	URB9	UBKD	UERR	UDOE	UFE	UPE
UnFRS	Reserved	UPEN	UPSEL		UXB9	USTP	UCHAR	
UnMDSL1	URTS	UFCE	UERD	UETD	UCKS	UBRK	UATN	UMOD
UnBAUD	UDIV7:0							
UnPSR	UPSC					UDIV10:8		
UnOVR	Reserved				UOVSR			
UnMDSL2	Reserved							USMD
UnSPOS	Reserved				USAMP			

Microwire/SPI Registers	15 . . . 9	8	7	6	5	4	3	2	1	0
MWnDAT	MWDAT									
MWnCTL1	SCDV	SCIDL	SCM	EIW	EIR	EIO	ECHO	MOD	MNS	MWEN
MWnSTAT	Reserved							OVR	RBF	BSY

ACB Registers	7	6	5	4	3	2	1	0
ACBnSDA	DATA							
ACBnST	SLVSTP	SDAST	BER	NEGACK	STASTR	NMATCH	MASTER	XMIT
ACBnCST	ARPMATCH	MATCHAF	TGSCL	TSDA	GMATCH	MATCH	BB	BUSY
ACBnCTL1	STASTRE	NMINTE	GCMEN	ACK	DMAEN	INTEN	STOP	START
ACBnADDR	SAEN	ADDR						
ACBnCTL2	SCLFRQ6:0							ENABLE
ACBnADDR2	SAEN	ADDR						
ACBnCTL3	Reserved					ARPEN	SCLFRQ8:7	

ADC Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCGCR	MUXOUTEN	INTEN	NREF_CFG		PREF_CFG		Reserved			MUX_CFG				DIFF	ADCIN	CLKEN
ADCACR	CNVT	TRG	PRM	Reserved										CLKDIV		CLKSEL
ADCCNTRL	Reserved													AUTO	EXT	POL
ADCSTART	Write any value.															
ADCSCDLY	ADC_DIV		ADC_DELAY1										ADC_DELAY2			
ADCRESLT	ADC_DONE	ADC_OFLW	Reserved	SIGN	ADC_RESULT											

TWM Registers	15 . . . 8	7	6	5	4	3	2	1	0
TWCFG	Reserved			WDSDME	WDCT0I	LWDCNT	LTWMT0	LTWCP	LTWCFG
TWCP	Reserved						MDIV		
TWMT0	PRESET								
T0CSR	Reserved				FRZT0E	WDTLD	T0INTE	TC	RST
WDCNT	Reserved	PRESET							
WDSDM	Reserved	RSTDATA							

MFT16 Registers	15 . . . 8	7	6	5	4	3	2	1	0
TCNT1_n	TCNT1								
TCRA_n	TCRA								
TCRB_n	TCRB								
TCNT2_n	TCNT2								
TPRSC_n	Reserved				CLKPS				
TCKC_n	Reserved			C2CSEL			C1CSEL		
TMCTRL_n	Reserved	TEN	TAOUT	TBEN	TAEN	TBEDG	TAEDG	TMDSEL	
TICTL_n	Reserved	TDIEN	TCIEN	TBIEN	TAIEN	TDPND	TCPND	TBPND	TAPND
TICLR_n	Reserved					TDCLR	TCCLR	TBCLR	TACLR

VTU Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODE_n	TMOD4		T8RUN	T7RUN	TMOD3		T6RUN	T5RUN	TMOD2		T4RUN	T3RUN	TMOD1		T2RUN	T1RUN
IO1CTL_n	P4POL	C4EDG			P3POL	C3EDG			P2POL	C2EDG			P1POL	C1EDG		
IO2CTL_n	P7POL	C7EDG			P6POL	C6EDG			P5POL	C5EDG			P5POL	C5EDG		
INTCTL_n	I4DEN	I4CEN	I4BEN	I4AEN	I3DEN	I3CEN	I3BEN	I3AEN	I2DEN	I2CEN	I2BEN	I2AEN	I1DEN	I1CEN	I1BEN	I1AEN
INTPND_n	I4DPD	I4CPD	I4BPD	I4APD	I3DPD	I3CPD	I3BPD	I3APD	I2DPD	I2CPD	I2BPD	I2APD	I1DPD	I1CPD	I1BPD	I1APD
CLK1PS_n	C2PRSC								C1PRSC							
COUNT1_n	CNT1															
PERCAP1_n	PCAP1															
DTYCAP1_n	DCAP1															
COUNT2_n	CNT2															
PERCAP2_n	PCAP2															
DTYCAP2_n	DCAP2															
CLK2PS_n	C4PRSC								C3PRSC							
COUNT3_n	CNT3															
PERCAP3_n	PCAP3															
DTYCAP3_n	DCAP3															
COUNT4_n	CNT4															
PERCAP4_n	PCAP4															
DTYCAP4_n	DCAP4															

I^2S Registers	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
I2SCLK	Reserved											WSRES					CLKDIV								Reserved				M S	CLKSEL		CLKEN
I2SRXCTL	Reserved											RX ST	Res	RXFIFOTHRESH			RXALIGN		Res.		RX LD	RX RD	RX LI	RX RI	RX ER	RXRES						RXMOD
I2STXCTL	Reserved									LSB FILL	FLU SH	TX ST	Res.	TXFIFOTHRESH			TXA IGN		TX FIFO ENL	TX FIFO ENR	TX LD	TX RD	TX LI	TX RI	TX ER	TXRES						TXMOD
I2STXDATALEFT	TXDATALEFT																															
I2STXDATARIGHT	TXDATARIGHT																															
I2SRXDATALEFT	RXDATALEFT																															
I2SRXDATARIGHT	RXDATARIGHT																															
I2SSTAT	Res.	WSSTATUS 4:3		RXSTATUSL				WSSTATUS2:0				RXSTATUSR					RXER IRQ	RXL IRQ	RXR IRQ	TXSTATUSL					TXER IRQ	TXL IRQ	TXR IRQ	TXSTATUSR				

ASC Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASCDMASEL0	DCH15:0															
ASCDMASEL1	Reserved														DCH17:16	
ASCDDMASELn	Reserved			DDMA[2n+1]					Reserved			DDMA[2n]				
ASCINTSEL	Reserved							ICH8:0								
ADMAS	Reserved														RRSTN	PBUSY
ADMAC	Reserved															DSPRSTN

DSP Interface Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDATA	PDATA															
PADR	PADR															
PCFG	MEMSEL				RRIE2	RRIE1	RRIE0	WFEIE	WFFIE	RFNEIE	RFFIE	RS	DRS		AIM	DSPR
PSTS	RCOMIM2:0			RRI2	RRI1	RRI0	PSEMI	WFEI	WFFI	RFNEI	RFFI	R		PRST	WTIP	RTIP
PSEM	PSEM															
PMASK	PMASK															
PCLEAR	PCLEAR															
APBP_SEM	APBP_SEM															
APBP_COM0	APBP_COM0															
APBP_REP0	APBP_REP0															
APBP_COM1	APBP_COM1															
APBP_REP1	APBP_REP1															
APBP_COM2	APBP_COM2															
APBP_REP2	APBP_REP2															

Codec Registers	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCDCBASIC	DFS	AFS	FLSDAC	FLSADC	Reserved		SEN	MUTER	MUTEL	DACOSR		DACSTMODE			ADC2ON	ADC1ON
TCDCDACSTATUS	DAC STATUS	Reserved		RIGHTFIFO					Reserved			LEFTFIFO				
TCDCADCSTATUS	ADC1 STATUS	ADC2 STATUS	Res.	ADC2FIFO					Reserved			ADC1FIFO				
TCDCDSP	Res.	ST CLPR	ST CLPL	SIDETONEATTEN				CLKPH	DIGMICGAIN			CUST COMP	ADC2DIT OFF	ADC1DIT OFF	DACDIT OFF	DACDIT ON
TCDCADCANA1	Res.	HPF		MUTE	MICSEL	Reserved				MICMODE	Reserved		MICGAIN			
TCDCADCANA2	Res.	HPF		MUTE	MICSEL	Reserved				MICMODE	Reserved		MICGAIN			
TCDCADC1CLK	Reserved				CLKTIE	ADCCLKSRC			ADCCLKDIV							
TCDCADC2CLK	Reserved				CLKTIE	ADCCLKSRC			ADCCLKDIV							
TCDCDACCLK	Reserved		SEL6M	DACRNG		DACCLKSRC			DACCLKDIV							
TCDCFIFO	DMAR	DMAL	DMA ADC2	DMA ADC1	DACFIFOTRIG				ADC2FIFOTRIG				ADC1FIFOTRIG			
TCDCIRQEN	Reserved		STCLP	MICCLP	ZXDR	ZXDL	DAC DOWN	ADC2 DOWN	ADC1 DOWN	DACUP	ADC2UP	ADC1UP	RGT FIFO	LFT FIFO	ADC2FIFO	ADC1FIFO
TCDCIRQPNDCLR	Reserved		STCLP	MICCLP	ZXDR	ZXDL	DAC DOWN	ADC2DOWN	ADC1DOWN	DAC UP	ADC2UP	ADC1UP	RGT FIFO	LFT FIFO	ADC2FIFO	ADC1FIFO
TCDCCOMPC0	COMPC0															
TCDCCOMPC1	COMPC1															
TCDCCOMPC2	COMPC2															
TCDCDEBUG	Reserved									SFTRST	Reserved			GPIO		
TCDCADC1	ADCDATA															
TCDCADC2	ADCDATA															
TCDCLEFT	LEFTDATA															
TCDCRIGHT	RIGHTDATA															
TCDCMONITOR	Res.	SDE	SDLIMIT			MD	DCLIMIT									

RTC Registers	31 . . . 16	15 . . . 8	7	6	5	4	3	2	1	0
RTCCST	Reserved					RTSTRT	RTPRST	RTDIV		
RTUDST	Reserved					RTUCP3	RTUCP2	RTUCP1	RTURTC	RTUDIV
RTCEIST	Reserved							RTCEVT3	RTCEVT2	RTCEVT1
RTCIEN	Reserved							RTCIEN3	RTCIEN2	RTCIEN1
RTCPRD	Reserved	RTPCNT								
RTCRD	RTCCNT									
RTCLD	RTCCLD									
RTCCMP1	Reserved	RTCCMP1								
RTCCMP2	RTCCMP2									
RTCCMP3	RTCCMP3									

36 Electrical Specifications

36.1 Absolute Maximum Ratings(1)

If Military/Aerospace specified devices are required, contact the Texas Instruments Semiconductor Sales Office/Distributors for availability and specifications.

	MIN	MAX	UNIT
Supply voltage (IOVCC, RFVCC, TCDACVCC, UVCC)		3.6	V
Supply voltage (VCC, PLLVCC, ADVCC, TCADCVCC)		2.0	V
All input and output voltages with respect to GND	–2.0	Supply +0.2	V
ESD protection level (Human Body Model)		2	kV
Allowable sink/source current per signal pin		±10	mA
Total current into IOVCC pins		200	mA
Total current into VCC pins (source)		200	mA
Total current out of GND pins (sink)		200	mA
Latch-up immunity		±200	mA
Temperature under bias	–40	85	°C
Storage temperature range	–65	150	°C

(1) Absolute maximum ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications are not ensured when operating the device at absolute maximum ratings.

36.2 DC Electrical Characteristics: Temperature: –40°C ≤ T_A ≤ 85°C)

PARAMETER		CONDITIONS	MIN	MAX	UNIT
V_{CC}	Core Logic Supply Voltage		1.62	1.98	V
IOV_{CC}	I/O Supply Voltage		2.25/2.97	3.6	V
RFV_{CC}	RF I/O Supply Voltage		2.25	3.6	V
$PLLV_{CC}$	Analog PLL Supply Voltage		1.62	1.98	V
ADV_{CC}	ADC Supply Voltage		1.62	1.98	V
UV_{CC}	USB Supply Voltage		3	3.6	V
$TCADCV_{CC}$	Codec ADC Supply Voltage		1.62	1.98	V
$TCDACV_{CC}$	Codec DAC Voltage		2.97	3.6	V
V_{IL}	Logical 0 Input Voltage (except as noted below)		–0.2	0.2 IOV_{CC}	V
V_{IH}	Logical 1 Input Voltage (except as noted below)		0.7 IOV_{CC}	IOV_{CC} + 0.2	V
Vxl1	X1CKI Logical 0 Input Voltage(1)	External X1 clock	-0.2	0.2 $PLLV_{CC}$	V
Vxh1	X1CKI Logical 1 Input Voltage(1)	External X1 clock	0.7 $PLLV_{CC}$	PLLVcc + 0.2	V
Vxl2	X2CKI Logical 0 Input Voltage(1)	External X2 clock	-0.2	0.2 $PLLV_{CC}$	V
Vxh2	X2CKI Logical 1 Input Voltage(1)	External X2 clock	0.7 $PLLV_{CC}$	$PLLV_{CC}$ + 0.2	V
V_{hys}	Hysteresis Loop Width (1)		0.05 IOV_{CC}		V
I_{OL}	Logical 0 Output Current (except as noted below)	V_{OL} = 0.4 V, IOV_{CC} = 3 V	10		mA
I_{OH}	Logical 1 Output Current (except as noted below)	V_{OH} = 2.4 V, IOV_{CC} = 3 V	–10		mA
I_{OLACB}	SDAn, SCLn Logical 0 Output Current	V_{OL} = 0.4 V, IOV_{CC} = 2.25 V	3		mA
I_{OHW}	Weak Pullup Current(1)	V_{IL} = 0 V, IOV_{CC} = 3.6 V	–50	–200	µA
I_L	High Impedance Input Leakage Current(2)	0 V ≤ Vin ≤ IOV_{CC}	–2	2	µA
$I_{O(Off)}$	Output Leakage Current (I/O pins in input mode)	0 V ≤ Vout ≤ IOV_{CC}	–2	2	µA
I_{cca1}	Digital Supply Current Active Mode(3)	V_{CC} = 1.98 V, IOV_{CC} = 3.6 V		200	mA
I_{ccps}	Digital Supply Current Power Save Mode(4)	V_{CC} = 1.98 V, IOV_{CC} = 3.6 V		4	mA
I_{ccid}	Digital Supply Current Idle Mode(5)	V_{CC} = 1.98 V, IOV_{CC} = 3.6 V		2	mA

(1) Specified by design
(2) Only for digital inputs. Some analog inputs such as TCMIC1P and TCMIC2P have input protection devices which conduct if the input is taken high.
(3) Run from internal memory (RAM), Iout = 0 mA, X1CKI = 12 MHz, both PLLs enabled at 60 MHz
(4) Running from internal memory (RAM), Iout = 0 mA, XCKI1 = 12 MHz, PLLs disabled, X2CKI = 32.768 kHz
(5) Iout = 0 mA, XCKI1 = Vcc, X2CKI = 32.768 kHz

DC Electrical Characteristics: Temperature: –40°C ≤ T_A ≤ 85°C) *(continued)*

PARAMETER		CONDITIONS	MIN	MAX	UNIT
I_{ccq}	Digital Supply Current Halt Mode[5][6]	V_{CC} = 1.98 V, IOV_{CC} = 3.6 V, 20°C		400	µA

(6) Halt current approximately doubles for every 20°C.

36.3 USB Transceiver Electrical Characteristics

Temperature: –40°C ≤ T_A ≤ 85°C)

PARAMETER		CONDITIONS	MIN	MAX	UNIT
V_{IL}	USB Input Low Voltage			0.8	V
V_{IH}	USB Input High Voltage (driven)		2		V
V_{IHZ}	USB Input High Voltage (floating)[1]		2.7	3.6	V
V_{DI}	Differential Input Sensitivity[1]	(D+) – (D–)	–0.2	0.2	V
V_{CM}	Differential Common Mode Range[1]		0.8	2.5	V
V_{SE}	Single-Ended Receiver Threshold[1]		0.8	2	V
V_{OL}	Output Low Voltage	R_L = 1.5 kΩ to 3.6 V	0	0.3	V
V_{OH}	Output High Voltage	R_L = 15 kΩ to 0 V	2.8		V
V_{OSE0}	SE0 Voltage[1]			0.8	V
V_{OSE1}	SE1 Voltage[1]		0.8		V
V_{CRS}	Crossover Voltage[1]		1.3	2	V
I_{OZ}	TRI-STATE Data Line Leakage Current[1]	0 V < V_{IN} < 3.3 V	–10	10	µA
C_{TRN}	Transceiver Capacitance[1]			20	pF
R_{PUI}	Bus Pullup on Upstream Port (idle bus)[1]		0.9	1.575	kΩ
R_{PUA}	Bus Pullup on Upstream Port (port receiving)[1]		1.425	3.09	kΩ
R_{PD}	Bus Pulldown on Downstream Port[1]		14.25	24.8	kΩ
Z_{INP}	Input Impedance Exclusive of Pullup/Pulldown[1]		300		kΩ
V_{TERM}	Termination Voltage[1]		3	3.6	V

(1) Specified by design.

36.4 Telematics Codec Electrical Characteristics

Temperature: –40°C ≤ T_A ≤ 85°C)

PARAMETER(1)		CONDITIONS	MIN	TYP	MAX	UNIT
DUAL AUDIO ADC						
$VADC_{PB}$	Passband	Lower (HPF Mode 1), Fs = 8 kHz		300		Hz
		Upper, Fs = 8 kHz		3300		
$VADC_{RIP}$	Ripple	Same as above, relative to 1 kHz		±0.25		dB
$VADC_{SBA}$	Stopband Attenuation	Above 4.5 kHz		60		dB
		HPF Notch, 50/60 Hz, HPF = 1 and HPF = 2, worst case		55		dB
$VADC_{SNR}$	Signal-to-Noise Ratio	1 kHz, –3 dBFS relative to idle noise, Fs = 8 kHz, differential, A-weighted		85		dB
		Same as above, but single-ended		75		dB
$VADC_{DR}$	Dynamic Range	1 kHz, –60 dB input, AES-17, Fs = 8 kHz, differential, A-weighted		87		dB
$VADC_{LEVEL}$	Input Level	–3 dBFS, MICGAIN = 0, differential		0.13		V_{RMS}
				0.354		$V_{P-P(diff)}$
$VADC_{PSRR}$	Power Supply Rejection Ratio	Per application schematic, V_{RIPPLE} = 200 mVP-P, F_{RIPPLE} = 217 Hz, MICGAIN = 0		40		dB
$VADC_{THD+N}$	Total Harmonic Distortion, ADC + microphone + noise	–3 dB, 1 kHz input, relative to 0 dBFS, differential		0.015%		
$VADC_{CM}$	Common Mode Voltage, ADC + microphone			0.9		V
$VADC_{OFFSET}$	Offset, microphone preamp	Input referenced to MICGAIN = 1111		2.6		mV
$VADC_{RMICDIFF}$	Differential Input Resistance	TCMICxP to TCMICxN pins		150		kΩ
$VADC_{RMICSE}$	Single-Ended Input Resistance	TCMICxP/TCMICxN pin to AC ground (TCVBUFx pin)		75		kΩ
$VADC_{ROVRP}$	VREF Output Resistance			10		Ω
$VADC_{PSRRVRP}$	VREF PSRR	Per application schematic, V_{RIPPLE} = 200 mV_{P-P}, F_{RIPPLE} = 217 Hz		40		dB
$VADC_{GCR}$	Gain Control Range	Minimum		0		dB
		Maximum		30		dB
$VADC_{SS}$	Gain Control Step Size			2		dB
$VADC_{DELAY}$	ADC Group Delay	Mean from 300 Hz to 3.3 kHz		0.2		ms
STEREO DAC						
$SDAC_{PB}$	Passband Frequency	-3 dB point, FIR optimized(2)		22		kHz
$SDAC_{RIP}$	Ripple	DC to 20 kHz, Fs = 48 kHz, FIR optimized(2)		±0.08		dB
$SDAC_{SBA}$	Stop Band Attentuation	Above Nyquist frequency, FIR optimized(2)		70		dB
$SDAC_{SNR}$	Signal to Noise Ratio	1 kHz, 0 dBFS relative to idle noise, differential, A-weighted, DAC asynchronous to HCLK Clock, clocked from PLL		86		dB
$SDAC_{DR}$	Dynamic Range	1 kHz, –60 dBFS, AES-17, differential, A-weighted, DAC synchronous to HCLK Clock, clocked from 12 MHz Main Clock		95		dB
$SDAC_{THD+N}$	Total Harmonic Distortion + noise	–3 dB, 1 kHz input, relative to 0 dBFS, differential		0.01%		
$SDAC_{LEVEL}$	Line Output Level	1 kHz, 20 kΩ load		1		VRMS
$SDAC_{LOAD}$	Minimum Line Output Load	1 kHz, 1 V RMS		20		kΩ
$SDAC_{DLY}$	DAC Group Delay	Fs = 48 kHz		0.8		ms

(1) Specified by design
(2) DAC frequency response is tested using FIR compensation filter coefficients optimized for the CP3-DB-SP33 development board at a 48 kHz sampling frequency. At a 125× oversampling rate, these are C0 = 0121h, C1 = FB44h, C2 = 6C79h. At a 128× oversampling rate, these are C0 = 00E2h, C1 = FA7Dh, C2 = 6999h.

36.5 ADC Electrical Characteristics

Temperature: –40°C ≤ T_A ≤ 85°C)

PARAMETER		CONDITIONS	MIN	TYP	MAX	UNIT
V_{PREF}	ADC Positive Reference Input(1)		1.62		1.98	V
V_{NREF}	ADC Negative Reference Input (1)		0		0.25	V
	ADC Input Range (1)		V_{NREF}		V_{PREF}	V
	Clock Frequency			12		MHz
t_C	Conversion Time (10-bit result)(1)			14		µs
INL	Integral Non-Linearity				±2	LSB
DNL	Differential Non-Linearity				±0.7	LSB
C_{ADCIN}	Total Capacitance of ADC Input (1)		9		20	pF
C_{ADCINS}	Switched Capacitance of ADC Input (1)		8		10	pF
R_{ADCIN}	Resistance of ADC Input Path (1)		0.1		12	kΩ
C_{ADCIN}	Total Capacitance of ADC Reference Input (1)		50		100	pF
C_{ADCINS}	Switched Capacitance of ADC Reference Input (1)		8		10	pF
R_{ADCIN}	Resistance of ADC Reference Input Path(1)		0.2		0.6	kΩ

(1) Specified by design.

36.6 Output Signal Levels

The following output signals are powered by the digital IO supply (IOV_{CC}) or, in the case of the Bluetooth signals and port pins PF[6:0], the RF supply (RFV_{CC}).

summarizes the states of the output signals during the reset state (when V_{CC} power exists in the reset state) and during the Power Save mode.

This device has many dedicated input signals which are not internally pulled to a supply; these must be driven or tied off to a voltage lower than 0.5V or higher than IOV_{CC} – 0.5V, to assure that the current in Power Save mode does not exceed 1 mA. An input voltage between 0.5V and (VCC – 0.5V) may result in power consumption exceeding 1 mA.

Table 36-1. Output Pins During Reset and Power-Save Mode

SIGNALS ON A PIN	RESET STATE (with V_{CC})	POWER SAVE MODE	COMMENTS
PE[15:0]	TRI-STATE	Previous state	I/O ports will maintain their values when entering power-save mode. There is no logic function to support this, so if the CPU and GPIO peripheral have a clock, the device is capable of altering the state of its port pins.
PF[15:0]	TRI-STATE	Previous state	
PG[15:0]	TRI-STATE	Previous state	
PH[7:0]	TRI-STATE	Previous state	

36.7 Clock and Reset Timing

Table 36-2. Clock and Reset Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
CLOCK INPUT SIGNALS					
t_{X1p}	CLKIN period	Rising Edge (RE) to next RE, See Figure 36-1	83.33	83.33	ns
t_{X1h}	CLKIN high time, external clock	At 2 V level (both edges), See Figure 36-1	(0.5 Tclk) – 5		ns
t_{X1l}	CLKIN low time, external clock	At 0.8 V level (both edges), See Figure 36-1	(0.5 Tclk) – 5		ns
t_{X2p}	X2 period (2)	RE on X2 to next RE on X2, See Figure 36-1	10,000		ns
t_{X2h}	X2 high time, external clock	At 2V level (both edges), See Figure 36-1	(0.5 Tclk) – 500		ns
t_{X2l}	X2 low time, external clock	At 0.8V level (both edges), See Figure 36-1	(0.5 Tclk) – 500		ns
CLOCK OUTPUT SIGNALS					
t_{CLKp}	HCLK Clock period (3)	Rising Edge (RE) to next RE, See Figure 36-1	42,667	16.6	ns
t_{CLKh}	HCLK Clock high time	At 2 V level (both edges), See Figure 36-1	21,333	20.83	ns
t_{CLKl}	HCLK Clock low time	At 0.8 V level (both edges), See Figure 36-1	21,333	20.83	ns
t_{CLKr}	HCLK Clock rise time on RE of CLKIN	At 2 V level (both edges), See Figure 36-1		5	ns
t_{CLKf}	HCLK Clock fall time on FE of CLKIN	At 0.8 V level (both edges), See Figure 36-1		5	ns
RESET AND NMI INPUT SIGNALS					
t_{IW}	$\overline{\text{NMI}}$ Pulse Width, See (1)	$\overline{\text{NMI}}$ Falling Edge (FE) to RE	20		
t_{RST}	$\overline{\text{RESET}}$ Pulse Width	$\overline{\text{RESET}}$ FE to RE, See Figure 36-3	100		
V_{TRIP}	POR Rising Trigger Voltage	See Figure 36-4	1.11	1.54	V
t_{TRIP}	V_{CC} Rise Time to V_{TRIP}	See Figure 36-4		50	ms
t_D	V_{CC} Rise Time from V_{TRIP} to V_{CC}	See Figure 36-4		800	µs

(1) Specified by design.
(2) Only when operating with an external square wave on X2CKI; otherwise a 32 kHz crystal network must be used between X2CKI and X2CKO. If Slow Clock is internally generated from Main Clock, it may not exceed this limit.
(3) T_{clk} is the actual clock period of the CPU clock used in the system.
The value of T_{clk} is system dependent.
The maximum cycle time in Power Save mode is 42,667 ns (= 1/12 MHz × 2 × 256).
The maximum cycle time in Active mode is 1333 ns (= 1/12 MHz × 16).
The minimum cycle time in Active mode is 16.6 ns (= 1/60 MHz).

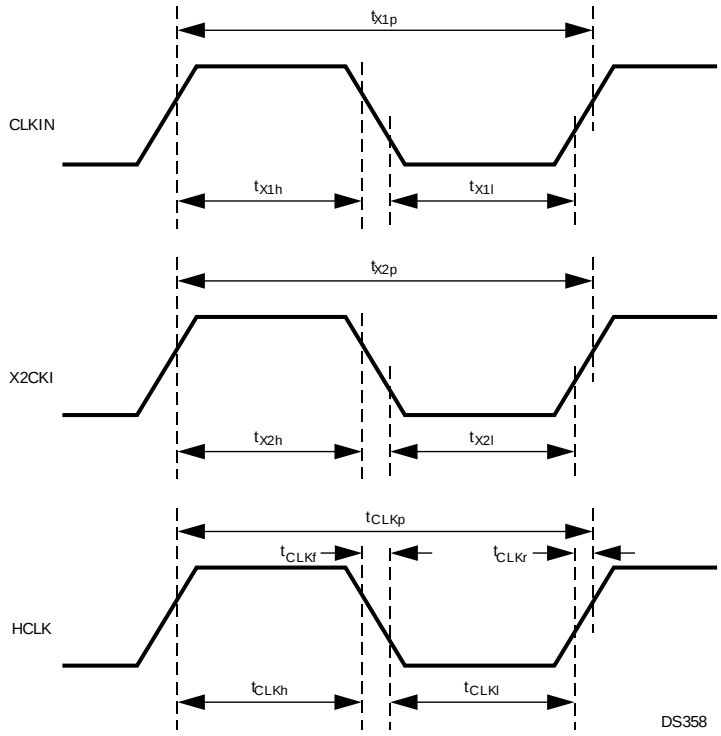


Figure 36-1. Clock Timing

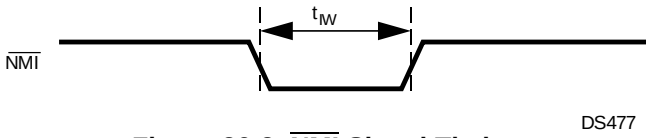


Figure 36-2. $\overline{\text{NMI}}$ Signal Timing

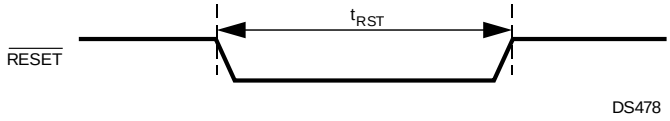


Figure 36-3. Non-Power-On Reset

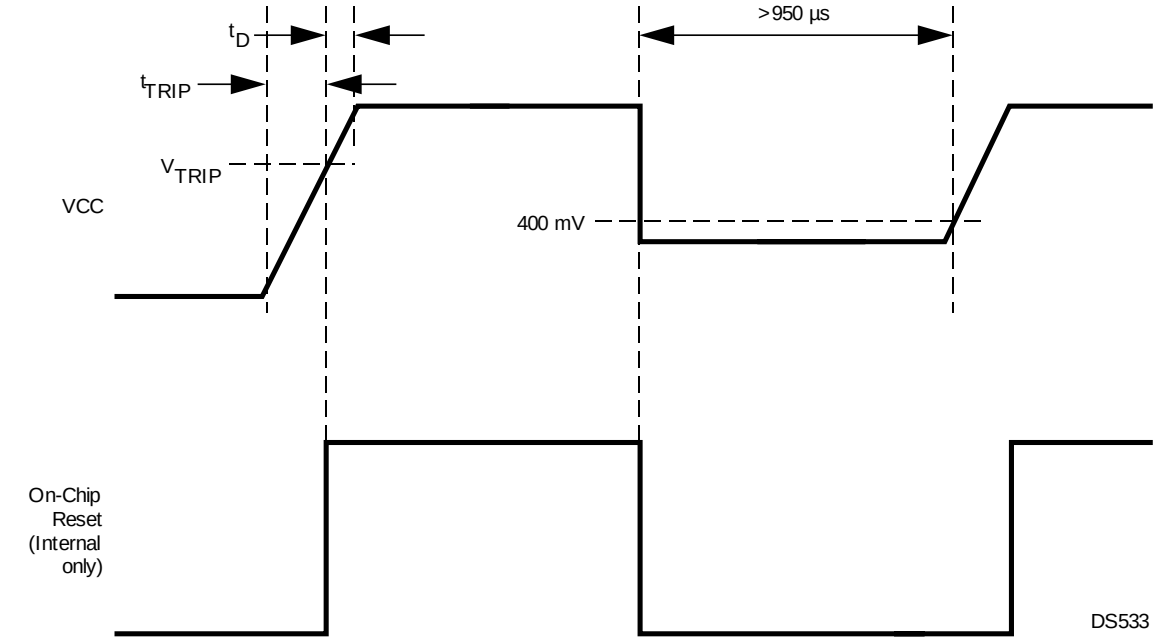


Figure 36-4. Power-On Reset

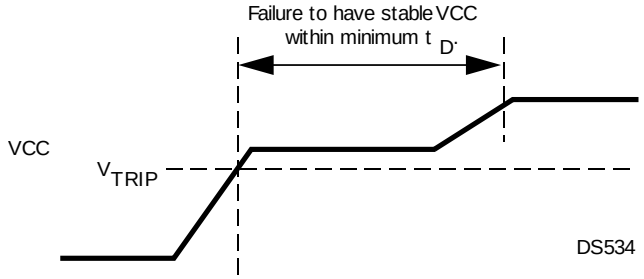


Figure 36-5. Bad Power-On Reset

36.8 UART Timing

Table 36-3. UART Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
UART INPUT SIGNALS					
t_{ls}	Input setup time RXD (asynchronous mode)	Before Rising Edge (RE) on PCLK Clock, See Figure 36-6			ns
t_{lh}	Input hold time aRXD (asynchronous mode)	After RE on PCLK Clock, See Figure 36-6			ns
t_{CKX}	CKX period (synchronous mode)	See Figure 36-7	250		ns
t_{RXS}	RXD setup time (synchronous mode)	Before Falling Edge (FE) on CKX, See Figure 36-7	40		ns
t_{RXH}	RXD hold time (synchronous mode)	Before Falling Edge (FE) on CKX, See Figure 36-7	40		ns
UART OUTPUT SIGNALS					
t_{COv1}	TXD output valid (all signals with propagation delay from CLKRE)	After RE on PCLK Clock, See Figure 36-6		35	ns
t_{TXD}	TXD output valid	After RE on CKX, See Figure 36-7		40	ns

(1) Specified by design.

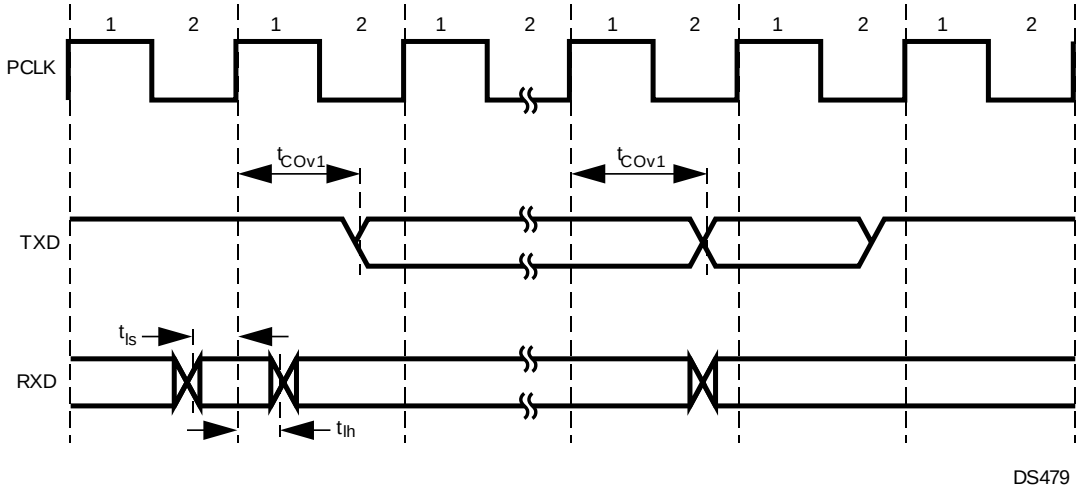


Figure 36-6. UART Asynchronous Mode Timing

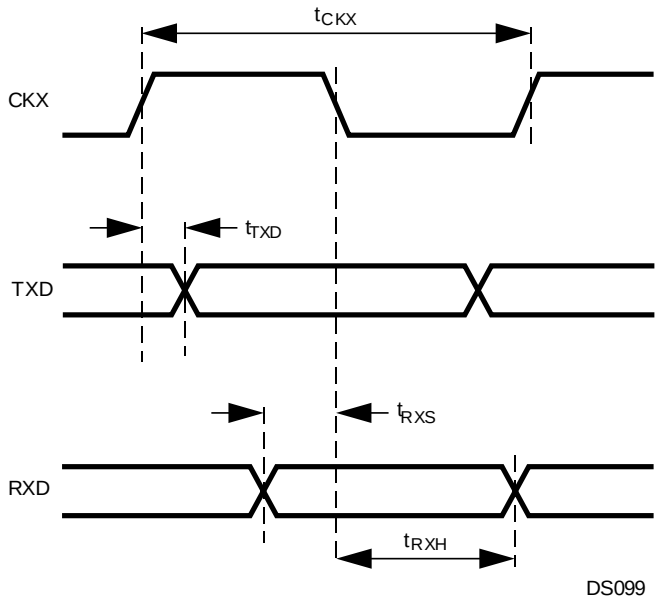


Figure 36-7. UART Synchronous Mode Timing

36.9 I/O Port Timing

Table 36-4. I/O Port Timing

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
I/O PORT INPUT SIGNALS					
t_{IS}	Input setup time	Before Rising Edge (RE) on PCLK Clock, See Figure 36-8			ns
t_{Ih}	Input hold time	After RE on PCLK Clock, See Figure 36-8			ns
I/O PORT OUTPUT SIGNALS					
t_{COv1}	Output valid time	After RE on PCLK Clock, See Figure 36-8		15	ns
t_{TXD}	Output Floating Time	After RE on PCLK Clock, See Figure 36-8		15	ns

(1) Specified by design.

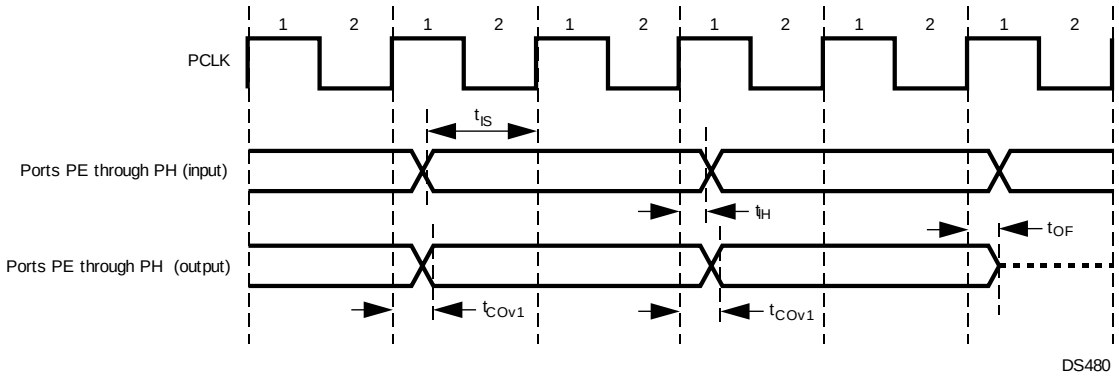


Figure 36-8. I/O Port Timing

36.10 Advanced Audio Interface (AAI) Timing

Table 36-5. Advanced Audio Interface (AAI) Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
AAI INPUT SIGNALS					
t_{RDS}	Receive Data Setup Time	Before Falling Edge (FE) on SRCLK, See Figure 36-9 and Figure 36-11	20		ns
t_{RDH}	Receive Data Hold Time	After FE on SRCLK, See Figure 36-9 and Figure 36-11	20		ns
t_{FSS}	Frame Sync Setup Time	Before Rising Edge (RE) on SRCLK , See Figure 36-9	20		ns
t_{FSH}	Frame Sync Hold Time	After RE on SRCLK, See Figure 36-9	20		ns
AAI OUTPUT SIGNALS					
t_{CP}	Receive/Transmit Clock Period	RE on SRCLK/SCK to RE on SRCLK/SCK , SeeFigure 36-9	976.6		ns
t_{CL}	Receive/Transmit Low Time	FE on SRCLK/SCK to RE on SRCLK/SCK , See Figure 36-9	488.3		ns
t_{CH}	Receive/Transmit High Tim	RE on SRCLK/SCK to FE on SRCLK/SCK, See Figure 36-9	488.3		ns
t_{FSVH}	Frame Sync Valid High	RE on SRCLK/SCK to RE on SRFS/SFS , See Figure 36-9 and Figure 36-11		20	ns
t_{FSVL}	Frame Sync Valid Low	RE on SRCLK/SCK to FE on SRFS/SFS, See Figure 36-9 and Figure 36-11		20	ns
t_{TDV}	Transmit Data Valid	RE on SCK to STD Valid, See Figure 36-10 and Figure 36-12		20	ns

(1) Specified by design.

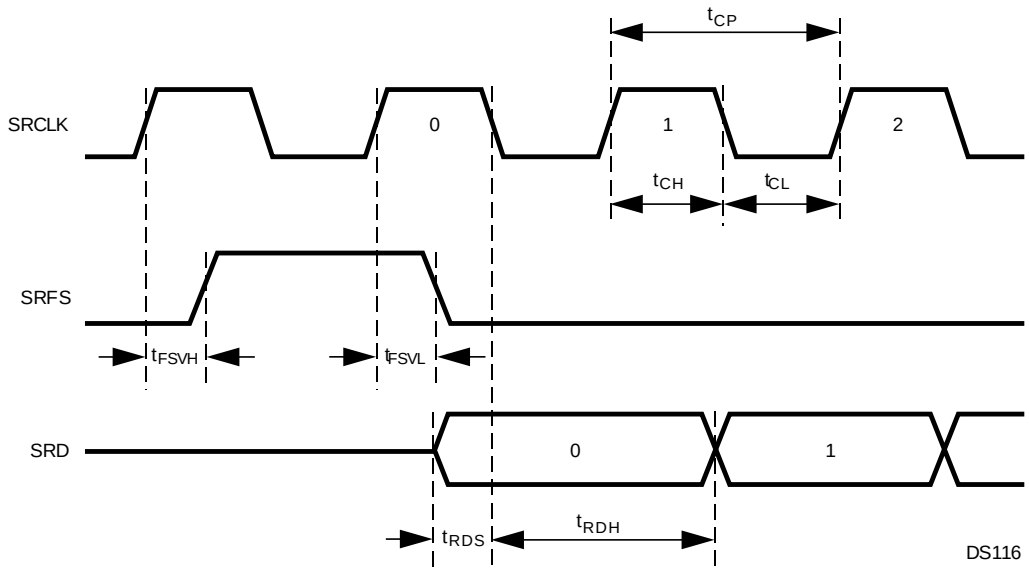


Figure 36-9. Receive Timing, Short Frame Sync

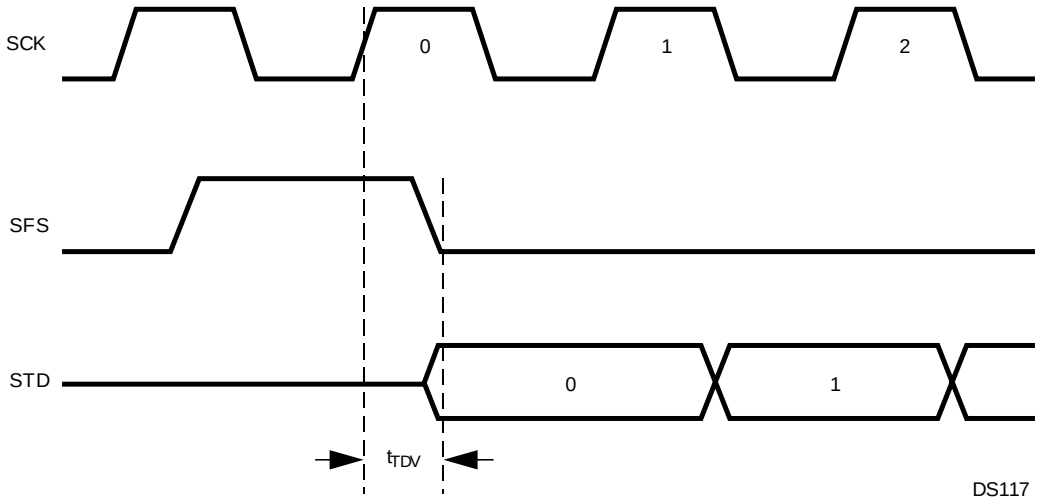


Figure 36-10. Transmit Timing, Short Frame Sync

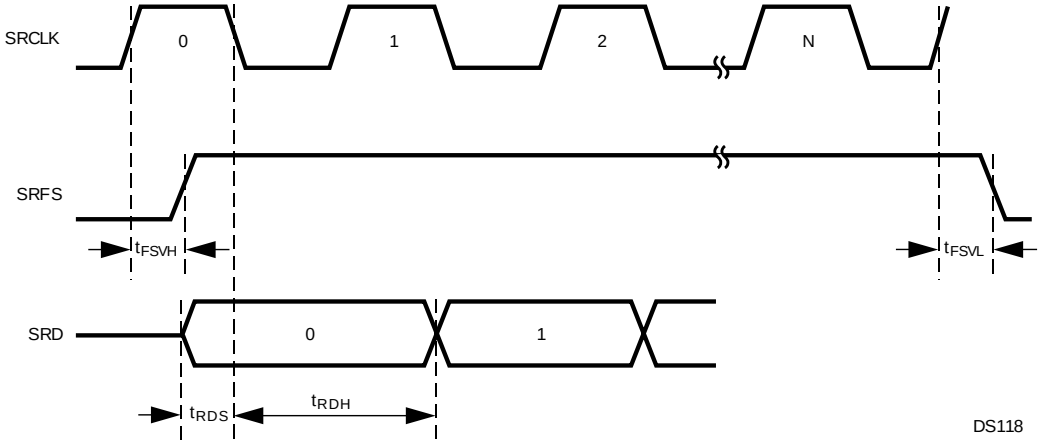


Figure 36-11. Receive Timing, Long Frame Sync

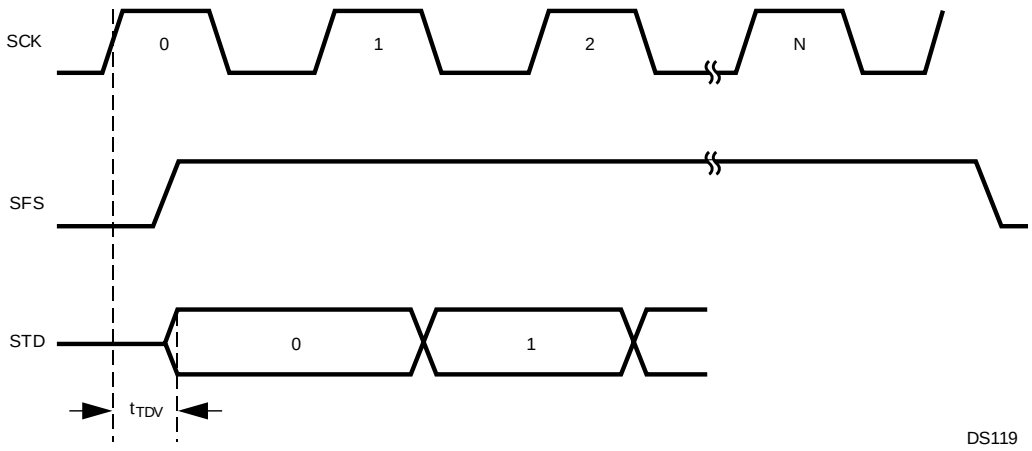


Figure 36-12. Transmit Timing, Long Frame Sync

36.11 Microwire/SPI Timing

Table 36-6. Microwire/SPI Timing

DESCRIPTION[1]		REFERENCE	MIN	MAX	UNIT
MICROWIRE/SPI INPUT SIGNALS					
t_{MSKh}	Microwire Clock High	At 2.0V (both edges), See Figure 36-13	80		ns
t_{MSKl}	Microwire Clock Low	At 0.8V (both edges), See Figure 36-13	80		ns
t_{MSKp}	Microwire Clock Period	SCIDL bit = 0; Rising Edge (RE) MSKn to next RE MSKn, See Figure 36-13	200		ns
		SCIDL bit = 1; Falling Edge (FE) MSKn to next FE MSKn, See Figure 36-14	200		
t_{MSKh}	MSKn Hold (slave only)	After $\overline{MWCSn}$ goes inactive, See Figure 36-13	40		ns
t_{MSKs}	MSKn Setup (slave only)	Before $\overline{MWCSn}$ goes active, See Figure 36-13	80		ns
t_{MWCSh}	$\overline{MWCSn}$ Hold (slave only)	SCIDL bit = 0: After FE MSKn, See Figure 36-13	40		ns
		SCIDL bit = 1: After RE MSKn, See Figure 36-14	40		
t_{MWCSs}	$\overline{MWCSn}$ Setup (slave only)	SCIDL bit = 0: Before RE MSKn, See Figure 36-13	80		ns
		SCIDL bit = 1: Before FE MSKn, See Figure 36-14	80		
t_{MDIh}	Microwire Data In Hold (master)	Normal Mode: After RE MSKn, See Figure 36-13	0		ns
		Alternate Mode: After FE MSKn, See Figure 36-15	0		
	Microwire Data In Hold (slave)	Normal Mode: After RE MSKn, See Figure 36-13	40		ns
		Alternate Mode: After FE MSKn, See Figure 36-15	40		
t_{MDIs}	Microwire Data In Setup	Normal Mode: Before RE MSKn, See Figure 36-13	80		ns
		Alternate Mode: Before FE MSKn, See Figure 36-15	80		
MICROWIRE/SPI OUTPUT SIGNALS					
t_{MSKh}	Microwire Clock High	At 2.0 V (both edges), See Figure 36-13	40		ns
t_{MSKl}	Microwire Clock Low	At 0.8 V (both edges), See Figure 36-13	40		ns
t_{MSKp}	Microwire Clock Period	SCIDL bit = 0: Rising Edge (RE) MSKn to next RE MSKn, See Figure 36-13			ns
		SCIDL bit = 1: Falling Edge (FE) MSKn to next FE MSKn, See Figure 36-14	100		
t_{MSKd}	MSKn Leading Edge Delayed (master only)	Data Out Bit 7 Valid, See Figure 36-13	0.5 t_{MSK}	1.5 t_{MSK}	ns
t_{MDOf}	Microwire Data Float (slave only)	After RE on $\overline{MWCSn}$, See Figure 36-13		25	ns
t_{MDOh}	Microwire Data Out Hold	Normal Mode: After FE MSKn, See Figure 36-13			ns
		Alternate Mode: After RE MSKn, SeeFigure 36-14	0		
t_{MDOnf}	Microwire Data No Float (slave only)	After FE on $\overline{MWCSn}$, See Figure 36-14	0	25	ns
t_{MDOv}	Microwire Data Out Valid	Normal Mode: After FE on MSKn, See Figure 36-13			ns
		Alternate Mode: After RE on MSKn, See Figure 36-13		25	
t_{MITOp}	MDODIn to MDIDOn (slave only)	Propagation Time Value is the same in all clocking modes of the Microwire, See Figure 36-17		25	ns

(1) Specified by design.

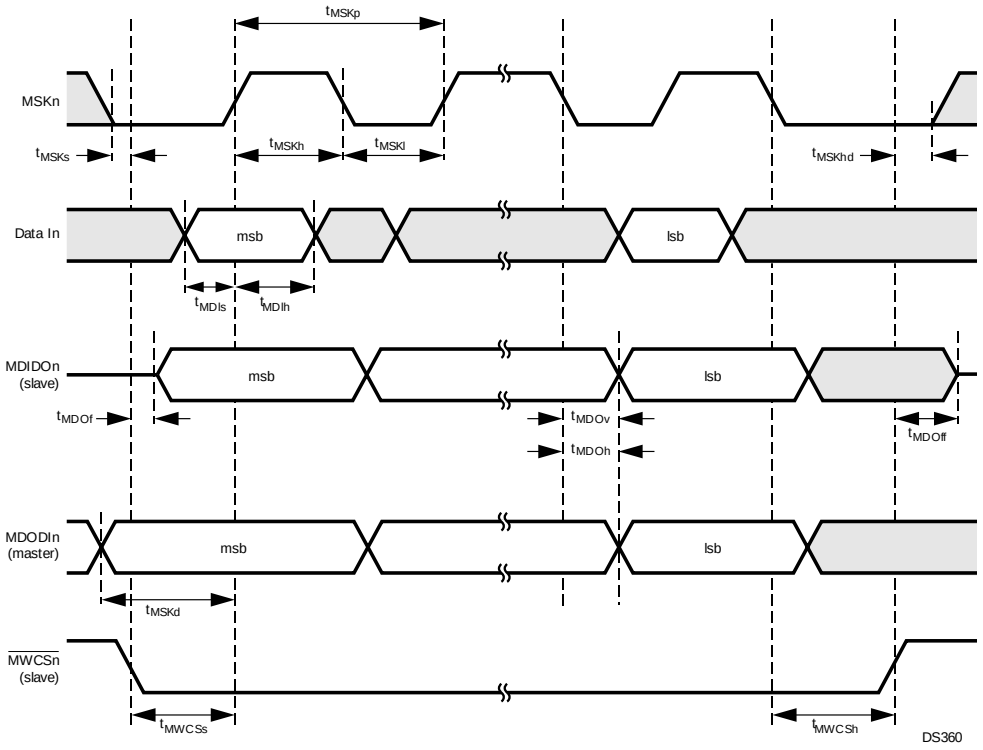


Figure 36-13. Microwire Transaction Timing, Normal Mode, SCIDL = 0

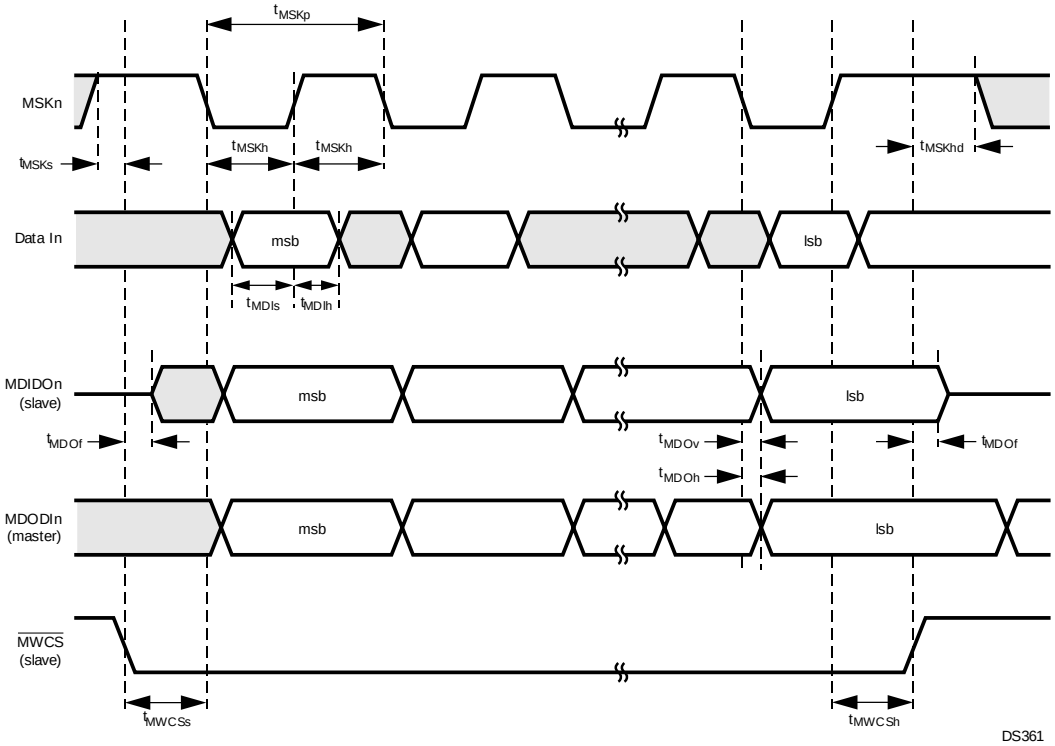


Figure 36-14. Microwire Transaction Timing, Normal Mode, SCIDL = 1

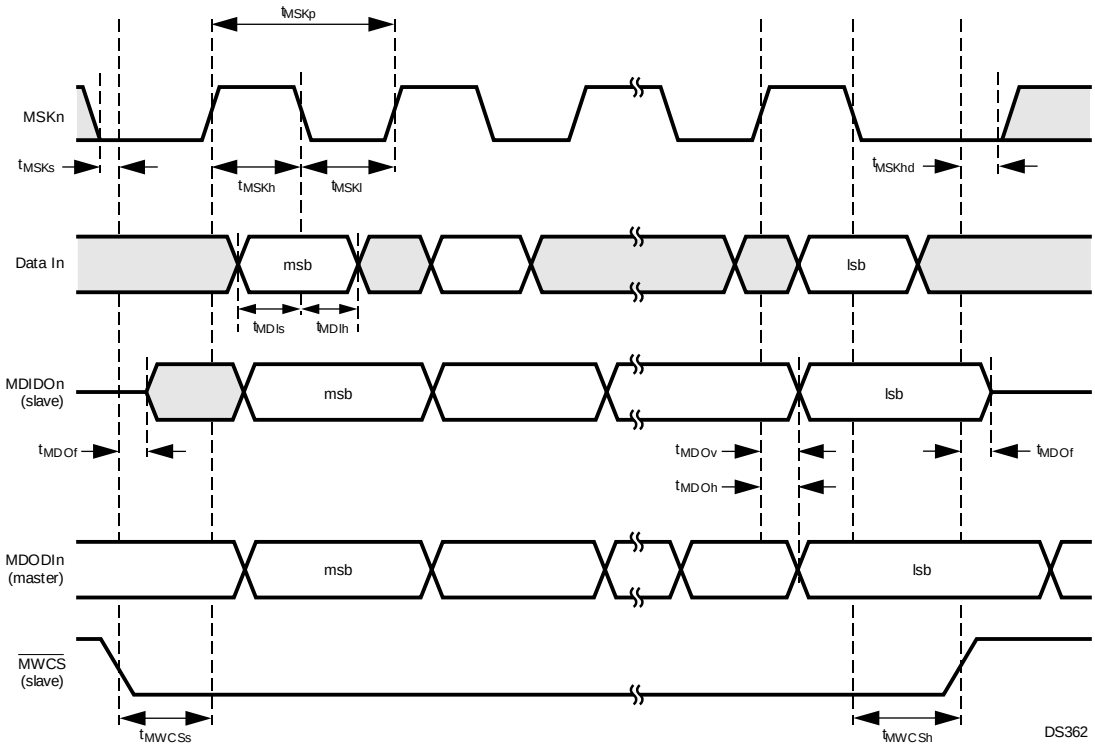


Figure 36-15. Microwire Transaction Timing, Alternate Mode, SCIDL = 0

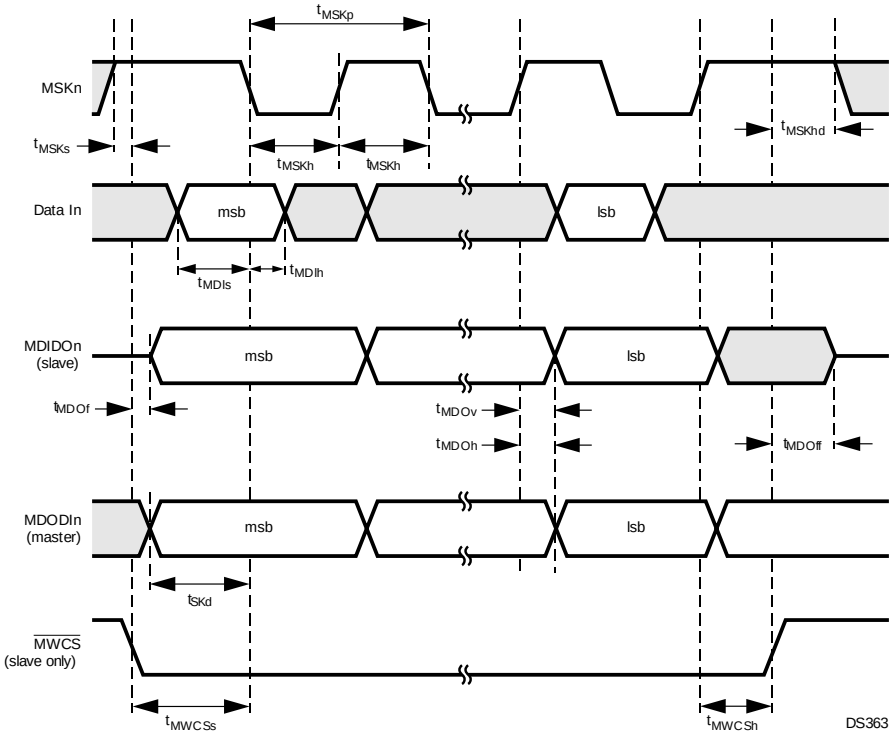


Figure 36-16. Microwire Transaction Timing, Alternate Mode, SCIDL = 1

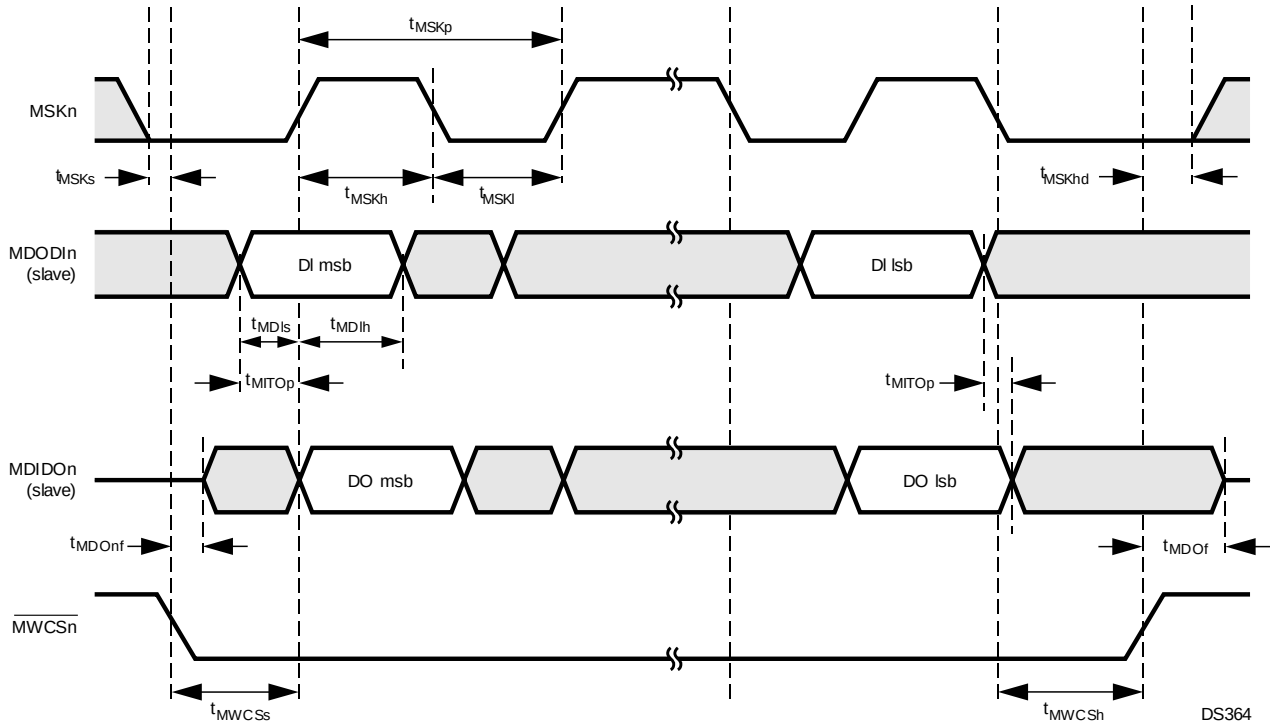


Figure 36-17. Microwire Transaction Timing, Data Echoed to Output, Normal Mode, SCIDL = 0, ECHO = 1, Slave Mode

36.12 ACCESS.BUS Timing

Table 36-7. ACCESS.BUS Timing

DESCRIPTION[1]		REFERENCE	MIN	MAX	UNIT
ACCESS.BUS INPUT SIGNALS					
t_{BUFi}	Bus free time between Stop and Start Condition	See Figure 36-19	$t_{SCLhigho}$		ns
t_{CSTOsi}	SCLn setup time	Before Stop Condition, See Figure 36-19	(8 × t_{CLK}) - t_{SCLri}		ns
t_{CSTRhi}	SCLn hold time	After Start Condition, See Figure 36-19	(8 × t_{CLK}) - t_{SCLri}		ns
t_{CSTRsi}	SCLn setup time	Before Start Condition, See Figure 36-19	(8 × t_{CLK}) - t_{SCLri}		ns
t_{DHCsi}	Data High setup time	Before SCLn Rising Edge (RE), See Figure 36-20	2 × t_{CLKp}		ns
t_{DLCsi}	Data Low setup time	Before SCLn RE, See Figure 36-19	2 × t_{CLKp}		ns
t_{SCLri}	SCLn signal rise time	See Figure 36-18		1000	ns
t_{SCLfi}	SCLn signal fall time	See Figure 36-18		300	ns
$t_{SCLlowi}$	SCLn low time	After SCLn Falling Edge (FE), See Figure 36-21	16 × t_{CLKp}		ns
$t_{SCLhighi}$	SCLn high time	After SCLn RE, See Figure 36-21	16 × t_{CLKp}		ns
t_{SDAri}	SDAn signal rise time	See Figure 36-18		1000	ns
t_{SDAfi}	SDAn signal fall time	See Figure 36-18		300	ns
t_{SDAhi}	SDAn hold time	After SCLn FE, See Figure 36-21	0		ns
t_{SDAsi}	SDAn setup time	Before SCLn RE, See Figure 36-21	2 × t_{CLKp}		ns
ACCESS.BUS OUTPUT SIGNALS					
t_{BUFo}	Bus free time between Stop and Start Condition	See Figure 36-19	$t_{SCLhigho}$		ns

(1) Specified by design.

Table 36-7. ACCESS.BUS Timing (continued)

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
t_{CSTOso}	SCLn setup time	Before Stop Condition, See Figure 36-19	$t_{SCLhigho}$		ns
t_{CSTRho}	SCLn hold time	After Start Condition, See Figure 36-19	$t_{SCLhigho}$		ns
t_{CSTRso}	SCLn setup time	Before Start Condition, See Figure 36-20	$t_{SCLhigho}$		ns
t_{DHCso}	Data High setup time	Before SCLn RE, See Figure 36-20	$t_{SCLhigho}$ -t_{SDAro}		ns
t_{DLCso}	Data Low setup time	Before SCLn RE, See Figure 36-19	$t_{SCLhigho}$ -t_{SDAfo}		ns
t_{SCLfo}	SCLn signal Fall time	See Figure 36-18		300(2)	ns
t_{SCLro}	SCLn signal Rise time	See Figure 36-18		See (3)	ns
$t_{SCLlowo}$	SCLn low time	After SCLn FE, See Figure 36-21	($K \times t_{CLK}$) -1(4)		ns
$t_{SCLhigho}$	SCLn high time	After SCLn RE, See Figure 36-21	($K \times t_{CLK}$) -1		ns
t_{SDAfo}	SDAn signal Fall time	See Figure 36-18		300	ns
t_{SDAro}	SDAn signal Rise time	See Figure 36-18			ns
t_{SDAho}(5)	SDAn hold time	After SCLn FE, See Figure 36-21	(7 × t_{CLK}) - t_{SCLfo}		ns
t_{SDAvo}	SDAn valid time	After SCLn FE, See Figure 36-21		(7 × t_{CLK}) + t_{RD}	ns

(2) Assuming signal's capacitance up to 400 pF.
(3) Depends on the signal's capacitance and the pullup value. Must be less than 1 ms.
(4) K is as specified in ACBnCTL2.SCLFRQ×2. K > 15.
(5) Note that the transmitter must internally provide a hold time of at least 300 ns on SDA to bridge the SCL fall time.

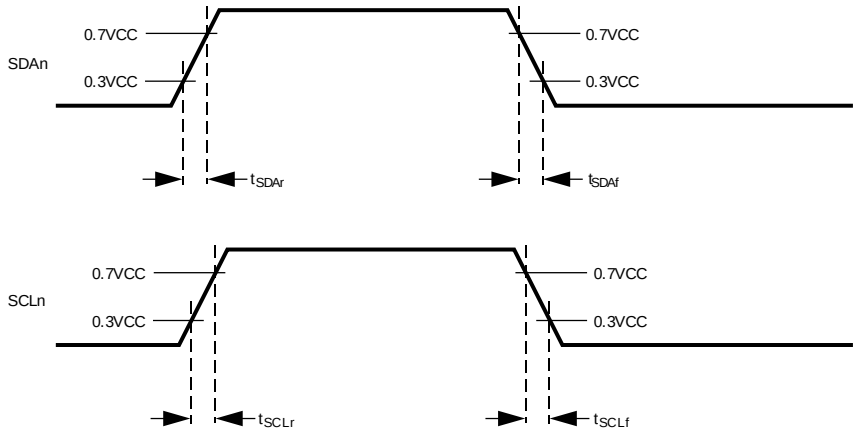
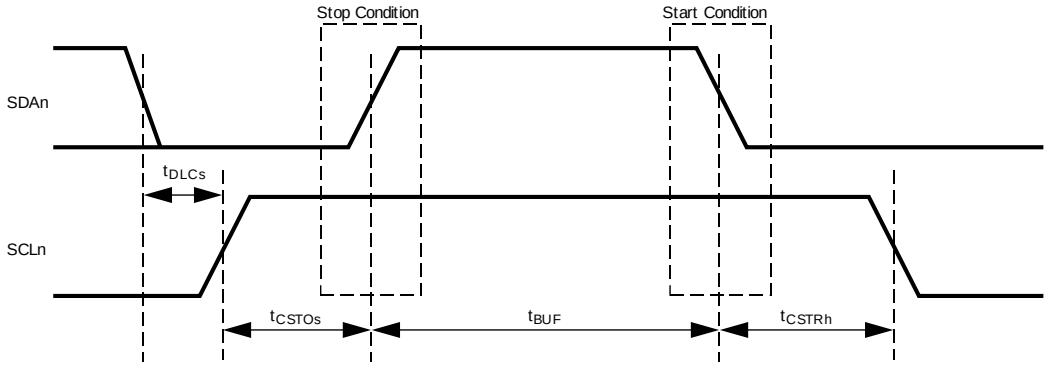
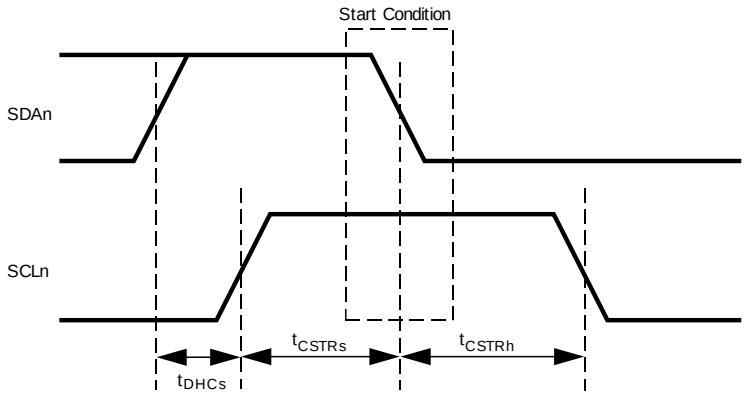


Figure 36-18. ACB Signals (SDAn and SCLn) Timing



Note: In the timing tables the parameter name is added with an "o" for output signal timing and "i" for input signal timing. DS366

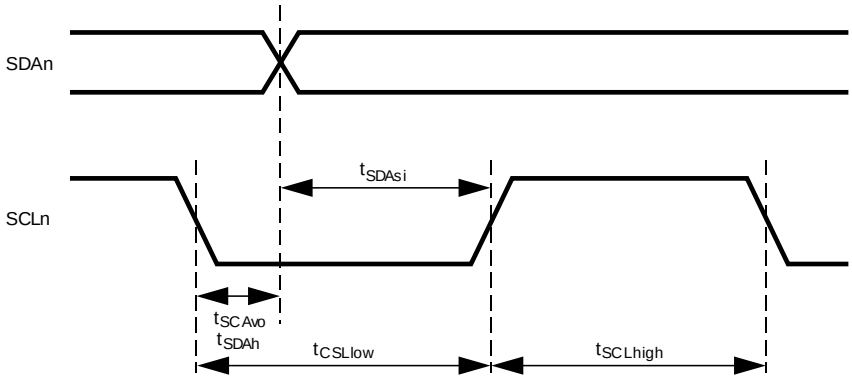
Figure 36-19. ACB Start and Stop Condition Timing



Note: In the timing tables the parameter name is added with an "o" for output signal timing and "i" for input signal timing.

DS367

Figure 36-20. ACB Start Condition Timing



Note: In the timing tables the parameter name is added with an "o" for output signal timing and "i" for input signal timing. unless the parameter already includes the suffix.

DS368

Figure 36-21. ACB Data Timing

36.13 USB Port AC Characteristics

Table 36-8. USB Port Signals

DESCRIPTION(1)		CONDITIONS(2)	MIN	TYP	MAX	UNIT
T_R	Rise time	C_L = 50 pF	4		20	ns
T_F	Fall Time	C_L = 50 pF	4		20	ns
T_{RFM}	Fall/Rise time matching (T_R/T_F)	C_L = 50 pF	90%		110%	
Z_{DRV}	Driver Output Impedance	C_L = 50 pF	28		43	Ω

(1) Specified by design.
(2) Waveforms measured at 10% to 90%.

36.14 I²S Interface Timing

Table 36-9. I²S Interface Signals

DESCRIPTION(1)		REFERENCE	LOWER LIMIT		UPPER LIMIT		UNIT
			MIN	MAX	MIN	MAX	
I²S INTERFACE TRANSMITTER SIGNALS							
T	I²S clock period	See Figure 36-22	T_t				ns
T_t	Minimum transmitter clock period	T > T_t, See Figure 36-22					ns
t_{CH}	I²S clock high	V_{ih}, See Figure 36-22	0.35 T_t (master)	0.35 T_t (slave)			ns
t_{CL}	I²S clock low	V_{il}, See Figure 36-22	0.35 T_t (master)	0.35 T_t (slave)			ns
t_{CR}	I²S clock rise time (slave mode)	See Figure 36-22			0.15 T_t		ns
t_{TD}	Delay	Rising Edge (RE) on I²S Clock, See Figure 36-22				0.8 T_t	ns
t_{TH}	Hold time	RE on I²S Clock, See Figure 36-22	0				ns
I²S INTERFACE RECEIVER SIGNALS							
T	I²S clock period	See Figure 36-22	T_t				ns
T_r	Minimum receiver clock period	T > T_t, See Figure 36-22					ns
t_{CH}	I²S clock high	V_{ih}, See Figure 36-22	0.35 T_r (master)	0.35 T_r (slave)			ns
t_{CL}	I²S clock low	V_{il}, See Figure 36-22	0.35 T_r (master)	0.35 T_r (slave)			ns
t_{RS}	Setup time	RE on I²S Clock		0.2 Tr			ns
t_{RH}	Hold time	RE on I²S Clock		0			

(1) Specified by design.

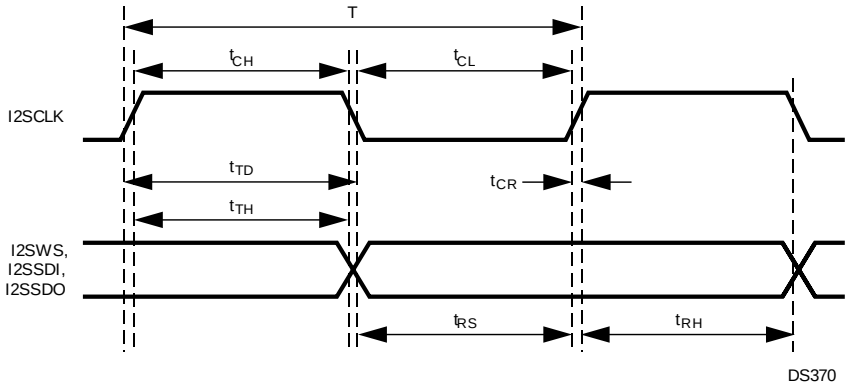


Figure 36-22. I²S Interface Timing

36.15 Multi-Function Timer (MFT) Timing

Table 36-10. Multi-Function Timer Input Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
t_{TAH}	TAn high time	Rising edge (RE) on PCLK clock, See Figure 36-23	T_{CLK} + 5		ns
t_{TAL}	TAn low time	RE on PCLK clock, See Figure 36-23	T_{CLK} + 5		ns
t_{TBH}	TBn high time	RE on PCLK clock, See Figure 36-23	T_{CLK} + 5		ns
t_{TBL}	TBn low time	RE on PCLK clock, See Figure 36-23	T_{CLK} + 5		ns

(1) Specified by design.

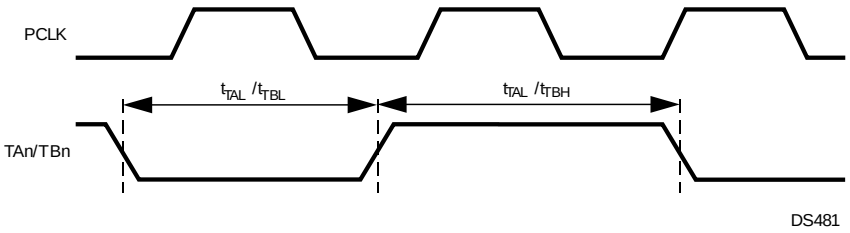


Figure 36-23. Multi-Function Timer Input Timing

36.16 Versatile Timing Unit (VTU) Timing

Table 36-11. Versatile Timing Unit Input Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
t_{TIOH}	TIOn_y Input High Time	Rising Edge (RE) on PCLK Clock, See Figure 36-24	1.5 × T_{CLK} + 5		ns
t_{TIOL}	TIOn_y Input Low Time	RE on PCLK Clock, See Figure 36-24	1.5 × T_{CLK} + 5		ns

(1) Specified by design.

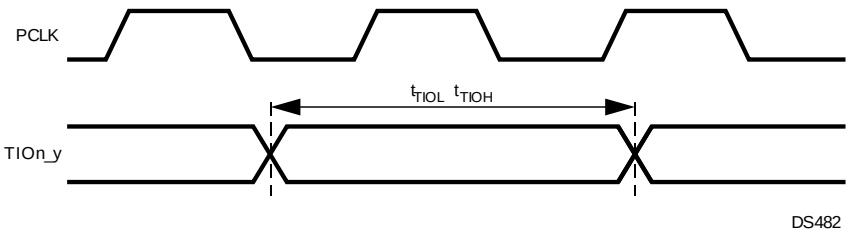


Figure 36-24. Versatile Timing Unit Input Timing

36.17 External Memory Interface

Table 36-12. External Memory Signals

DESCRIPTION(1)		REFERENCE	MIN	MAX	UNIT
EXTERNAL MEMORY INPUT SIGNALS					
t_{DSU}	Data Bus Input Setup	Before Rising Edge (RE) on HCLK Clock, See Figure 36-25	5		ns
t_{DIH}	Data Bus Input Hold	After RE on HCLK Clock , See Figure 36-25	0		ns
EXTERNAL MEMORY OUTPUT SIGNALS					
t_{AV}	Address Bus Valid	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26		9	ns
t_{AH}	Address Bus Hold	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26	0		ns
t_{CSL}	Chip Select Low	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26		9	ns
t_{CSH}	Chip Select Hold	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26	0		ns

(1) Specified by design.

Table 36-12. External Memory Signals (continued)

DESCRIPTION[1]		REFERENCE	MIN	MAX	UNIT
t_{BEL}	Byte Enable Low	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26		9	ns
t_{BEH}	Byte Enable Hold	After RE on HCLK Clock, See Figure 36-25 and Figure 36-26	0		ns
t_{OEL}	Output Enable Low	After RE on HCLK Clock, See Figure 36-25		9	ns
t_{OEH}	Output Enable Hold	After RE on HCLK Clock, See Figure 36-25	0		ns
t_{WEL}	Write Enable Low	After RE on HCLK Clock, See Figure 36-26		9	ns
t_{WEH}	Write Enable Hold	After RE on HCLK Clock, See Figure 36-26	0		ns
t_{DOV}	Data Bus Output Valid	After RE on HCLK Clock, See Figure 36-26		9	ns
t_{DOH}	Data Bus Output Hold	After RE on HCLK Clock, See Figure 36-26	0		ns

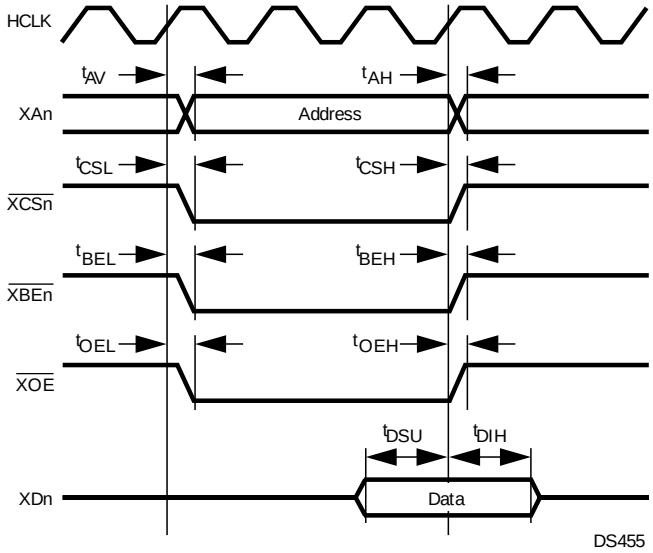


Figure 36-25. Memory Read Cycle

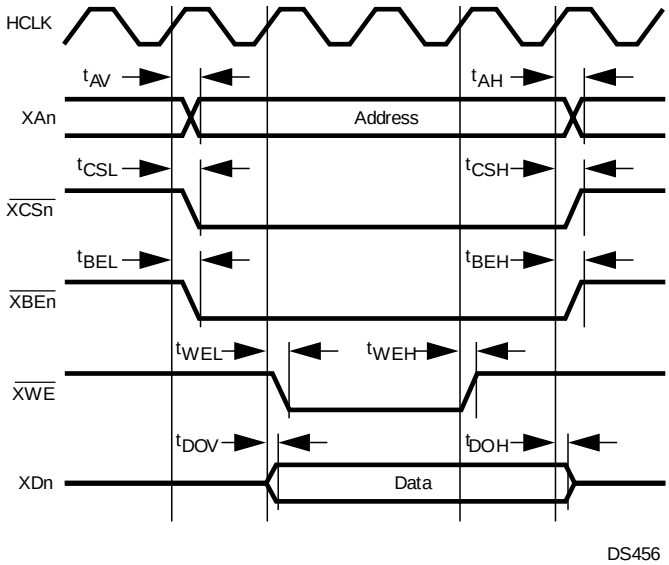
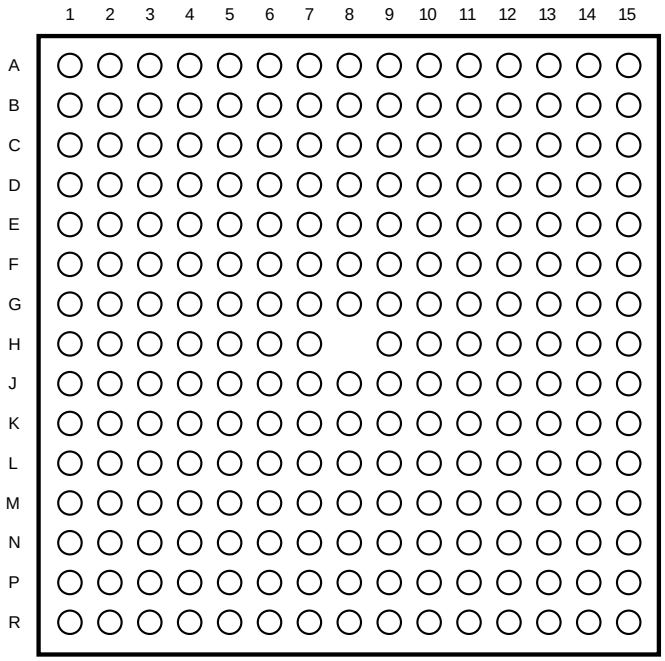


Figure 36-26. Memory Write Cycle

37 Pin Assignments

FBGA PACKAGE
224-PIN
(TOP VIEW)



DS337

Table 37-1. Pin Assignments for 224-Pin Package

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
A1	No connect	IOVCC
A2	PE10	IOVCC
A3	PE12	IOVCC
A4	PH13	IOVCC
A5	PH2	IOVCC
A6	PG14	IOVCC
A7	IOGND	
A8	PF1	RFVCC
A9	PF5	RFVCC
A10	PG8	IOVCC
A11	PG4	IOVCC
A12	PH12	IOVCC
A13	PH10	IOVCC
A14	PH8	IOVCC
A15	TCRP	CODEC OUT
B1	PH14	IOVCC
B2	PE8	IOVCC
B3	PE9	IOVCC
B4	PE13	IOVCC
B5	PH3	IOVCC
B6	PG15	IOVCC
B7	PG10	IOVCC
B8	PF0	RFVCC

Table 37-1. Pin Assignments for 224-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
B9	PF6	RFVCC
B10	IOVCC	
B11	IOGND	
B12	PH11	IOVCC
B13	PH9	IOVCC
B14	PH7	IOVCC
B15	TCLP	CODEC OUT
C1	SDA1	IOVCC
C2	PH15	IOVCC
C3	PE11	IOVCC
C4	IOGND	
C5	PE15	IOVCC
C6	PF11	IOVCC
C7	RFCE	RFVCC
C8	RFDATA	RFVCC
C9	VCC	
C10	PG5	IOVCC
C11	PG1	IOVCC
C12	IOVCC	
C13	TCDACGND	CODEC OUT
C14	TCRN	CODEC OUT
C15	TCLN	CODEC OUT
D1	IOVCC	
D2	PE6	IOVCC
D3	SCL1	IOVCC
D4	PE14	IOVCC
D5	PH1	IOVCC
D6	PG13	IOVCC
D7	PF4	RFVCC
D8	RFVCC	
D9	PG9	IOVCC
D10	PG3	IOVCC
D11	PF7	IOVCC
D12	TCADCVCC	CODEC IN
D13	TCADCGND	CODEC IN
D14	TCDACVCC	CODEC OUT
D15	TCVCM2	CODEC IN
E1	DSPTDI	IOVCC
E2	DSPTMS	IOVCC
E3	PE4	IOVCC
E4	PE7	IOVCC
E5	IOVCC	
E6	PG12	IOVCC
E7	PF3	RFVCC
E8	RFGND	
E9	PG6	IOVCC
E10	PG0	IOVCC

Table 37-1. Pin Assignments for 224-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
E11	TCMIC1P	CODEC IN
E12	TCVRFN	CODEC IN
E13	TCVBUF2	CODEC IN
E14	TCMIC2N	CODEC IN
E15	TCMIC2P	CODEC IN
F1	XA22	IOVCC
F2	XA0	IOVCC
F3	IOGND	
F4	$\overline{\text{MSEO}}$	IOVCC
F5	PE5	IOVCC
F6	PG11	IOVCC
F7	PF2	RFVCC
F8	GND	
F9	PG2	IOVCC
F10	ADGND	
F11	ADVCC	
F12	TCVCM1	CODEC IN
F13	TCMIC1N	CODEC IN
F14	TCVBUF1	CODEC IN
F15	TCVRFP	CODEC IN
G1	XA20	IOVCC
G2	XA2	IOVCC
G3	XA21	IOVCC
G4	IOVCC	
G5	DSPTDO	IOVCC
G6	MCKO	IOVCC
G7	CLKIN	RFVCC
G8	PG7	IOVCC
G9	ADC6	ADVCC
G10	ADC8	ADVCC
G11	ADC3	ADVCC
G12	ADC2	ADVCC
G13	ADC1	ADVCC
G14	VREF	ADVCC
G15	ADC0	ADVCC
H1	XA5	IOVCC
H2	XA18	IOVCC
H3	XA4	IOVCC
H4	XA19	IOVCC
H5	XA3	IOVCC
H6	XA1	IOVCC
H7	DSPTCK	IOVCC
H9	PF14	IOVCC
H10	PH0	IOVCC
H11	PH6	IOVCC
H12	ADC7	ADVCC
H13	ADC5	ADVCC

Table 37-1. Pin Assignments for 224-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
H14	ADC4	ADVCC
H15	ADC9	ADVCC
J1	IOVCC	
J2	GND	
J3	XA7	IOVCC
J4	XA16	IOVCC
J5	XA6	IOVCC
J6	XA17	IOVCC
J7	IOGND	
J8	XD6	IOVCC
J9	XD9	IOVCC
J10	XD13	IOVCC
J11	PH5	IOVCC
J12	PF15	IOVCC
J13	IOGND	
J14	VCC	
J15	GND	
K1	XA8	IOVCC
K2	XA14	IOVCC
K3	XA9	IOVCC
K4	XA15	IOVCC
K5	IOGND	
K6	VCC	
K7	IOGND	
K8	IOGND	
K9	XD21	IOVCC
K10	PLLVCC	
K11	PLLGND	
K12	$\overline{\text{RESET}}$	IOVCC
K13	PH4	IOVCC
K14	PF13	IOVCC
K15	PF12	IOVCC
L1	XA10	IOVCC
L2	XA12	IOVCC
L3	XA11	IOVCC
L4	IOVCC	
L5	XA13	IOVCC
L6	XD3	IOVCC
L7	IOVCC	
L8	XD8	IOVCC
L9	XD11	IOVCC
L10	XD18	IOVCC
L11	XD16	IOVCC
L12	PF8	IOVCC
L13	X2CK0	PLLVCC
L14	X2CKI	PLLVCC
L15	IOVCC	

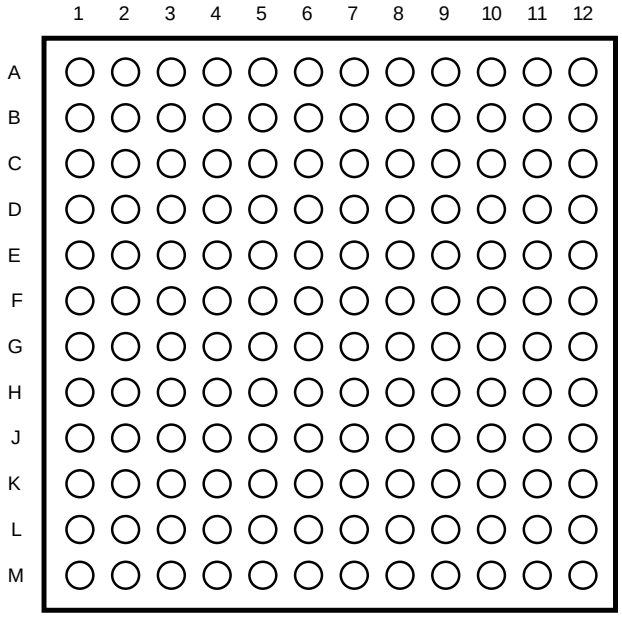
Table 37-1. Pin Assignments for 224-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
M1	$\overline{\text{XWE}}$	IOVCC
M2	$\overline{\text{XBE1}}$	IOVCC
M3	$\overline{\text{XBE0}}$	IOVCC
M4	$\overline{\text{XOE}}$	IOVCC
M5	IOVCC	
M6	XD4	IOVCC
M7	XD24	IOVCC
M8	XD23	IOVCC
M9	XD20	IOVCC
M10	IOGND	
M11	ENV0	IOVCC
M12	$\overline{\text{RDY}}$	IOVCC
M13	UGND	
M14	X1CKO	PLLVCC
M15	X1CKI	PLLVCC
N1	IOGND	
N2	$\overline{\text{XCS1}}$	IOVCC
N3	$\overline{\text{XBE2}}$	IOVCC
N4	XD30	IOVCC
N5	XD28	IOVCC
N6	XD26	IOVCC
N7	VCC	
N8	XD22	IOVCC
N9	IOVCC	
N10	XD17	IOVCC
N11	TDI	IOVCC
N12	TCK	IOVCC
N13	UVCC	
N14	VBUS	UVCC
N15	D+	UVCC
P1	$\overline{\text{XBE3}}$	IOVCC
P2	$\overline{\text{XCS2}}$	IOVCC
P3	XD31	IOVCC
P4	XD2	IOVCC
P5	XD27	IOVCC
P6	XD25	IOVCC
P7	GND	
P8	IOGND	
P9	XD19	IOVCC
P10	XD15	IOVCC
P11	TDO	IOVCC
P12	PF10	IOVCC
P13	PE1	IOVCC
P14	PE0	IOVCC
P15	D–	UVCC
R1	$\overline{\text{XCS0}}$	IOVCC
R2	XD0	IOVCC

Table 37-1. Pin Assignments for 224-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
R3	XD1	IOVCC
R4	XD29	IOVCC
R5	XD5	IOVCC
R6	XD7	IOVCC
R7	IOVCC	
R8	XD10	IOVCC
R9	XD12	IOVCC
R10	XD14	IOVCC
R11	ENV1	IOVCC
R12	TMS	IOVCC
R13	PF9	IOVCC
R14	PE3	IOVCC
R15	PE2	IOVCC

FBGA PACKAGE
144-PIN
(TOP VIEW)



DS426

Table 37-2. Pin Assignments for 144-Pin Package

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
A1	PE9	IOVCC
A2	PE10	IOVCC
A3	PE13	IOVCC
A4	PH3	IOVCC
A5	PF11	IOVCC
A6	PF3	RFVCC
A7	PF2	RFVCC
A8	CLKIN	RFVCC
A9	GND	
A10	PG1	IOVCC

Table 37-2. Pin Assignments for 144-Pin Package (continued)

PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
A11	PH9	IOVCC
A12	PH8	IOVCC
B1	PE5	IOVCC
B2	PE7	IOVCC
B3	PE8	IOVCC
B4	PE12	IOVCC
B5	PH2	IOVCC
B6	RFCE	RFVCC
B7	PF0	RFVCC
B8	VCC	
B9	PG0	IOVCC
B10	TCRP	CODEC OUT
B11	TCRN	CODEC OUT
B12	TCLN	CODEC OUT
C1	XA2	IOVCC
C2	XA0	IOVCC
C3	PE6	IOVCC
C4	IOGND	
C5	PE15	IOVCC
C6	PF4	RFVCC
C7	PF1	RFVCC
C8	IOVCC	
C9	PG2	IOVCC
C10	TCDACVCC	CODEC OUT
C11	TCDACGND	CODEC OUT
C12	TCLP	CODEC OUT
D1	XA19	IOVCC
D2	XA3	IOVCC
D3	XA1	IOVCC
D4	PE11	IOVCC
D5	PE14	IOVCC
D6	IOGND	
D7	RFDATA	RFVCC
D8	PG3	IOVCC
D9	PH10	IOVCC
D10	TCADCGND	CODEC IN
D11	TCADCVCC	CODEC IN
D12	TCVBUF2	CODEC IN
E1	XA5	IOVCC
E2	XA18	IOVCC
E3	XA4	IOVCC
E4	XA20	IOVCC
E5	IOVCC	
E6	IOVCC	
E7	RFVCC	
E8	PH7	IOVCC
E9	TCMIC2N	CODEC IN

Table 37-2. Pin Assignments for 144-Pin Package (continued)

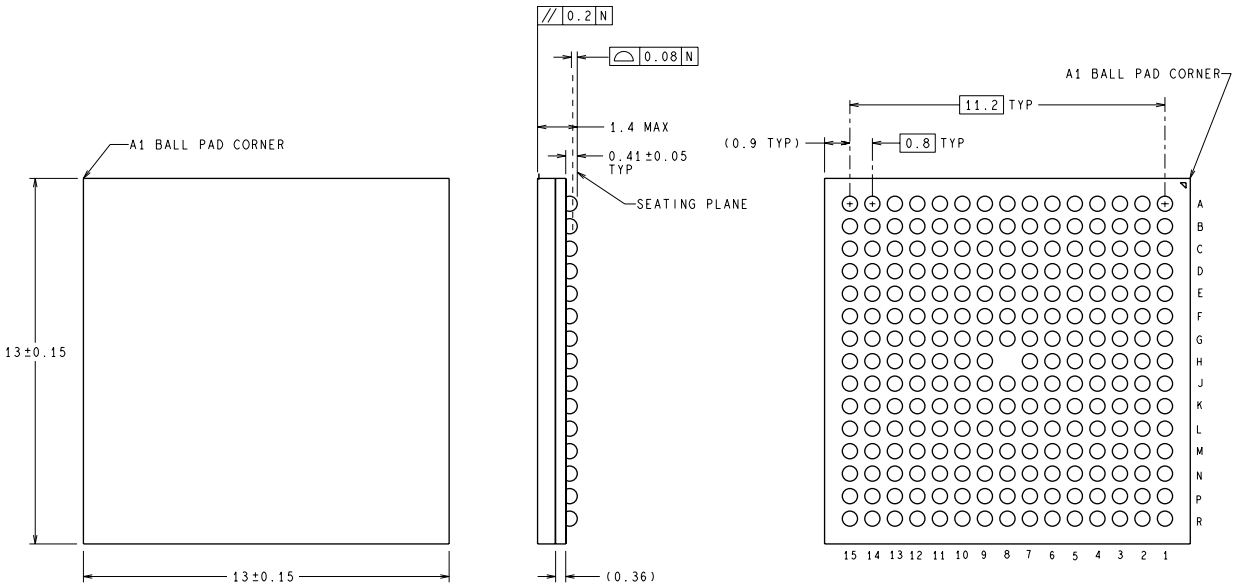
PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
E10	TCMIC2P	CODEC IN
E11	TCVRFN	CODEC IN
E12	TCMIC1P	CODEC IN
F1	XA7	IOVCC
F2	XA16	IOVCC
F3	XA6	IOVCC
F4	XA17	IOVCC
F5	IOGND	
F6	PH1	IOVCC
F7	RFGND	
F8	TCVCM2	CODEC IN
F9	TCMIC1N	CODEC IN
F10	TCVBUF1	CODEC IN
F11	TCVRFP	CODEC IN
F12	TCVCM1	CODEC IN
G1	IOVCC	
G2	GND	
G3	VCC	
G4	IOGND	
G5	XA12	IOVCC
G6	GND	
G7	TDO	IOVCC
G8	PE3	IOVCC
G9	ADC1	ADVCC
G10	ADC0	ADVCC
G11	ADGND	
G12	ADVCC	
H1	XA15	IOVCC
H2	XA8	IOVCC
H3	XA14	IOVCC
H4	XA9	IOVCC
H5	$\overline{\text{XWE}}$	IOVCC
H6	IOVCC	
H7	ENV1	IOVCC
H8	X2CK0	PLLVCC
H9	X2CKI	PLLVCC
H10	$\overline{\text{RESET}}$	IOVCC
H11	VCC	
H12	GND	
J1	XA13	IOVCC
J2	XA10	IOVCC
J3	$\overline{\text{XOE}}$	IOVCC
J4	XD2	IOVCC
J5	XD5	IOVCC
J6	XD9	IOVCC
J7	XD15	IOVCC
J8	TCK	IOVCC

Table 37-2. Pin Assignments for 144-Pin Package (continued)

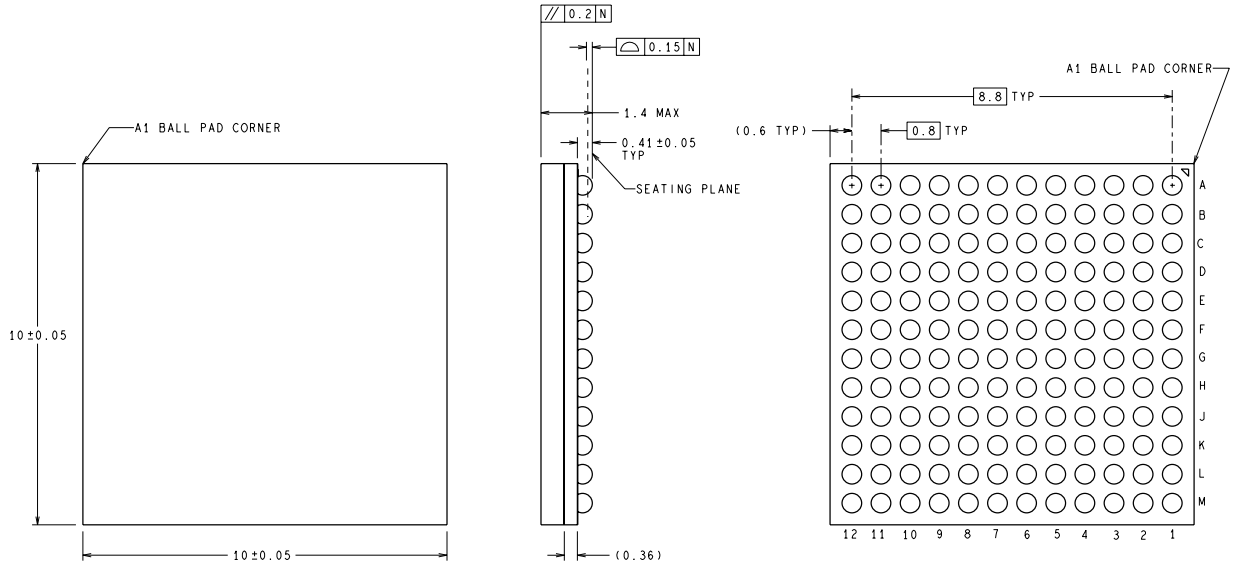
PIN	SIGNAL NAME	POWER SUPPLY DOMAIN
J9	X1CKI	PLLVCC
J10	X1CKO	PLLVCC
J11	PLLGND	
J12	PLLVCC	
K1	XA11	IOVCC
K2	$\overline{\text{XBE0}}$	IOVCC
K3	XCS0	IOVCC
K4	XD4	IOVCC
K5	XD6	IOVCC
K6	XD10	IOVCC
K7	XD13	IOVCC
K8	TMS	IOVCC
K9	PF10	IOVCC
K10	UVCC	
K11	UGND	
K12	D+	UVCC
L1	$\overline{\text{XBE1}}$	IOVCC
L2	$\overline{\text{XCS1}}$	IOVCC
L3	XD0	IOVCC
L4	XD3	IOVCC
L5	VCC	
L6	IOGND	
L7	XD14	IOVCC
L8	TDI	IOVCC
L9	PF8	IOVCC
L10	PE0	IOVCC
L11	VBUS	UVCC
L12	D–	UVCC
M1	XD1	IOVCC
M2	IOVCC	
M3	IOGND	
M4	XD7	IOVCC
M5	XD8	IOVCC
M6	XD11	IOVCC
M7	XD12	IOVCC
M8	ENV0	IOVCC
M9	$\overline{\text{RDY}}$	IOVCC
M10	PF9	IOVCC
M11	PE2	IOVCC
M12	PE1	IOVCC

37.1 Physical Dimensions (millimeters) unless otherwise noted

FBGA PACKAGE
224-PINS



FBGA PACKAGE
144-PINS



PACKAGING INFORMATION

Orderable part number	Status (1)	Material type (2)	Package | Pins	Package qty | Carrier	RoHS (3)	Lead finish/ Ball material (4)	MSL rating/ Peak reflow (5)	Op temp (°C)	Part marking (6)
CP3SP33SMS/NOPB	Active	Production	NFBGA (NZN) | 224	160 | JEDEC TRAY (10+1)	Yes	SNAGCU	Level-3-260C-168 HR	-40 to 85	CP3SP33SMS
CP3SP33SMS/NOPB.A	Active	Production	NFBGA (NZN) | 224	160 | JEDEC TRAY (10+1)	Yes	SNAGCU	Level-3-260C-168 HR	-40 to 85	CP3SP33SMS
CP3SP33SMSX/NOPB	Active	Production	NFBGA (NZN) | 224	1000 | LARGE T&R	Yes	SNAGCU	Level-3-260C-168 HR	-40 to 85	CP3SP33SMS
CP3SP33SMSX/NOPB.A	Active	Production	NFBGA (NZN) | 224	1000 | LARGE T&R	Yes	SNAGCU	Level-3-260C-168 HR	-40 to 85	CP3SP33SMS

(1) **Status:** For more details on status, see our product life cycle.

(2) **Material type:** When designated, preproduction parts are prototypes/experimental devices, and are not yet approved or released for full production. Testing and final process, including without limitation quality assurance, reliability performance testing, and/or process qualification, may not yet be complete, and this item is subject to further changes or possible discontinuation. If available for ordering, purchases will be subject to an additional waiver at checkout, and are intended for early internal evaluation purposes only. These items are sold without warranties of any kind.

(3) **RoHS values:** Yes, No, RoHS Exempt. See the TI RoHS Statement for additional information and value definition.

(4) **Lead finish/Ball material:** Parts may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead finish/Ball material values may wrap to two lines if the finish value exceeds the maximum column width.

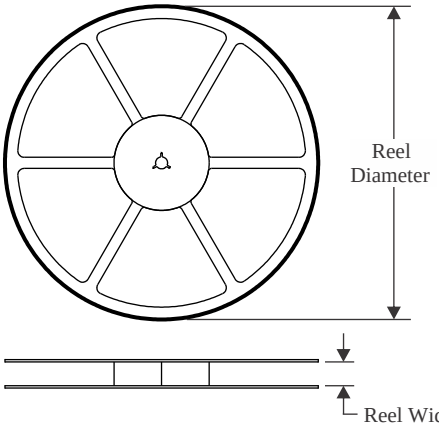
(5) **MSL rating/Peak reflow:** The moisture sensitivity level ratings and peak solder (reflow) temperatures. In the event that a part has multiple moisture sensitivity ratings, only the lowest level per JEDEC standards is shown. Refer to the shipping label for the actual reflow temperature that will be used to mount the part to the printed circuit board.

(6) **Part marking:** There may be an additional marking, which relates to the logo, the lot trace code information, or the environmental category of the part.

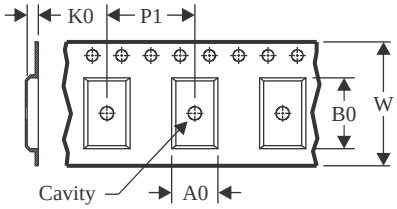
Multiple part markings will be inside parentheses. Only one part marking contained in parentheses and separated by a "~" will appear on a part. If a line is indented then it is a continuation of the previous line and the two combined represent the entire part marking for that device.

TAPE AND REEL INFORMATION

REEL DIMENSIONS

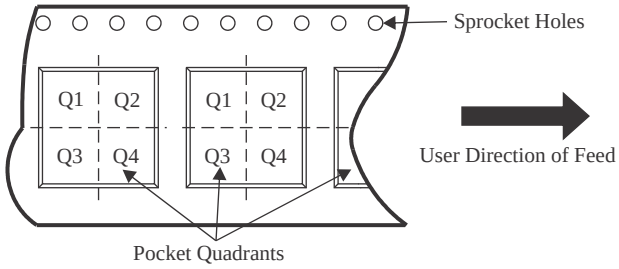


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

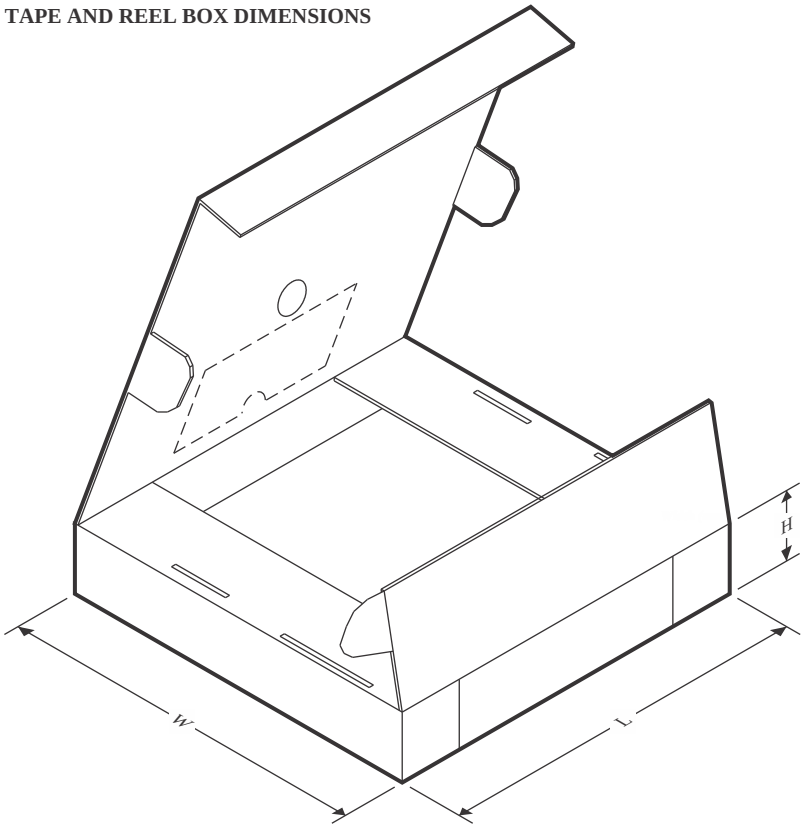
QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
CP3SP33SMSX/NOPB	NFBGA	NZN	224	1000	330.0	24.4	13.4	13.4	2.1	16.0	24.0	Q1

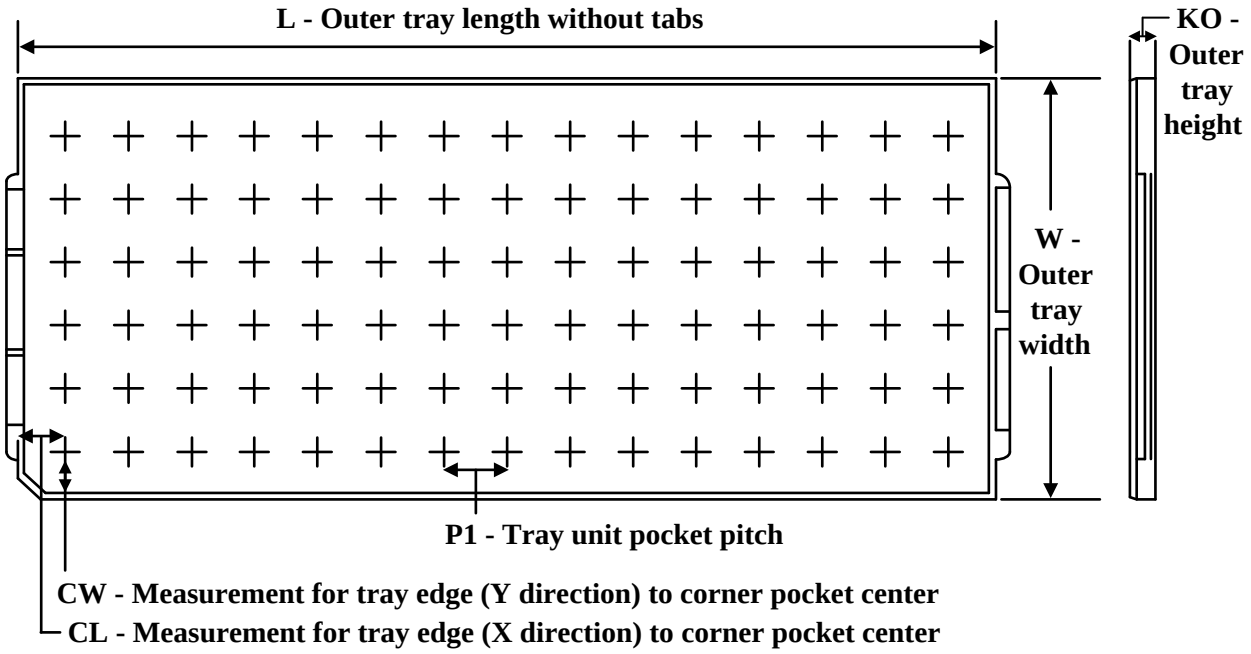
TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
CP3SP33SMSX/NOPB	NFBGA	NZN	224	1000	356.0	356.0	45.0

TRAY

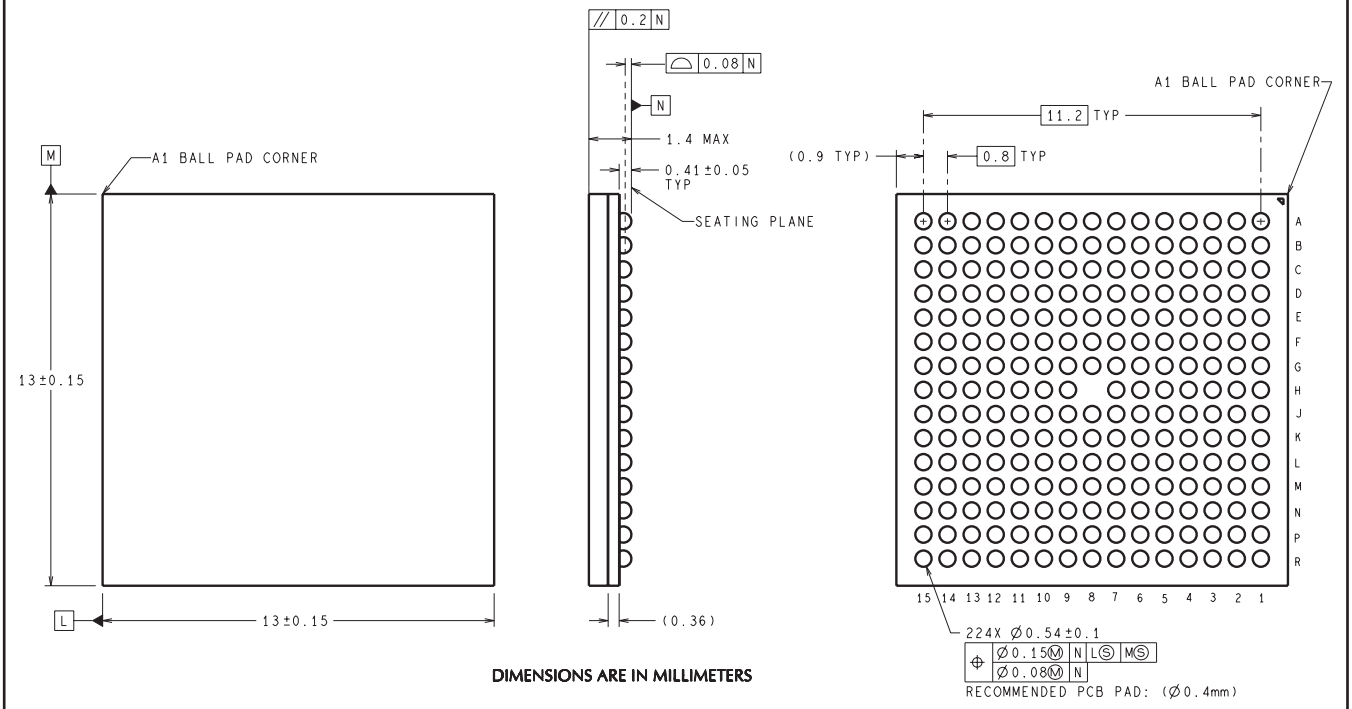


Chamfer on Tray corner indicates Pin 1 orientation of packed units.

*All dimensions are nominal

Device	Package Name	Package Type	Pins	SPQ	Unit array matrix	Max temperature (°C)	L (mm)	W (mm)	K0 (µm)	P1 (mm)	CL (mm)	CW (mm)
CP3SP33SMS/NOPB	NZN	NFBGA	224	160	8 X 20	150	322.6	135.9	7620	15	15	15.45
CP3SP33SMS/NOPB.A	NZN	NFBGA	224	160	8 X 20	150	322.6	135.9	7620	15	15	15.45

NZN0224A



DIMENSIONS ARE IN MILLIMETERS

SLC224A (Rev A)

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale, TI’s General Quality Guidelines, or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Last updated 10/2025