TMS320DSC24 Digital Multimedia Processor

Technical Reference Manual

Version 1

Literature Number: SPRU574
November 2002

TEXAS INSTRUMENTS

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third–party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Mailing Address:

Texas Instruments
Post Office Box 655303
Dallas, Texas 75265

Read This First

About This Manual

This reference guide serves as a reference for the Texas Instruments TMS320DSC24 low-power, enhanced-architecture, dual-core digital signal processor (DSP), and is intended to assist hardware and software engineers in developing applications using this device. It describes both cores of the chip and their coprocessors—the TMS320C54x DSP CPU, the ARM7TDMI microcontroller unit (MCU), the variable length coder, and the imaging extension coprocessor—and peripherals, together with the memory and peripheral interface associated with each core.

Throughout this book, the TMS320DSC24 dual-core device is referred to as the DSC24. Where the ARM7TDMI MCU core is referred to separately, the alphanumeric designation is shortened to ARM. Information about the two-processor cores in relevant chapters is provided separately, rather than combining similar features where applicable.

Notational Conventions

This book uses the following conventions.

- Instruction Sets
  1. The TMS320DSC24 DSP CPU can use either of two forms of the instruction set: a mnemonic form or an algebraic form. This book uses the mnemonic form of the instruction set. For information about the mnemonic form of the instruction set, see *TMS320C54x DSP Reference Set, Volume 2: Mnemonic Instruction Set*. For information about the algebraic form of the instruction set, see *TMS320C54x DSP Reference Set, Volume 3: Algebraic Instruction Set*. These references are both listed in the section titled *Related Documentation From Texas Instruments.*
  2. The TMS320DSC24 MCU CPU uses its own instruction set. For information about the MCU's instruction set, see *TMS470R1x User's Guide*, also shown in *Related Documentation From Texas Instruments.*
- Program listings and program examples are shown in a special typeface.

  Here is a segment of a program listing:

```
langnp1033
STL A,*AR1+ ;Int_RAM(I)=0
RSBX INTM ;Globally enable interrupts
B MAIN_PG ;Return to foreground program
```

- Square brackets, [and], identify an optional parameter. If you use an optional parameter, specify the information within the brackets; do not type the brackets themselves.

Related Documentation From Texas Instruments

The following books provide related documentation for the TMS320DSC24. To obtain a copy of any of these TI documents, call the Texas Instruments Literature Response Center at (800) 477–8924. When ordering, please identify the book by its title and literature number. Many of these documents are located on the Internet at http://www.ti.com.

- **TMS320DSC24 Digital Multimedia Processor** (literature number SPRS195) data sheet contains the electrical and timing specifications for this device, as well as signal descriptions and pin outs for all of the available packages.

- **TMS320C54x DSP Reference Set, Volume 2: Mnemonic Instruction Set** (literature number SPRU172) describes the TMS320C54x digital signal processor mnemonic instructions individually. Also includes a summary of instruction set classes and cycles.
- **TMS320C54x DSP Reference Set, Volume 3: Algebraic Instruction Set** (literature number SPRU179) describes the TMS320C54x digital signal processor algebraic instructions individually. Also includes a summary of instruction set classes and cycles.
- **TMS470R1x User's Guide** (literature number SPNU134) describes the TMS470R1x RISC microcontroller, its architecture (including registers), the ICEBreaker module, interfaces (memory, coprocessor, and debugger), 16–bit and 32–bit instruction sets, and electrical specifications.
- **TMS320C54x DSKplus User's Guide** (literature number SPRU191) describes the TMS320C54x digital signal processor starter kit (DSK), which allows you to execute custom C54x. code in real time and debug it line by line. Covered are installation procedures, a description of the debugger and the assembler, customized applications, and initialization routines.
- **TMS320C54x Assembly Language Tools User's Guide** (literature number SPRU102) describes the assembly language tools (assembler, linker, and other tools used to develop assembly language code), assembler directives, macros, common object file format, and symbolic debugging directives for the C54x generation of devices.
- **TMS320C54x C Source Debugger User's Guide** (literature number SPRU099) tells you how to invoke the C54x emulator, evaluation module, and simulator versions of the C source debugger interface. This book discusses various aspects of the debugger interface, including window management, command entry, code execution, data management, and breakpoints. It also includes a tutorial that introduces basic debugger functionality.
- **TMS320C54x Code Generation Tools Getting Started Guide** (literature number SPRU147) describes how to install the TMS320C54x assembly language tools and the C compiler for the C54x devices. The installation for MS–DOS, OS/2, SunOS., Solaris, and HP–UX. 9.0x systems is covered.
- **TMS320C54x Evaluation Module Technical Reference** (literature number SPRU135) describes the C54x evaluation module, its features, design details, and external interfaces.
- ***TMS320C54x Optimizing C/C++ Compiler User's Guide*** (literature number SPRU103) describes the C54x C compiler. This C compiler accepts ANSI standard C source code and produces TMS320. assembly language source code for the C54x generation of devices.
- ***TMS320C54x Simulator Getting Started Guide*** (literature number SPRU137) describes how to install the TMS320C54x simulator and the C source debugger for the C54x. The installation for MS–DOS, PCDOS, SunOS, Solaris, and HP–UX systems is covered.
- ***TMS320 Third–Party Support Reference Guide*** (literature number SPRU052) alphabetically lists over 100 third parties that provide various products that serve the family of TMS320 digital signal processors. A myriad of products and applications are offered—software and hardware development tools, speech recognition, image processing, noise cancellation, modems, etc.
- ***TMS320 DSP Development Support Reference Guide*** (literature number SPRU011) describes the TMS320_ DSP family of digital signal processors and the tools that support these devices. Included are code–generation tools (compilers, assemblers, linkers, etc.) and system integration and debug tools (simulators, emulators, evaluation modules, etc.). Also covered are available documentation, seminars, the university program, and factory repair and exchange.

Trademarks

- TMS320C54x, C54x, TMS320, and MicroStar BGA are trademarks of Texas Instruments.
- ARM7TDMI, ARM, Thumb, and Multi–ICE are registered trademarks of ARM Limited.
- ICEBreaker, ARM7, and Embedded ICE are trademarks of ARM Limited.
- MS–DOS is a registered trademark of Microsoft Corporation.
- OS/2 and PC-DOS are trademarks of International Business Machines Corporation.
- SunOS and Solaris are trademarks of Sun Microsystems, Inc.
- HP-UX is a trademark of Hewlett-Packard Company.
- Other trademarks are the property of their respective owners.

How to Use This Manual

The following table summarizes the DSC24 and information contained in this manual.

If you are looking for information about:	Turn to these chapters:
Processing data flows	Chapter 1.3
System architecture	Chapter 2
MCU subsystem	Chapter 3
Power saving	Chapters 3.7 and 6.7
Imaging peripherals	Chapter 4
On-screen display	Chapter 4.5
External memory interface	Chapter 5
DSP subsystem	Chapter 6
iMX coprocessor	Chapter 6.5
VLC coprocessor	Chapter 6.6
ARM-DSP communication	Chapter 7
Block diagram	Appendix A
Memory map	Appendix B

Information About Cautions and Warnings

This book may contain cautions and warnings.

This is an example of a caution statement.

A caution statement describes a situation that could potentially damage your software or equipment.

This is an example of a warning statement.

A warning statement describes a situation that could potentially cause harm to you.

The information in a caution or a warning is provided for your protection. Read each caution and warning carefully.

Contents

List of Figures

Figure *Page*

List of Tables

Table *Page*

1 Introduction

This chapter introduces the TMS320DSC24 (DSC24)—a multicore device consisting of a programmable digital signal processor (DSP) and a microcontroller unit (MCU).

This introduction includes a general description of the chip and a list of its key features.

For specific information about the 54x DSP core or the RISC MCU core, see the appropriate user guide listed under *Related Documentation from Texas Instruments* in the preface of this manual.

1.1 Description

The DSC24 chip is designed to implement the digital processing of imaging, video, and audio data. The DSC24 integrates an ARM7 RISC microcontroller core with emulation facilities, a TMS320C54x DSP subsystem with its own program and data memories, and two DSP coprocessors to speed up imaging functions. The device also includes peripherals designed to interface the chip with various processors, peripherals, and memories. A simplified block diagram is shown in Figure 1–1.

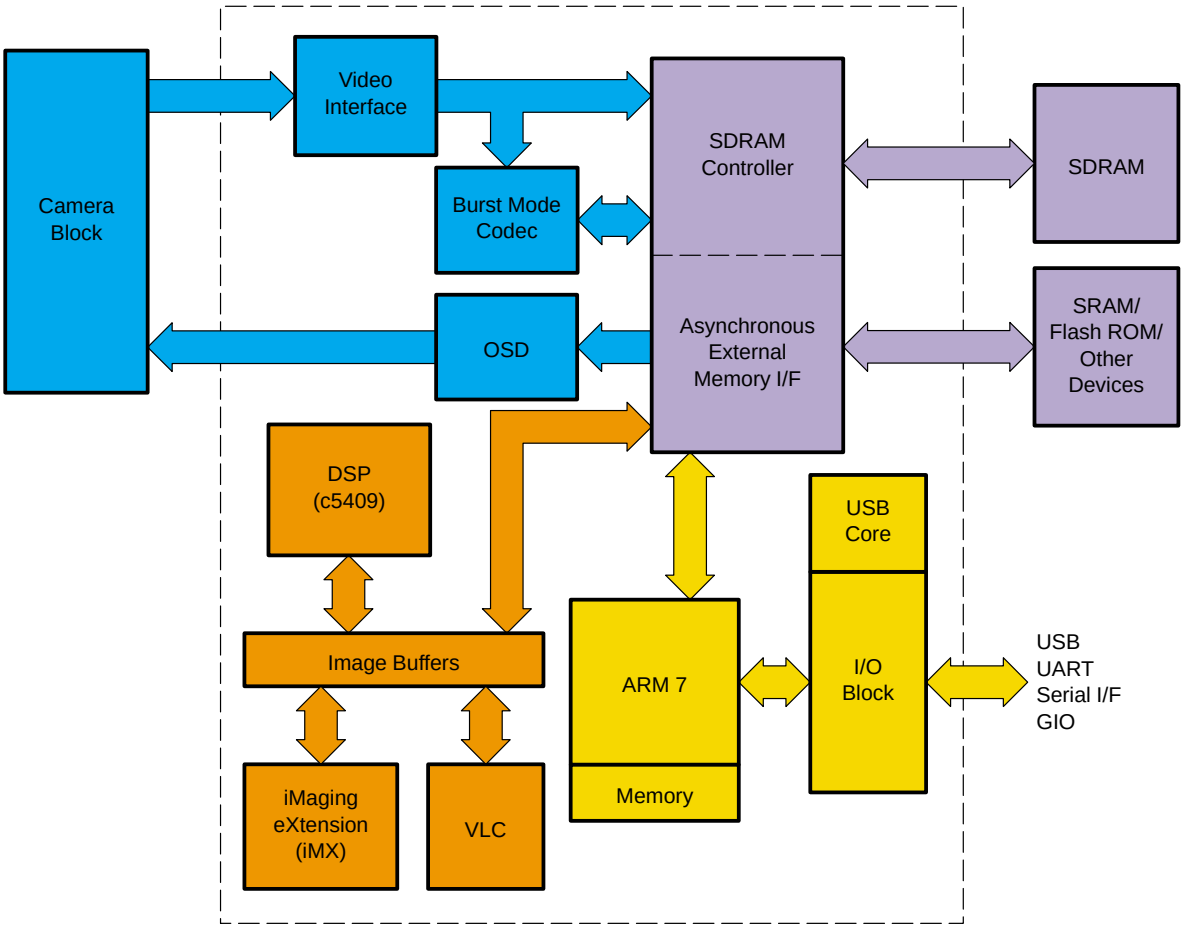


Figure 1–1. TMS320DSC24 Block Diagram

The different blocks that make up the DSC24 are grouped into five entities:

- ARM subsystem and its peripherals
- DSP subsystem and its own memory
- Two coprocessors and their own memory
- Memory controller block
- Imaging peripheral block

The features of each of these blocks are given in the following sections.

1.2 Features

The five blocks make the DSC24 a powerful low-power chip.

- The ARM subsystem is built around an ARM7 32-bit RISC processor. This subsystem block features a USB controller, general-purpose I/O pins, up to three serial ports, two UART ports, four timers, and a watchdog timer. The ARM subsystem controls the different system clocks and reduces the power of the whole system. An external microcontroller requests the ARM bus and controls the DSC system.
- The DSP subsystem and the coprocessors subsystem are made up of a ‘C5409 DSP core from Texas Instruments, an imaging extension coprocessor (iMX) to compute heavy imaging algorithms, and a variable length coding coprocessor (VLC) to speed up Huffman coding. The DSP subsystem has its own memory. The DSP also features two multichannel buffered serial ports (McBSP) that can be used to input and output audio data.
- The external memory interface (EMIF) block is made up of an SDRAM controller and an asynchronous external memory interface (AMIF).
- The imaging peripheral block includes a video interface to input and output video data, a burst compression expansion block to compress image data on the fly, and an on screen display (OSD) module that allows the user to superpose real images with text or graphics.

The DSC24 chip offers four different modes of operations. These modes of operations are called mode 0, mode 1, mode 2, and mode 3. Some features may or may not be available depending on the mode used. The MODE_SEL1 and MODE_SEL0 pins select the device-operating mode.

1.2.1 Features of the ARM Subsystem Block

1.2.1.1 ARM7 32-Bit RISC Processor

The ARM7, which is a 32-bit RISC processor, is built into the DSC24 as a system controller. The ARM7 CPU performs general system control such as system initialization, configuration, user interface, and user command implementation.

The embedded ARM processor features are:

- 32-bit RISC processor
- 32K-byte built-in RAM
- 32-/16-bit command sets
- Memory format: little endian format fixed
- Two interrupt systems (FIQ, IRQ)
- Eight external terminal interrupt systems
- Four hardware timers
- Watchdog timer
- JTAG base emulation
- External memory interface

1.2.1.2 USB Controller

The USB controller built in the DSC24 conforms to the USB specification revision 1.1. The USB controller corresponds to full-speed devices and works in slave mode. The USB has a built-in transceiver, its own 2K bytes of RAM for FIFO, a programmable packet size, and supports remote wake-up.

1.2.1.3 UART Ports

The DSC24 supports two-channel start-stop synchronization serial ports (UART). Among the two channels, one channel (UART0) supports a full set of UART communications. A 32-byte FIFO is built in for each transmitting side and receiving side and interrupts can be generated due to errors or various states of the FIFO. Data is 8 or 7 bits. Even, odd, or no parity bit can be selected and 1- or 2-stop bits can be selected. Transmission/reception timing uses the ARM clock, and any bit rate can be set in integral multiples of 1/16 of the ARM clock.

1.2.1.4 Serial Ports

The DSC24 has built-in three clock synchronized serial interfaces. Two of them are always available; the third serial port is available only in device mode 2. The transfer unit is 8 bits each time, is a simple three-line serial interface consisting of a clock line, output data line, and an input data line. It has a DMA transfer function with SDRAM for transfer of large quantities of data. Transmission is in 8-bit units and has 8-bit buffers on each of the transmitting and receiving sides. The LSB or MSB is first selected for transmitted/received data. An interrupt to the ARM is generated when an 8-bit transmission (reception) is complete. The clock polarity when there is no transfer is selectable.

1.2.1.5 General-Purpose Input Output Pins

The DSC24 has 32 general-purpose input/output pins (GIO) available when used in operating mode 3 and 21 GIO pins when used in any other modes. Using the GIO pins, the logic levels of the ports of an externally assigned device are read and controlled. These pins are often multiplexed with other functions.

1.2.1.6 External CPU Interface

By setting the EXTCPU pin to high, access to the DSC24 internal registers and memory areas by an external processor is possible. This mode is mostly used if the user wants to virtually replace the embedded ARM7 with another microcontroller or microprocessor. In this mode, the internal ARM is disconnected from its bus and cannot be used anymore.

Dynamic wait control is not performed by the interface with the external host controller unless the device-operating mode 2 is used. The number of *waits* of the external host controller must be set according to the maximum access time.

1.2.2 Features of the DSP Subsystem Block

1.2.2.1 16-Bit Fixed Decimal Point Digital Signal Processor (TMS320C5409)

The DSP subsystem is constructed around the TMS320C5409 from the Texas Instruments catalog DSP. The DSP subsystem also includes 32K words of expansion memory, an imaging extension coprocessor (iMX), and a variable length coding coprocessor (VLC). Both are used as image processing accelerators. The DSP subsystem also features a portion of shared memory called the image buffer. This memory is used to transfer data between different elements of the device.

The TI C54x core is a high-performance, low-power DSP core. The DSP software development tools for the C54x are mature and fully compatible with the DSC24 DSP subsystem.

To realize image processing, image compression processing, and voice processing, the DSP controls SDRAM, DMA transfer to/from external memory, and iMX/VLC accelerators.

The embedded ARM processor features are:

- 16-bit fixed decimal point digital signal processor (based on the ‘C5409)
- Maximum operating frequency of 94.5 MHz (see the data sheet, literature number SPRS195 for the actual value)
- 32K x 16-bit internal RAM + 32K x 16-bit expanded RAM (total 64K x 16 bits)
- Two multiplex channel buffered serial ports (McBSP)
- The ARM accesses DSP internal RAM via an expansion 16-bit host port interface (HPI)
- JTAG base emulation

1.2.2.2 DSP Hardware Accelerators (iMX/VLC)

The iMX is a parallel MAC engine for expanding DSP image processing performance. It has a flexible control, a memory interface, and four MACs. The iMX is effective for fast computation of image processing in regular block units.

As a general-purpose image processor, the types of image processing given below are possible by the iMX.

- First-order/second-order filter
- CFA interpolation filter
- Color space conversion
- Color signal down sampling
- Edge intensification
- Color signal component suppression
- DCT (discrete cosine transfer) and IDCT (inverse discrete cosine transfer)
- Table lookup

This list is not exhaustive. Many more functions can benefit from the iMX engine.

The VLC accelerator is a coprocessor that is optimized for quantization and Huffman encoding within the framework of JPEG compression and MPEG compression. This engine operates both the Huffman tables and quantization matrices loaded in advance by DSP. By pipelining in the design, high processing capability of other 30,000,000 DCT coefficients per second for compression can be obtained.

The VLC has the following functions:

- Quantization, zigzag scanning, and Huffman encoding for JPEG encoding
- Quantization, zigzag scanning, and Huffman encoding for MPEG–1 video encoding

The accelerator requires memory blocks for the input/output buffer, quantization matrices, and Huffman encoding tables. The memory structure supports ordinary encoding processes, JPEG minimum coding unit (MCU), and MPEG–1 macro blocks.

1.2.3 Features of the Imaging Peripheral Block

1.2.3.1 Video Interface Block

The video interface (I/F) module loads YCbCr or CCD/CMOS RAW data from an external camera module to the SDRAM or external memory. This module supplies HD/VD to the external timing generator or synchronizes to the HD/VD provided from the external timing generator. Other functions of this block are digital clamping and decimation.

- Generates timing signal VD/HD
- Can synchronize by externally generated HD/VD signals
- Supports sequential scanning and interlace scanning
- Automatic optical black clamping
- Programmable image load position and region setting
- Programmable horizontal/vertical culling function
- Supports maximum input pixel block of 40 MHz
- Supports maximum input image device size of 4096 x 4096
- Input format: YCbCr (16/12/8 bits), CCD RAW data (10 bits)
- Corresponds to 4:2:2/4:1:1 YCbCr format
- Y data offset value delete circuit

1.2.3.2 Burst Mode Compression/Expansion

When processing pictures in a continuous mode, the image processing time (image generation + JPEG) of the loaded image does not overtake the input data. Thus, there is a method in which the CCD/CMOS RAW data is stored in SDRAM and then the image processing is performed. However, with this method, the number of continuous photo images that can be taken at one time is restricted because the capacity of the SDRAM is limited. In the DSC24, built-in hardware compresses input *raw* data in real time by lossless compression and stores it in SDRAM.

- Input data: CCD/CMOS raw data
- Reversible compression by DPCM + Huffman encoding
- Synchronization by HD/VD signal is possible
- Supports sequential scanning and interlace scanning
- Programmable image loading position and region size
- Maximum pixel clock of 40 MHz
- Partial image expansion process is possible
- Simultaneous processing of image compression and expansion processes is possible
- Average compression ratio = 65%

1.2.3.3 On-Screen Display (OSD)

The on-screen display module of the DSC24 manages the video and bitmap data. This block superposes the real-time video information with some bitmap information stored within the SDRAM to create the final picture. This module has the following features.

- Two video display windows
- Two OSD display windows
- One cursor display
- Background color of each window can be specified
- Video-Window0 and Video-Window1 are displayed by YCbCr and OSD-Window0 and OSD-Window1 are displayed by bitmap
- 1-, 2-, 4-, or 8-bit width can be selected for bitmap
- Video-Window0 / Video-Window1 and OSD-Window0 / OSD-Window1 can be blended and displayed
- Five blending ratios can be selected
- Programmable bitmap color look-up table (24 bits x 256 words in RAM) and fixed CLUT (ROM) are available
- Interlaced and non-interlaced modes are supported
- VGA to NTSC/PAL conversion circuit built in (supports non-square pixel)

1.2.4 Features of the External Memory Interface Block

1.2.4.1 SDRAM Controller

The SDRAM controller block acts as the primary interface between SDRAM and all functional blocks such as the processors (ARM, DSP), video interface block, OSD, and DMA controller. This block supports SDRAM timing of a maximum of 80 MHz and provides continuous data access with low overhead.

- Supports 16/64/128/256/512-Mbit SDRAM
- Supports both 32-bit SDRAM or 16-bit SDRAM
- Maximum 80-MHz operation (see the data sheet, literature number SPRS195 for the actual value)
- Access in word, half word, or byte units (ARM)
- Mode setting, power down, and self refresh commands
- Programmable refresh interval
- Column access strobe (CAS) latency 1, 2, or 3 can be selected

- SDRAM size of up to 64M bytes can be used
- DMA transfer with peripheral modules
- SDRAM access priority is variable

1.2.4.2 Asynchronous External Memory Interface

The asynchronous external memory interface (AMIF) is constructed from an interface circuit to the external memory, a chip select output terminal, its control circuit, and a DMA controller. The AMIF accesses up to a maximum of 32M bytes (in case of data width 32 bits).

- Bus width of CS0 of 8 bits/16 bits is selected by the BUSWIDTH input.
- Data width of 8, 16, or 32 bits can be selected for each CS regions (CS1–4).
- If a 32-bit width is selected, the upper 16 bits of the data bus of SDRAM IF are used. In this case, SDRAM has 16-bit width.
- Write is possible in byte units (CS0–4).
- Using BUS_REQ and BUS_ACK signals, handshakes of open request and acknowledgment of bus rights with the external controller is possible.
- IDLE can be inserted in write after the read operation and in access to other CS regions.
- After reset is canceled, the ARM jumps to address 0x0 of CS0.

1.3 Examples of Data Flows

This section shows examples of data flow for processing. The user is not requested to follow these flows step by step. The information here explains how the DSC24 handles the data in an imaging and video system. Depending on the application and requirements, the software may use a different flow.

1.3.1 Image Compression Process Flow

An external frame buffer (SDRAM) is required for the DSC24 image compression process. In encoding of still images (JPEG), one frame's worth is used as *work* memory of the input image. In encoding of moving images (MPEG4), three frame's worth is used as *work* memory of the input image. Figure 1–2 shows the signal flow during the image compression process.

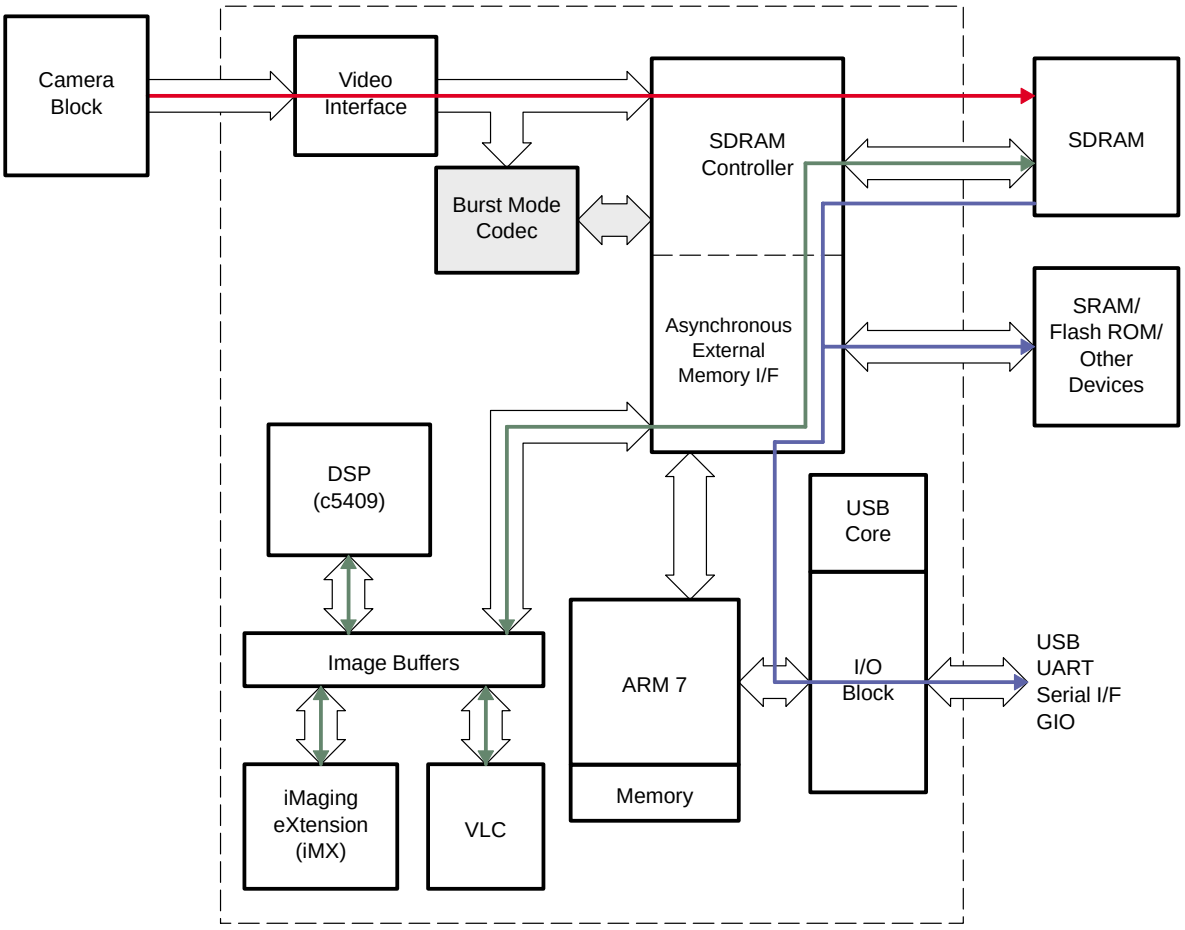


Figure 1–2. Still Image/Moving Image Compression Process

The process flow is as follows:

1. The ARM sets the size and position from HD/VD of the image loaded into the video interface block, then the video interface block writes Y/Cb/Cr/CCD RAW data into SDRAM.
2. The ARM issues an encoding start command to the DSP.
3. The DSP loads the image data into the image buffer and performs the image compression process using the iMX/VLC engine. The generated bit stream data is returned and written from the image buffer to SDRAM. This is repeated, and when a compression of one image is finished, an interrupt is generated to the ARM and it waits for the next command from the ARM.
4. The ARM transfers the generated bit stream data to an external device using the external memory I/F or communications port.

1.3.2 Image Expansion Process Flow

The JPEG/MPEG4 data in external memory is expanded and data for display is transferred to the LCD controller on the external memory I/F. In this case, the SDRAM or SRAM on the asynchronous external memory I/F can be used as *work* memory. Figure 1–3 shows the signal flow during the image expansion process.

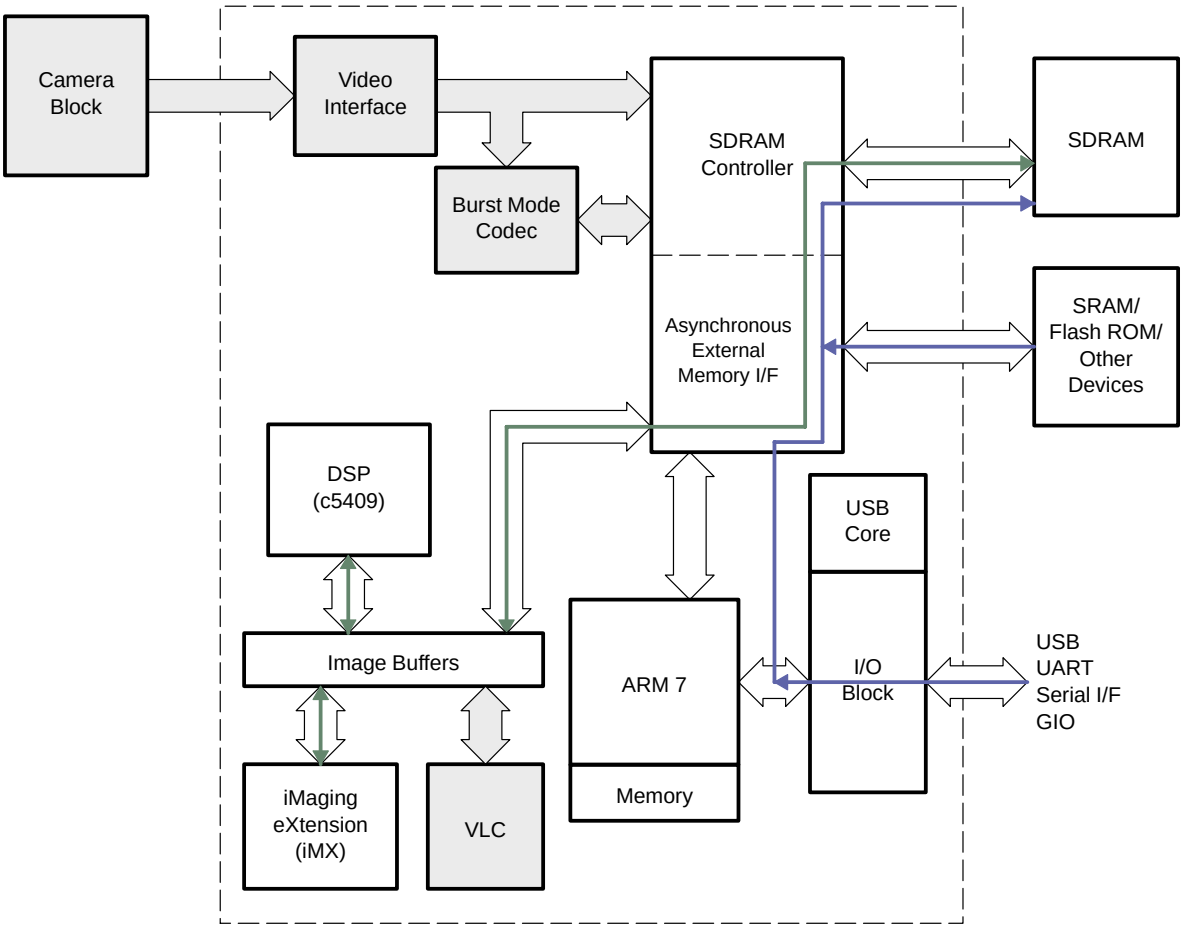


Figure 1–3. Still Image/Moving Image Expansion/Playback Process

The process flow is as follows:

1. The ARM loads the compressed bit stream data into external memory using the external memory I/F or communications port.
2. The ARM issues a decoding start command to DSP.
3. The DSP loads image data into the image buffer and performs the image expansion process using the iMX engine. The produced image data (RGB) is returned from the image buffer and written into SDRAM. Or, data for display is transferred directly in the LCD controller on the external memory I/F. This is repeated, and when expansion of one image is finished, interrupt is generated to the ARM and awaits the next command from the ARM.

1.3.3 Audio Voice Compression/Expansion Flow

The JPEG/MPEG4 data in external memory is expanded and data for the display is transferred to the LCD controller on the external memory I/F. In this case, SDRAM or SRAM on the asynchronous external memory I/F can be used as *work* memory. Figure 1–4 shows the process flow for the audio and voice data playback.

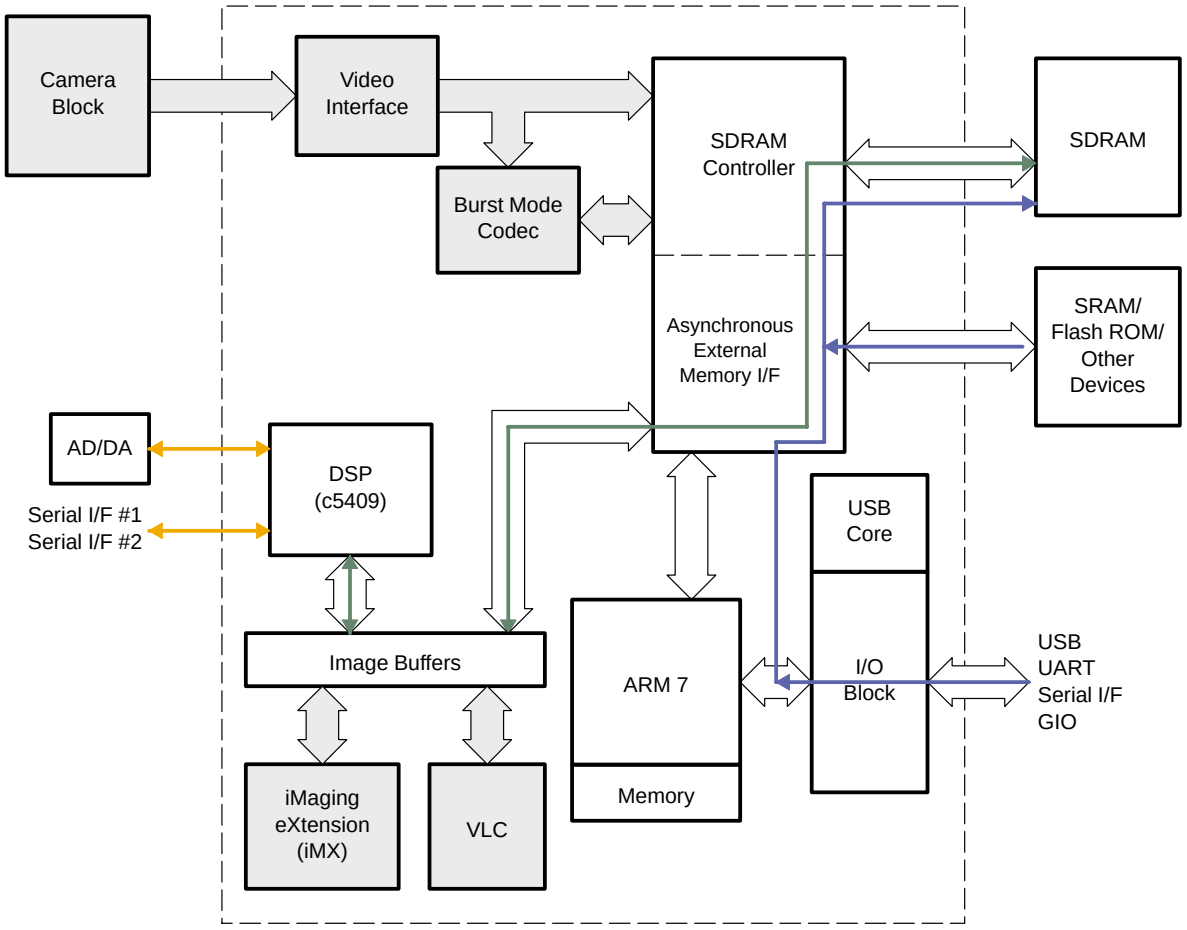


Figure 1–4. Audio/Voice Compression/Expansion Process

The process flow for the audio and voice data playback is as follows:

1. The ARM loads compressed bit stream data into external memory using the external memory I/F or communications port.
2. The ARM issues a playback start command to the DSP.
3. The DSP loads audio/voice data into the image buffer, and performs the audio/voice expansion process. The produced audio/voice data is transferred from the serial I/F connected to the DSP to an external DAC and played back. Since the DSP has two multichannel serial I/F, compressed data can be loaded and played back directly from the serial I/F.

The process flow for the audio and voice data compression is as follows:

4. The ARM issues a compression start command to DSP.
5. The DSP loads audio/voice data from an external ADC and performs the audio/voice compression process. The produced compressed data is transferred to external SDRAM or external SRAM via the image buffer.
6. The ARM transfers the compressed bit stream data to external memory using the external memory I/F or communications port.

1.3.4 Burst Image Loading Flow

When a camera is in continuous shot mode, RAW data from the CCD/CMOS sensor undergoes lossless compression as is without being loaded into external SDRAM and the image is loaded in real time to SDRAM. By using the burst mode codec engine, the memory size used by SDRAM is reduced. Figure 1–5 shows the process flow for the burst compression.

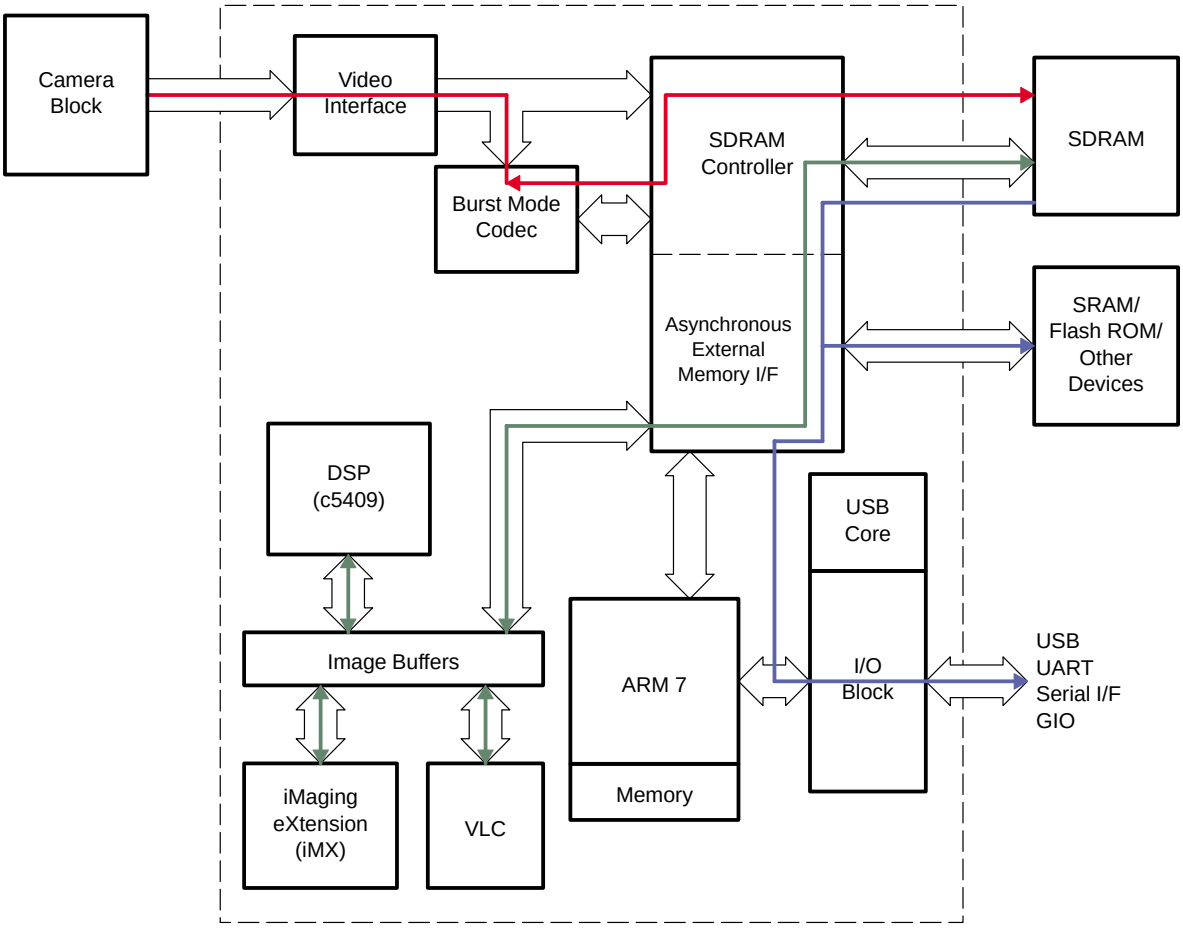


Figure 1–5. Still Image Compression Process (Burst Mode)

The process flow for the burst compression is as follows:

1. The ARM activates the video interface block and starts to load the CCD/CMOS sensor RAW data.
2. The ARM sets the size and position from HD/VD of the image loaded into the burst mode codec block. The burst mode codec block performs the lossless compression process of the CCD/CMOS sensor RAW data and then writes into SDRAM.
3. The ARM specifies the SDRAM address where the decompressed image is stored to the burst mode codec block. The burst mode codec block decompresses the specified image, writes it to SDRAM, and awaits the next command from the ARM. At this time, the image compression process and decompression process operate in parallel.
4. The ARM issues an encoding start command to the DSP.
5. The DSP loads the image data into the image buffer and performs the image compression process using the iMX/VLC engine. The generated bit stream data is returned from the image buffer and written into SDRAM. This is repeated, and when a compression of one image is finished, an interrupt is generated to the ARM, and the DSP awaits the next command from ARM.
6. The ARM transfers the generated bit stream data to an external device using the external memory I/F or communications port.

1.3.5 System Configuration When SDRAM Is Not Used

The DSC24 also supports a system configuration that does not use SDRAM. In this case, a general-purpose SRAM and so forth on the asynchronous external memory I/F is used as the frame buffer. The process flow is almost the same as the case where an external SDRAM is used, but the image size that can be handled and the processing time are restricted.

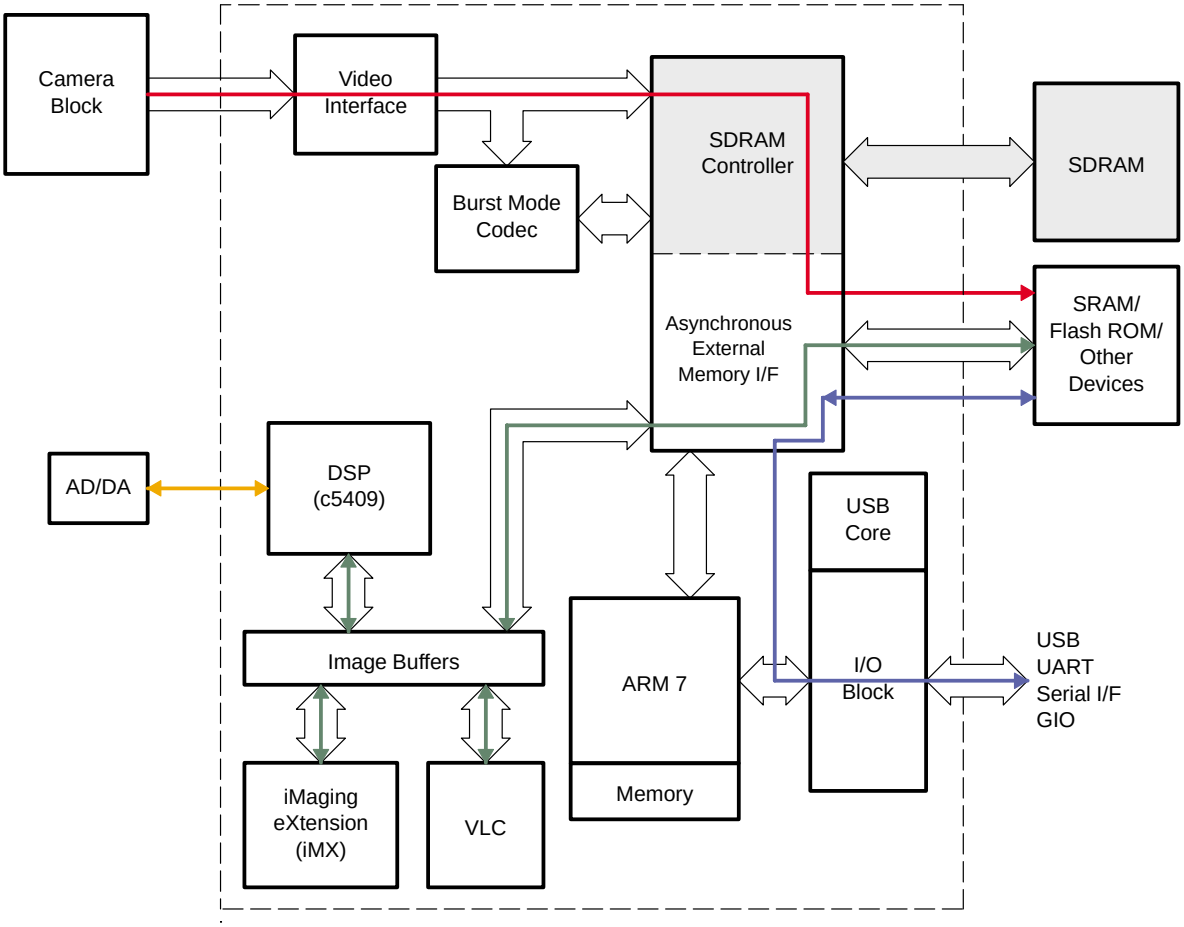


Figure 1–6. System That Does Not Use SDRAM

1.3.6 System Configuration Using External Host CPU

The DSC24 supports a system configuration in which the DSC24 is operated by an external host CPU. In this mode the internal ARM processor does not operate.

Since the interface with the external CPU is performed asynchronously, the speed of the internal CPU bus is slower than the case where the internal ARM CPU is used. The external CPU performs image/voice data compression/ expansion processing by writing image/voice data into SDRAM, or by writing data directly into the internal memory of DSC.

The following operations are possible in this mode.

- Reading/writing of registers of each module by the external CPU
- Reading/writing to external SDRAM by the external CPU
- Reading/writing to DSP internal RAM (32K word) by external CPU
- DMA transfer between SDRAM and the video interface block
- DMA transfer between SDRAM and the burst mode codec block
- DMA transfer between SDRAM and the image buffer block

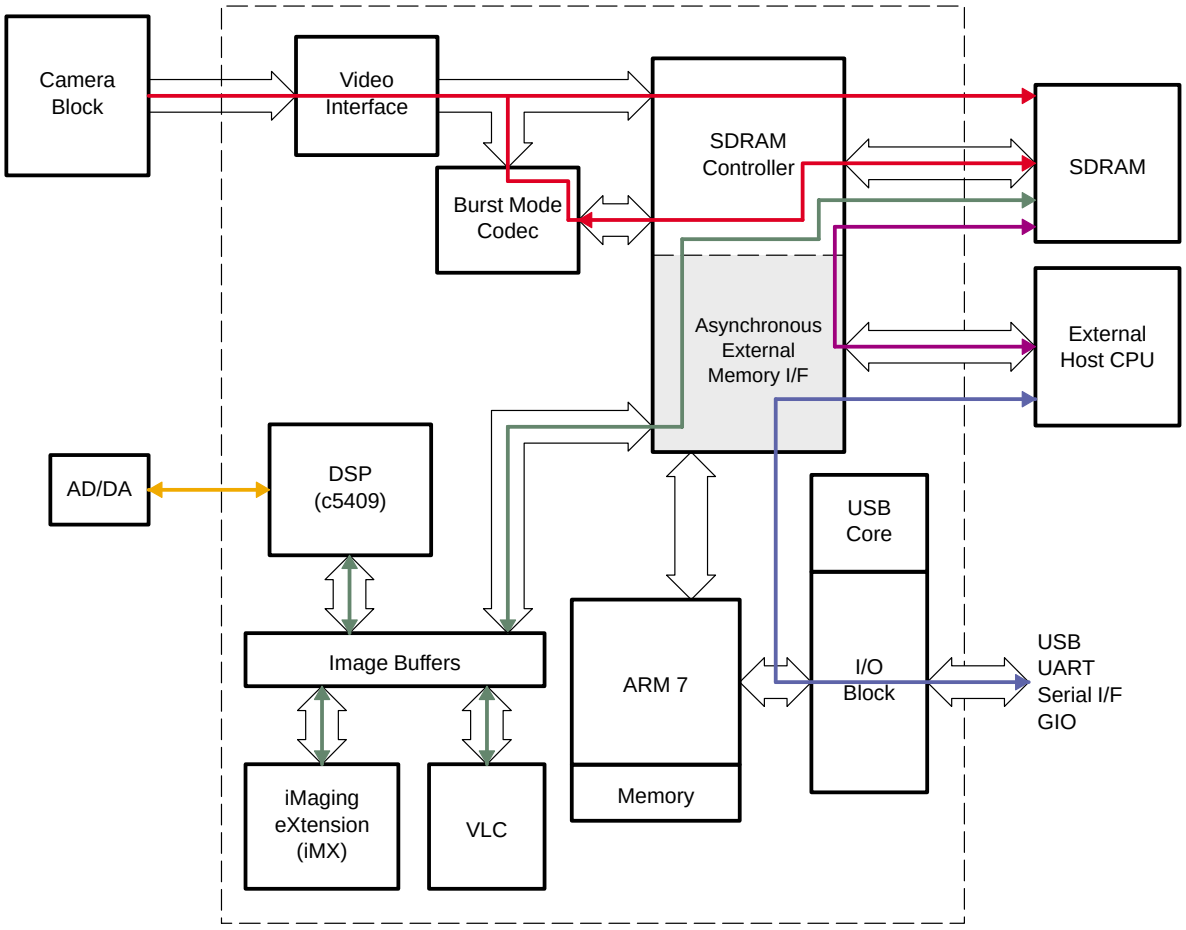


Figure 1–7. System Configuration That Uses an External Host CPU

2 DSC24 System Architecture

This chapter introduces the different blocks that make the TMS320DSC24 chip.

2.1 Introduction

This overview introduces the basic architecture of the TMS320DCS24 (DSC24). Only the initial information on the DSC24 is covered here. More information on the DSC24 is covered later in this document.

2.2 What is the TMS320DSC24?

The DSC24 is a high-performance single-chip processor that combines the following architectures:

- A DSP (TMS320C5409)
- A microcontroller (ARM7TDMI)
- Two DSP coprocessor engines [imaging extension (iMX) and a variable length decoder (VLC)]
- An external memory interface (SDRAM controller + asynchronous external memory interface)
- Imaging peripherals

Because of this high level of integration, the DSC24 is a perfect fit for demanding applications in video, audio, and imaging products.

Each of the five DSC24 subsystems is good at specific tasks that fit together to make a total system on a chip. The on-chip microcontroller runs the user operating system, handles complex protocols, performs control functions, and bit manipulations. In parallel with this, the DSP and coprocessors perform mathematically intensive algorithms, digital signal processing functions, and mathematical manipulations. The imaging peripherals and memory controller are maximized for throughput. The result is the DSC24, a high-performance multiprocessing imaging system on a single-chip.

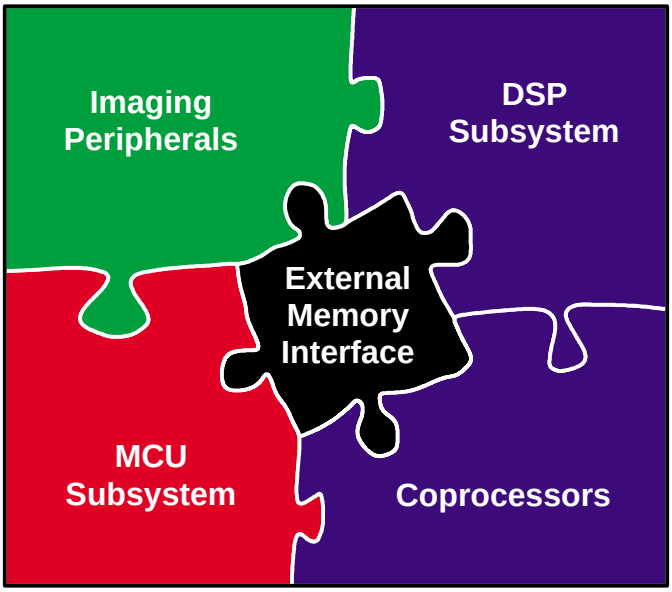


Figure 2–1. Five Blocks of the DSC24 System

2.3 MCU Subsystem Overview

The MCU subsystem is made up of the ARM7TMDI™ RISC processor core, static RAM memory, and peripherals. The MCU subsystem performs all the system control functions and is capable of supporting operating systems (OS). The ARM also controls the various peripheral modules including the timers, interrupt controller, burst compression engine, on-screen display, video interface, USB, and serial ports. The user interface software also runs on this processor.

The MCU is responsible for handling many system functions such as system-level initialization, configuration, user interface, user command execution, connectivity functions, and overall system control. The general-purpose ARM handles these functions because it has a larger program memory space and better context switching capability, and is thus more suitable for complex, multitasking, and general-purpose control tasks compared to the on-chip DSP.

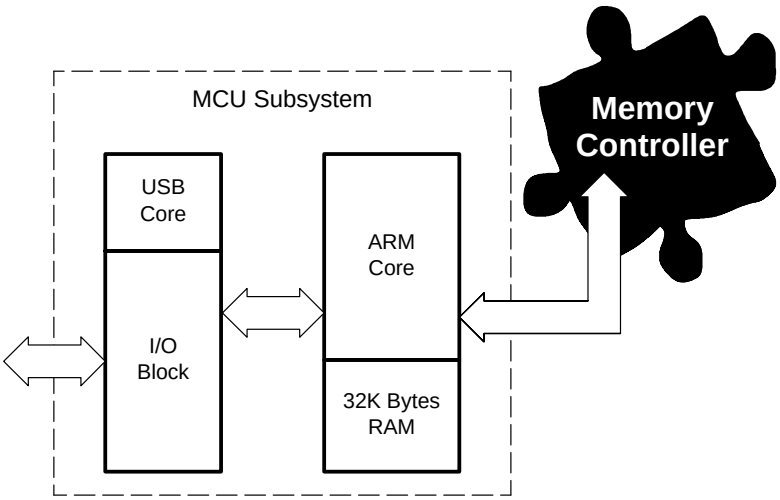


Figure 2–2. MCU Subsystem

The MCU and the DSP are loosely coupled processors and communicate through a dedicated interrupt driven based protocol using the DSP's host port interface (HPI). This allows the MCU to read/write directly to the DSP's internal memory for command and/or data transactions. Both the MCU and DSP have access to SDRAM to efficiently share large blocks of image data.

In addition to these features, an external host mode is available. Pulling the EXTCPU pin high activates this mode. When in external CPU mode, the internal ARM of the DSC24 is disconnected from its bus and an external device can take control of the chip. The external device gets access to the peripheral registers, the internal DSP RAM, as well as the external SDRAM.

2.4 DSP Subsystem and Coprocessors Overview

The DSP subsystem is based on the TMS320C5409 DSP from Texas Instruments, the DSP's local memory blocks, and a special memory block called image buffers. The coprocessor subsystem is made of two DSP coprocessors, the image extension (iMX) coprocessor, and the variable length coder (VLC) coprocessor.

The DSP subsystem performs all the processing of the image data. The data is fetched from the SDRAM into an image buffer by the DSP through SDRAM controller requests and the DSP performs all the image processing and compression required. The DSP activates the iMX or the VLC coprocessors to accelerate dedicated imaging functions and increase the performance of the DSP for imaging applications.

The iMX consists of four multiply accumulate (MAC) units in parallel and is particularly useful for matrix multiplication's. The VLC is optimized to perform quantization and Huffman encoding for JPEG or MPEG-1 compression. The DSP calls subroutines to operate the iMX and VLC.

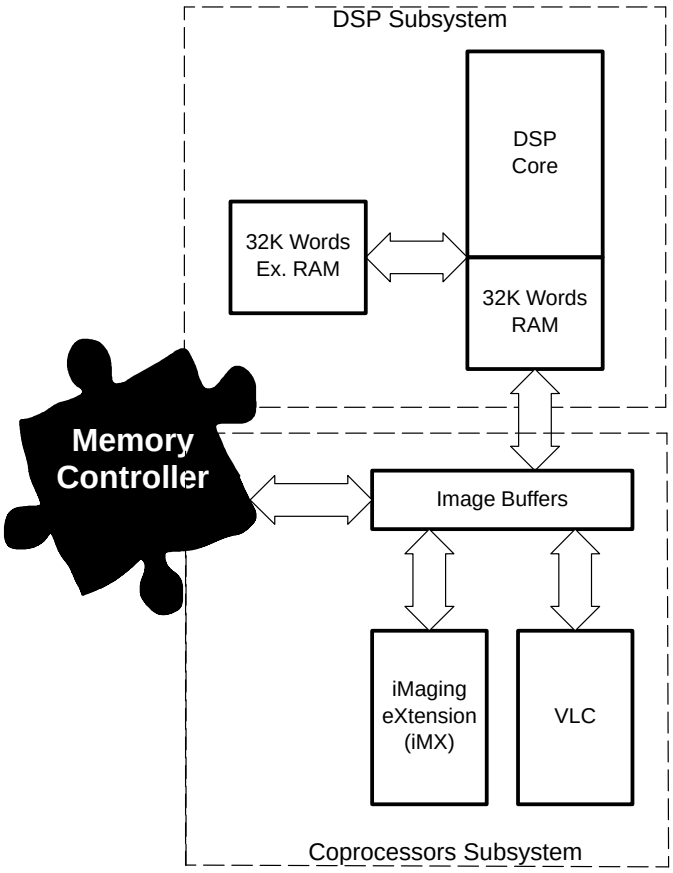


Figure 2–3. DSP and Coprocessors Subsystems

The DSP subsystem executes all computationally expensive signal- processing tasks such as image pipelining, JPEG, and image stitching. The DSP subsystem also handles all real-time I/O such as audio and modem I/O.

2.5 Imaging Peripherals Overview

There are three imaging peripherals included on the DSC24. They are the video interface, the burst mode compression/decompression unit, and the hardware graphics unit. These specialized imaging peripherals are used to read image data into the DSC24 system and create windowed, color display data for user interaction.

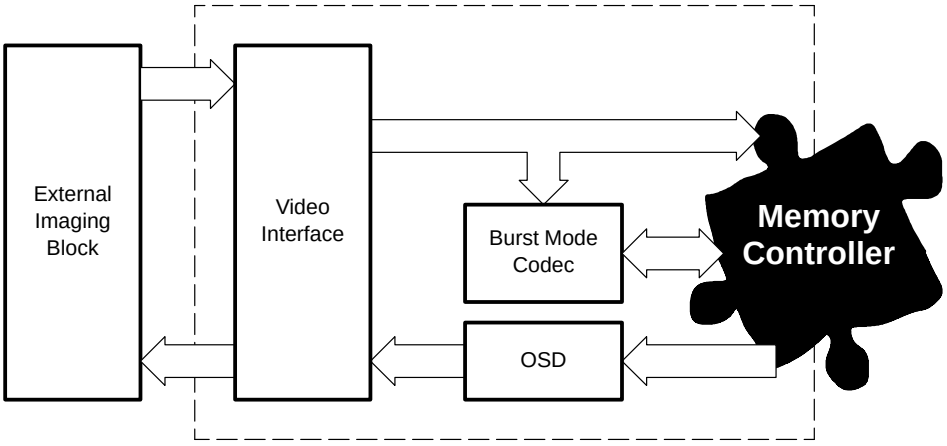


Figure 2–4. Imaging Peripherals

The video interface provides the necessary logic to interface to an external video encoder/decoder. The format of the video input can either be a YCrCb 4:2:2 format on an 8- or 16-bit bus or a YCrCb 4:1:1 format on a 12-bit data bus. The video output formats supported by the DSC24 are YCrCb 4:2:2 on a 16-bit bus and YCrCb 4:1:1 over a 12-bit bus. The video interface can also be connected to a 10-bit CCD or CMOS sensor.

The burst mode compression/decompression block compresses the raw image data from the video interface using a lossless algorithm (or lossy as selected by the user) and writes the compressed data to the SDRAM. The decompression engine then decompresses this data. The data is then processed and displayed or stored back to the SDRAM.

The on-screen display unit or OSD supports up to five hardware windows, one of which is a cursor, two others video data, and the last two are user-defined bitmaps that can be used to display a menu on the screen for example.

The ARM controls these imaging peripherals.

2.6 External Memory Interface Overview

The external memory interface (EMIF) is made of sub blocks: the asynchronous external memory interface or AMIF and the SDRAM controller or SDRAMC. A DMA controller or DMAC is also found within the AMIF. The external memory interface block acts as the main interface between the external memory and all the functional blocks such as the processors (ARM, DSP), video interface, burst engine, etc.

The asynchronous external memory interface is constructed from an interface circuit to the external memory, a chip select output terminal, its control circuit, and a DMA controller. It accesses up to a maximum of 32M bytes of different types of memories (in case of data width 32 bits).

The external memory area is divided into five blocks, one of which is reserved for SDRAM. The size and location of each one of these blocks is controlled by software. Each one of these memory areas has its own bus width and timing specifications.

The SDRAM controller supports up to 75 MHz, 32 bits wide SDRAM, and provides low-overhead continuous data accesses. The SDRAM controller also has the ability to prioritize the access units to support real-time data streams of video data in and display data out. The resulting high performance equals a system throughput of 320M bytes/sec. The SDRAM controller also provides power-down control for the SDRAM memory chip.

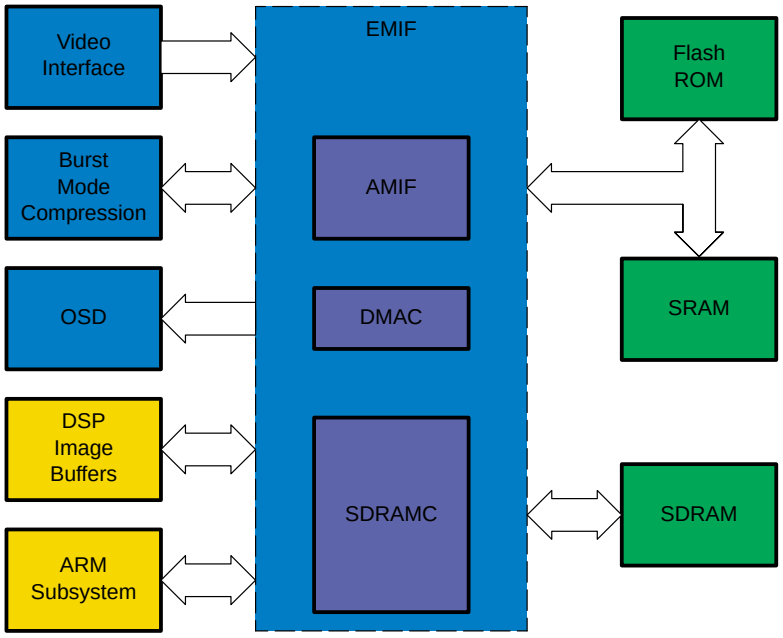


Figure 2–5. External Memory Interface

The EMIF also includes a DMA controller. The transfer of data between the video interface, the SDRAM, and different sections of the external general-purpose memory interface is possible without going through the MCU.

In addition, in the DSC24, the SDRAM, and the AMIF bus can be opened to an external device. By using this function, the external memory can be shared between external devices.

2.7 Different Mode of Operations

The DSC24 allows four different modes of operation. These modes exist to provide the user with a more flexible chip without having too many pins. These four device operating modes are called mode 0 or normal mode, mode 1, mode 2, and mode 3. Table 2–1 describes the differences between these modes. Each individual chapter in this document describes the differences for each individual block of the DSC24.

Table 2–1. Comparison Between the Different Modes of Operation

		Mode 0	Mode 1	Mode 2	Mode 3
Memory interface	Extra hardware wait states	Not possible	Not possible	Possible for the first external memory region (CS0)	Not possible
	Bus open mode	Available	Available	Not available†	Available
	Bus width of CS0 external memory area	8 or 16 bits	8 or 16 bits	16 or 32 bits‡	8 or 16 bits
	SDRAM bus width	16/32 bits‡	16/32 bits‡	16/32 bits‡	16/32 bits‡
Serial ports and GIOs	Number of synchronous serial ports	2	3	2	2
	Number of asynchronous serial ports	One 2-pin UART One 5-pin UART	Two 2-pin UARTs	Two 2-pin UARTs	One 2-pin UART One 5-pin UART
	Number of GIO	21	21	21	32
Clocking options	Watchdog timer output pin	Not available	Not available	Not available	MRST pin available
	Clock output pins	VCLK, clkoutput1, and clkoutput2	clkoutput1 and clkoutput2 only	clkoutput1 and clkoutput2 only	clkoutput1 and clkoutput2 only
Video interface	Video input field indicator	Not available	Available	Not available	Not available
	Number of video input ports	1	2	1	1
	Number of video output ports	1	1§	1	0
	Video output	Synchronized with internal VD/HD	Not available	Synchronized with external VD/HD	Not available
	Video loop through¶	Not possible	Not possible	Possible	Not possible

† The user can use two GIOs instead to open the bus to another device.
‡ The SDRAM bus width is limited to 16 bits when an external memory area is configured for 32 bits.
§ The video output pins are multiplexed with the pins of one of the video input ports
¶ In video loop through mode, the video-input signal is directly sent to the video output pins, the OSD is not used.

The MODE_SEL0 and MODEL_SEL1 pins select the mode of operation. These pins are static and should not be modified when the device is powered.

Table 2–2. Mode of Operation Selection

	MODE_SEL1	MODE_SEL0
Mode 0	0	0
Mode 1	0	1
Mode 2	1	0
Mode 3	1	1

NOTE:
In this document, mode 0 is considered as the standard mode of operation. The other modes are mentioned only when some differences exist.

CAUTION:
Depending on the device-operating mode used, some of the pins of the DSC24 are either input or output. Their initial state can be changed. It is highly recommended to choose which operating mode best suits the application being developed as early as possible in the development cycle. Changing the operating mode on a board already developed might damage your equipment.

3 MCU Subsystem

This chapter deals with the ARM core found in the DSC24, its I/O, and internal peripherals. The imaging peripherals are described in a different chapter.

3.1 Introduction

The MCU subsystem is made up of the ARM7TMDI RISC processor core, static RAM memory, and peripherals. The ARM controls the various peripheral modules including the timers, interrupt controller, video interface, burst compression engine, on-screen display, USB, and serial ports. The user interface should run on this processor.

In a system, the MCU is responsible for handling many system functions such as system-level initialization, configuration, user interface, user command execution, connectivity functions, and overall system control. The general-purpose MCU handles these functions because it has a larger program memory space and better context switching capability, and is thus more suitable for complex, multitasking, and general-purpose control tasks compared to the on-chip DSP.

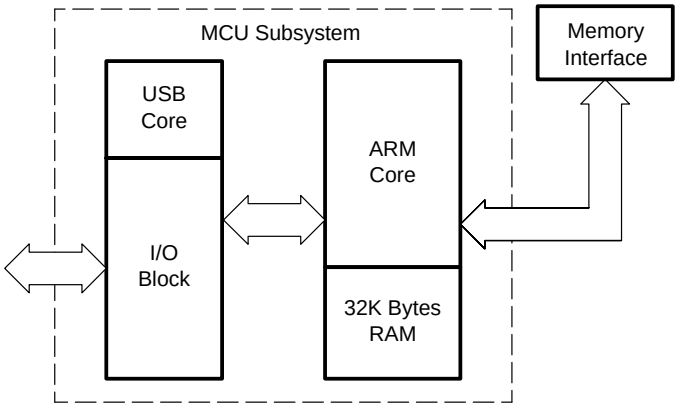


Figure 3–1. MCU Subsystem

The MCU subsystem of the DSC24 offers the following features:

- 32-bit RISC processor
- Maximum operating frequency of 40 MHz
- 32K byte built-in RAM
- 32/16-bit command sets
- Memory format: little endian format fixed
- Two interrupt systems (FIQ, IRQ)
- Eight external terminal interrupt systems
- x4 hardware timers
- Watchdog timer
- JTAG base emulation
- External memory interface

3.2 ARM7TDMI RISC Processor Core

The ARM7TDMI (ARM7) core is integrated in the DSC24 chip. The ARM7 core that is in the DSC24 is based on the TMS470R1x microcontroller from Texas Instruments. The ARM7 core is specified to run up to approximately 37.5 MHz (see the TMS320DSC24 GHK data manual, literature number SPRS195, for the exact value). The ARM subsystem has its own 32K bytes of local static RAM.

The ARM7 processor employs a unique architectural strategy known as the 16-bit instruction set (16-BIS), which makes it ideally suited to high-volume applications with memory restrictions or applications where code density is an issue. A 32-bit instruction set (32-BIS) mode is also available to run complex algorithms.

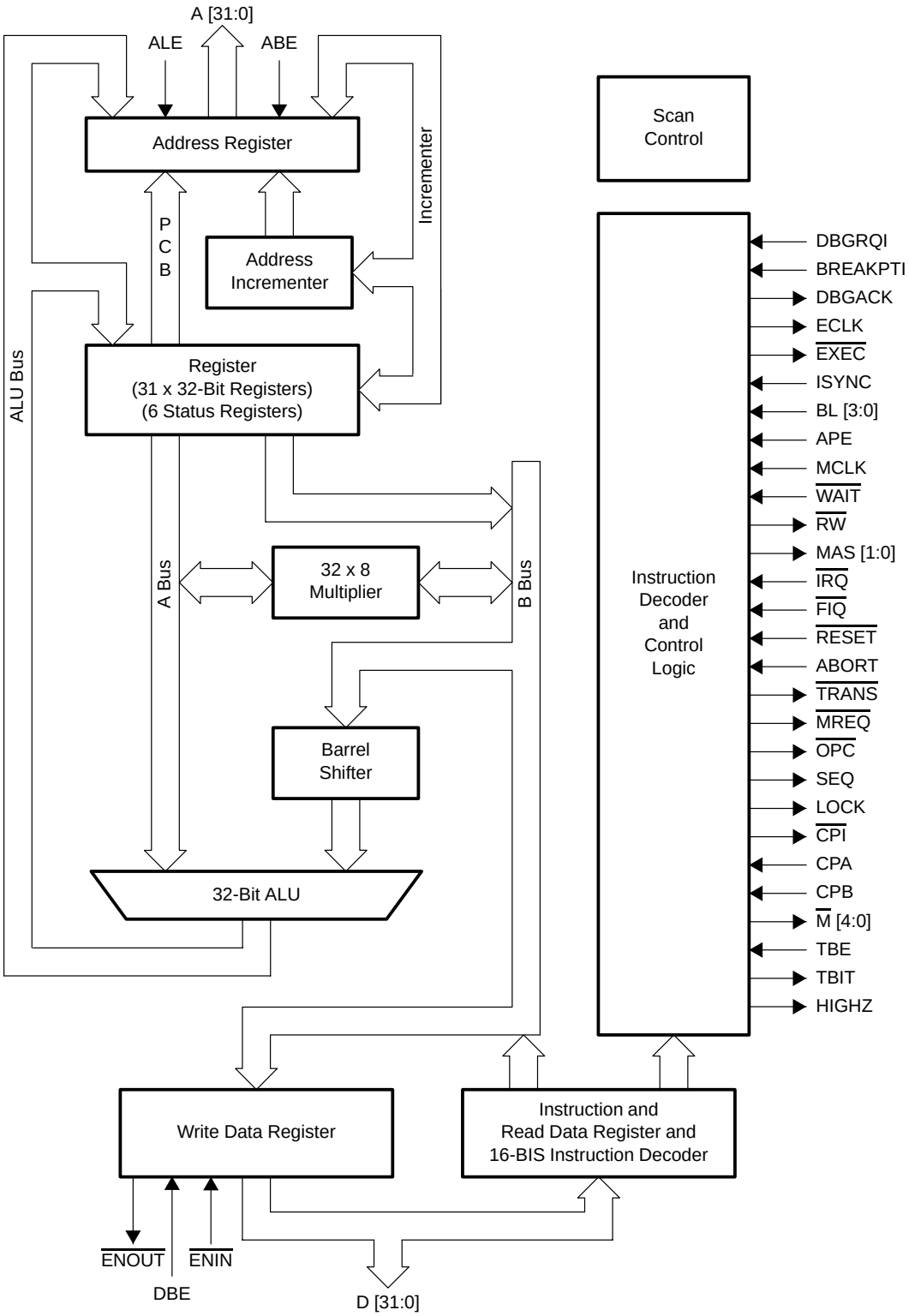


Figure 3–2. ARM7 Core

3.2.1 16-BIS/32-BIS Concept

The key idea behind the 16-BIS concept is that of a super-reduced instruction set. Essentially, the TMS470R1x processor has two instruction sets:

- ARM mode or 32-BIS: the standard 32-bit instruction set
- Thumb mode or 16-BIS: a 16-bit instruction set.

The 16-BIS (16-bit instruction length) allows the processor to approach twice the density of standard 32-BIS code while retaining most of the 32-BIS's performance advantage over a traditional 16-bit processor using 16-bit registers. This is possible because 16-BIS code operates on the same 32-bit register set as 32-BIS code. A 16-bit code is able to provide up to 65% of the code size of the 32-bit code, and 160% of the performance of an equivalent 32-BIS processor connected to a 16-bit memory system.

3.2.2 16-BIS/32-BIS Advantages

Each 16-bit instruction operates with the standard 32-bit register configuration, allowing excellent interoperability between 32-BIS and 16-BIS states. Each 16-bit instruction has a corresponding 32-bit instruction with the same effect on the processor model. The major advantage of a 32-bit architecture over a 16-bit architecture is its ability to manipulate 32-bit integers with single instructions and to address a large address space efficiently. When processing 32-bit data, a 16-bit architecture takes at least two instructions to perform the same task as a single 32-bit instruction. However, not all the code in a program processes 32-bit data (for example, code that performs character string handling), and some instructions, like branches, do not process any data at all. If a 16-bit architecture only has 16-bit instructions, and a 32-bit architecture only has 32-bit instructions, then overall, the 16-bit architecture has better code density and better than one half the performance of the 32-bit architecture. Clearly 32-bit performance comes at the cost of code density. The 16-bit instruction breaks this constraint by implementing a 16-bit instruction length on a 32-bit architecture, making the processing of 32-bit data efficient with a compact instruction coding. This provides far better performance than a 16-bit architecture, with better code density than a 32-bit architecture. The 16-BIS also has a major advantage over other 32-bit architectures with 16-bit instructions. This is the ability to switch back to full 32-bit code and execute at full speed. Thus, critical loops for applications such as fast interrupts and DSP algorithms can be coded using the full 32-BIS and linked with 16-BIS code. The overhead of switching from 16-bit code to 32-bit code is folded into sub-routine entry time. Various portions of a system can be optimized for speed or for code density by switching between 16-BIS and 32-BIS execution as appropriate.

3.2.3 References

For more information on the ARM7 core, see the *TMS470R1x 32-Bit RISC Micro-Controller Family* user's guide, literature number SPNU134.

3.3 MCU Subsystem Memory

3.3.1 Memory Map

The MCU can access all peripheral registers, RAM, and ROM area. The BUS controller is responsible to automatically generate the number of wait states required according to the peripheral module accessed by the MCU. The internal RAM (IRAM) of the ARM can be accessed through a 32-bit width data bus. It is recommended to store the main RTOS within this memory to get the best throughput. All registers of peripheral modules, except for the USB, are accessed through a 16-bit width data bus with no wait-state.

The BUSC module supports three type of data access mode, e.g. a byte, a half word, and a word which is 8 bits, 16 bits, and 32 bits respectively during data or instruction access to the external memories like SDRAM, flash memory and so on. It also supports dynamic allocation of external memory (dynamically allocated external memory or DAEM). The maximum number of partitioning areas is six. This feature is explained in the external memory interface chapter.

In addition, DSP memory can be accessed by the MCU through the HPI bridge interface (see the *ARM-DSP Communication* chapter).

The characteristics of the different memory spaces accessible by the MCU are summarized in Table 3–1 and Figure 3–3.

Table 3–1. MCU Memory Space

AREA NAME	START ADDRESS	END ADDRESS	SIZE	BUS WIDTH	NUMBER OF WAIT STATES
Reset vector ROM	0x0000:0000	0x0000:0003	4 bytes	32 bits	0
ARM internal RAM	0x0000:0004	0x0000:7FFF	32K bytes	8, 16, 32 bits	0
ARM peripheral	0x0003:0000	0x0003:0FFF	4K bytes	16 bits	0†
DSP memory	0x0004:0000	0x0004:FFFF	64K bytes	16 bits	1 minimum
DAEM	0x0010:0000	0x080F:FFFF	128M bytes	8, 16, 32 bits	1 minimum

† The number of cycles for USB register access varies depending on the internal state, however several cycles are required.

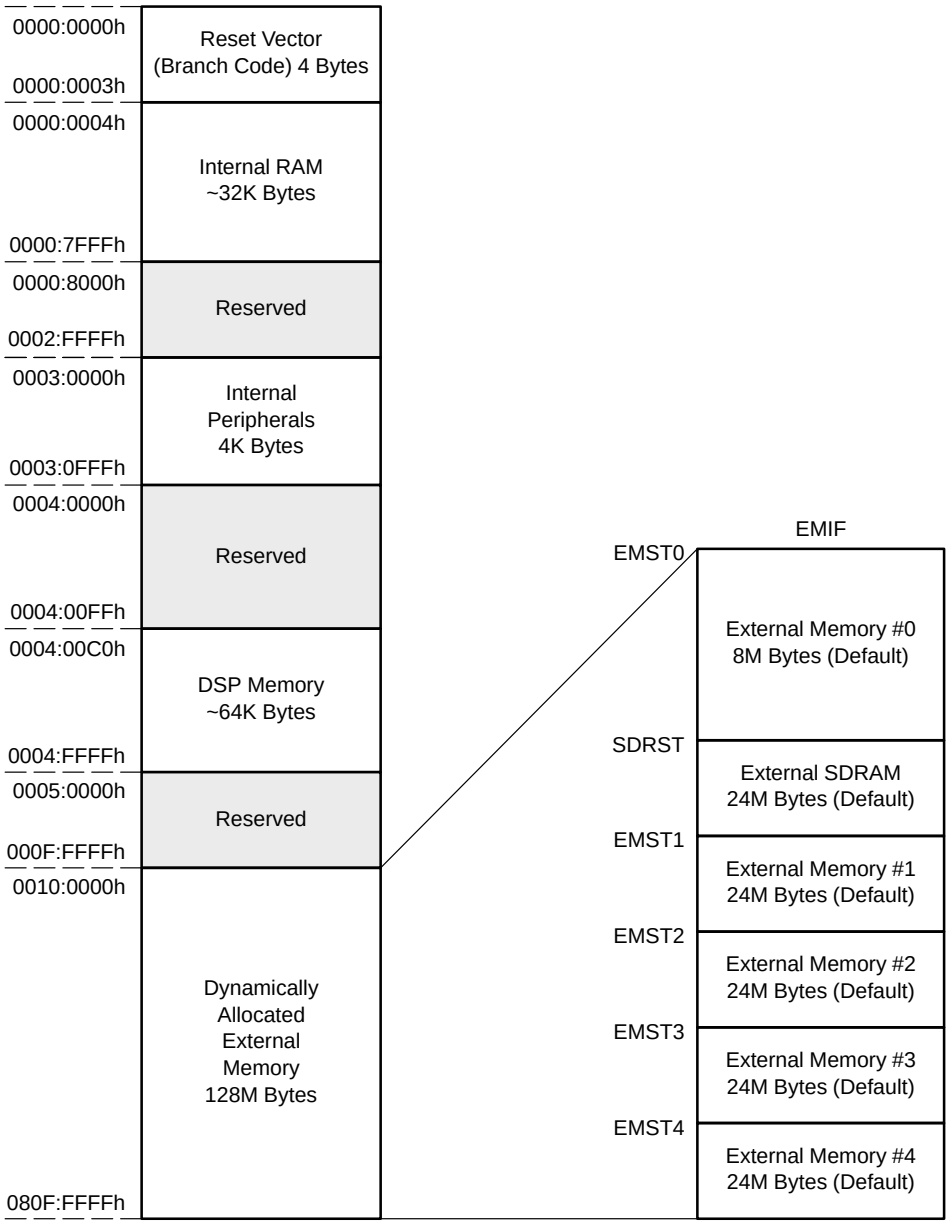


Figure 3–3. MCU Memory Map

3.3.2 Reset Vector ROM

The reset vector ROM (RVR), which is defined at 0x0000:0000 of the ARM internal memory space (see Table 3–1), is hard-coded in BUSC. The MCU accesses this area to fetch instructions after power-on reset. The BUSC provides the MCU with the following code. The code means that jump to the address 0x0010:0000.

```
0000:0000
EA 03 FF FE
; RVR:
B
0010_0000h
```

After the reset signal is cancelled, the DSC24 accesses the external memory address 0 (CS0). Therefore, the program must be written here in advance.

3.3.3 ARM Internal RAM

This is the ARM internal memory (IRAM). Since the MCU accesses this memory region in 1 cycle (no wait state), the RTOS (real time OS) or application programs can be operated more quickly here. Contents of this memory region can be accessed as byte (8 bits); half word (16 bits), or word (32 bits).

3.3.4 Peripheral Registers

The BUSC accesses all internal registers with 16-bit data width. A read and a write access is achieved within one cycle of the MCU clock, except for the USB register access (see the *Handshaking Wait-State Control* section).

Table 3–2. Peripheral Module Memory Map

PERIPHERAL AREA NAME	START ADDRESS	END ADDRESS
Timer 0	0x0003:0000	0x0003:007F
Timer 1	0x0003:0080	0x0003:00FF
Timer 2	0x0003:0100	0x0003:017F
Timer 3	0x0003:0180	0x0003:01FF
Serial 0	0x0003:0200	0x0003:027F
Serial 1	0x0003:0280	0x0003:02FF
Serial 2	0x0003:0300	0x0003:037F
UART 0	0x0003:0380	0x0003:03FF
UART 1	0x0003:0400	0x0003:047F
USB†	0x0003:0480	0x0003:04FF
WDT	0x0003:0500	0x0003:057F
Interrupt controller	0x0003:0580	0x0003:05FF
GIO	0x0003:0600	0x0003:067F
OSD	0x0003:0680	0x0003:06FF
DSP controller	0x0003:0700	0x0003:077F
Video interface	0x0003:0780	0x0003:07FF
Burst mode compression	0x0003:0800	0x0003:087F
Clock controller	0x0003:0880	0x0003:08FF
Bus controller	0x0003:0900	0x0003:097F
SDRAM controller	0x0003:0980	0x0003:09FF
External memory I/F	0x0003:0A00	0x0003:0A7F

† The number of cycles of USB register access varies depending on the internal state, however several cycles are required.

3.3.5 DSP Memory

When the MCU accesses the DSP internal memory through the HPI bridge module (HPIB), one wait-state is inserted automatically. In addition, extra wait cycles are inserted depending on the wait signal that is generated in the HPIB module.

The MCU has no access to the image buffer memories or the expansion memory of the DSP subsystem.

3.3.6 External Memory

As external memory interfaces, the DSC24 has an SDRAM interface and a general-purpose asynchronous memory interface. The MCU accesses general-purpose SRAM or flash ROM via the asynchronous memory interface module. In access from MCU to external memory, a wait state is automatically inserted and its cycle varies depending on SDRAM frequency, external memory access time, and so forth.

The SDRAM controller controls the SDRAM memory access. When the MCU accesses the SDRAM memory, an asynchronous transfer is achieved between SDRAMC and BUSC.

The dynamically allocated external memory region, which is directly accessible by the ARM, is a maximum of 128M bytes and can be divided into a maximum of six regions (SDRAM and EM0 to EM4). By changing the memory map dynamically, external memory of a different capacity can be used efficiently. Each of the six regions is used by setting a start address and a size. Byte (8 bits), half-word (16 bits), and word (32 bits) types of accesses are possible.

The details concerning the external memory accesses are given in the *External Memory Interface* chapter.

3.4 Internal Peripheral

3.4.1 Clock Controller

3.4.1.1 Description

The clock controller generates each of the clocks used within the chip (MCU, DSP, SDRAM, etc.). This MCU peripheral also controls the clock distribution to each module. In the DSC24, there are two digital PLLs built in. The user can freely set the frequencies of the ARM clock, SDRAM clock, and the DSP clock using these PLLs. To reduce system power, each clock frequency can be slew down and each clock distribution can be turned off.

The clock controller block has five different inputs that are connected externally.

- CLKIN1: External clock input (PCLK)
- CLKIN2: External clock input (SYSCLK)
- MXI, MXO: Externally attached XTAL oscillator input
- M48XI, M48XO: Externally attached 48-MHz XTAL oscillator input
- CLKSEL: ARM clock selection

It generates seven signals.

- CLK_ARM: ARM clock for MCU core and its related modules
- CLK_DSP: DSP clock for DSP core and its related modules
- CLK_SDR: SDR clock for external
- CLK_XI : XTAL clock such as TMR and WDT
- CLK_CCD: Video interface clock
- CLK_OSD: OSD clock
- CLK_USB: USB clock

In normal operation, a 13.5-MHz signal is applied to CLKIN1 and a 27-MHz signal is applied to CLKIN2.

3.4.1.2 Block Diagram

Figure 3–4 shows a block diagram of the clock controller. The small lines represent the selector position at the time of reset if the CLKSEL pin is tied low.

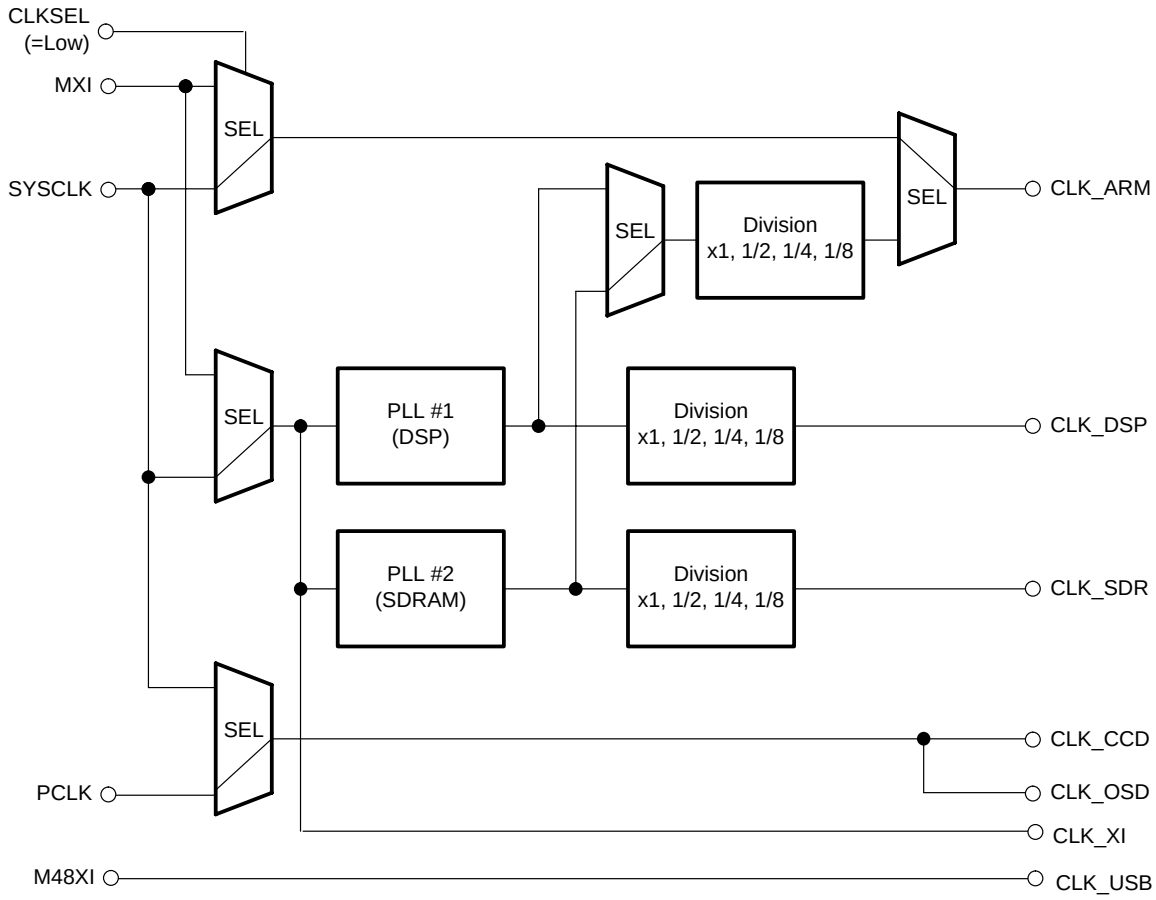


Figure 3–4. Clock Controller Block Diagram

Table 3–3 lists the registers that configure the clock controller.

Table 3–3. Clock Controller Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	00030880	PLLDSP	PLL DSP control
0x01	00030882	PLLSDR	PLL SDRAM control
0x02	00030884	CLKC	Clock controller
0x03	00030886	MOD1	Modules clock enable 1
0x04	00030888	MOD2	Modules clock enable 2
0x05	0003088A	MOD3	Modules clock enable 3

3.4.1.3 Clock Selection

For the ARM clock, two types of input can be selected for the clock source: SYSCLK of an external input clock and MXI input of the XTAL oscillator output. The CLKSEL pin selects the signal. Since the clock must be constantly supplied during reset and during DSC24 operation, the clock source must be selected prior to the ARM clock power being turned on. The CLKSEL pin status must be fixed low or high during normal operation.

Table 3–4. CLKSEL Pin Function

POSITION	FUNCTION
Low	SYSCLK input selected
High	MXI input selected

The ARM clock can also be selected as a division of the DSP PLL output or the SDRAM PLL output. Setting bits 6 and 7 of the CLKC register makes this selection. By using the PLL output clock, it is possible to set the ARM clock frequency that matches the operating mode that is used.

The DSP and SDRAM clocks are generated using respective PLLs. As a PLL reference clock input, the SYSCLK or MXI input of the XTAL oscillator output can be selected. Setting bit 12 of the CLKC register makes this selection. When the DSC24 is operating, the reference clock must be constantly supplied. Also, there are respective PLL output clock division circuits. The division ratios can be set to 1, 1/2, 1/4, or 1/8. Memory-mapped registers control the PLL and division circuit settings.

For the video interface and OSD clocks, the PCLK input or SYSCLK input can be selected by changing the status of the CLKC register bit 2. These clock signals can also have their phase inverted (bits 1 and 0).

The clock controller register is described in Table 3–5. Bits 15 to 13 are detailed in the power saving modes chapter 3, section 3.6.

Table 3–5. Clock Controller (CLKC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSC48	SLPMD	BPSMD	FINSEL	DCDIV1	DCDIV0	SCDIV1	SCDIV0	CASEL	CAMUX	CADIV1	CADIV0	EXCINV	CMUX	CCINV	COSDINV
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
De-fault	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–6. Clock Controller (CLKC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	OSC48	OSC48 cell power down 0: Cell enable 1: Cell disable (See chapter 3, section 3.6 for more information)
14	SLPMD	Sleep mode 0: Normal (can be cleared only by external interrupt signal) 1: Sleep (when PLL is powered down, all clocks are turned off) (See chapter 3, section 3.6 for more information)
13	BPSMD	Bypass mode. (PLL power down) 0: Normal 1: Bypass (See chapter 3, section 3.6 for more information)
12	FINSEL	PLL input clock selection If pin CLKSEL = 0: 0: SYSCLK input 1: MXI input If pin CLKSEL = 1: 0: MXI input 1: SYSCLK input **Note:** It is necessary to have selected FINSEL before changing the CAMUX, CADIV, or CASEL bits. In addition, the M/N setting of each PLL has to be done after this bit is set.

Table 3–6. Clock Controller (CLKC) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
11–10	DCDIV1–DCDIV0	DSP clock division DCDIV1 DCDIV0 0 0 1 division 0 1 2 divisions 1 0 4 divisions 1 1 8 divisions
9–8	SCDIV1–SCDIV0	SDRAM clock division SCDIV1 SCDIV0 0 0 1 division 0 1 2 divisions 1 0 4 divisions 1 1 8 divisions
7	CASEL	ARM clock select 0: External clock 1: Internal clock **Note:** It is necessary to have ended the FINSEL, CADIN, and CAMUX settings before selecting the internal clock.
6	CAMUX	ARM clock multiplexer 0: SDRAM clock 1: DSP clock
5–4	CADIV1–CADIV0	ARM clock division CADIV1 CADIV0 0 0 1 division 0 1 2 divisions 1 0 4 divisions 1 1 8 divisions **Note:** CADIV1:0 must be selected prior to changing the CASEL bit status.
3	EXCINV	External clock (SYSCLK) inversion bit Inverts polarity of external clock 0: Uninverted 1: Inverted
2	CMUX	CCD/OSD clock multiplexer 0: PCLK 1: SYSCLK
1	CCINV	CCD clock inversion bit 0: Uninverted 1: Inverted
0	COSDINV	OSD clock inversion bit 0: Uninverted 1: Inverted

3.4.1.4 Clock Frequency Setting

Figure 3–5 shows the PLL structure. The PLL is a built-in analog PLL constructed of a charge pump, loop filter, and VCO. Frequency is determined by setting the division ratio of the counter N that divides the input clock and the counter M that divides PLL output. The output frequency is determined by the following formula:

$$Fout = Fin \times (M/N)$$

With N = 1…4 and M = 1….31 (it is highly recommended to keep $M \leq 20$).

In the DSC24, two PLLs are built in and the output frequency can be set separately for each.

The time from when the values of M and N are changed until the PLL output is stable (locked state) is measured by a clock consisting of the input clock divided by N. During this time, clock output is stopped so that erroneous operation while the PLL is unstable does not occur. The unlock period is $8192 \times N$ cycles long. Bit 15 of each of the PLL control registers indicates if the PLL is locked or not.

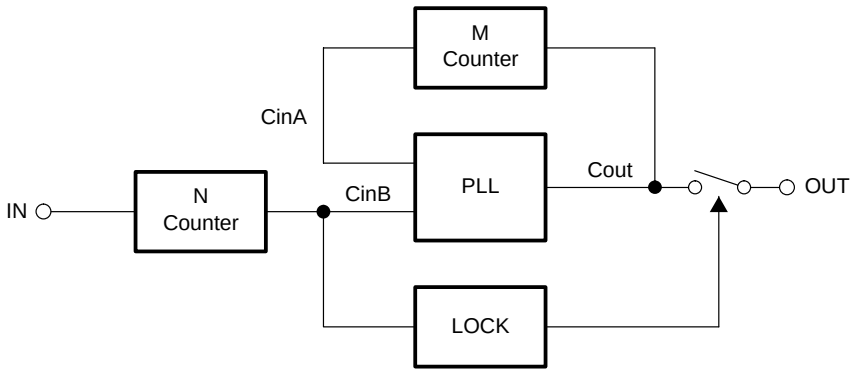


Figure 3–5. PLL Structure

3.4.1.5 Frequencies Settings Constraints

The following conditions must be used to set the frequency of each clock.

For Clk_arm:

Before selecting the internal clocking, the FINSEL, the CAMUX, and the PLL division (CADIV) values must be set. In addition, the M/N setting of each PLL can be set only after FINSEL has been selected.

When the PLL output is used to generate the Clk_arm signal, the PLL M/N settings and FINSEL must be set prior to selecting the CAMUX bit. The CASEL bit must be set to 0 during the modification of the PLL settings.

The recommended order to set the Clk_arm signal is as follows:

1. Set CASEL bit to 0 to select the external clock.
2. Set FINSEL bit according to the users needs.
3. Set the M/N ratio for one or both of the PLLs. The clock stabilizes at the requested frequencies after (Fin/N) x 8192 cycles. The clock is not output from the PLL at this time. Each time FINSEL is modified, this step must be done.
4. Set the CAMUX/CADIV bits. The user can change the status of these bits at the same time or separately.
5. Set CASEL bit to 1.

When this procedure is not followed, the operation is not assured.

For Clk_dsp:

When the FINSEL bit is modified, it is necessary to set the PLL values M and N again. Before doing these modifications, disable the Clk_dsp related module.

Clk_sdr:

When the FINSEL bit is modified, it is necessary to set the PLL values M and N again. Before doing these modifications, disable the Clk_sdr related module.

Other considerations:

Since the operating frequency of each module has an upper limit, a frequency below that limit must be set. See the data sheet, literature number SPRS195 for the limit values.

1. CLK_ARM < MCU maximum frequency
2. CLK_SDR < SDRAM maximum frequency
3. CLK_DSP < DSP maximum frequency

Also, there are restrictions on the frequency settings among different clocks under the following conditions:

- When the MCU accesses SDRAM: CLK_SDR > CLK_ARM
- DMA transfer between the image buffer and SDRAM: CLK_DSP > CLK_SDR
- DMA transfer between the image buffer and external memory: CLK_DSP > CLK_ARM

If any of the following conditions is not satisfied, the proper operation of the PLL is not assured.

- Fin/N > 5 MHz
- Fin x (M/N) > 50 MHz
- Fin x (M/N) < 300 MHz
- M ≤ 20
- N ≤ 4
- M/N ≥ 1

3.4.1.6 PLL Settings

The following registers control the DSP and SDRAM PLLs.

Table 3–7. PLL DSP Control (PLLDSP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DLMTR	RSV	RSV	RSV	RSV	RSV	RSV	DPM4	DPM3	DPM2	DPM1	DPM0	RSV	RSV	DPN1	DPN0
R/W	R	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W
Default	0	—	—	—	—	—	—	0	0	0	0	1	—	—	0	1

Table 3–8. PLL DSP Control (PLLDSP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	DLMTR	DSP clock lock monitor A 1 represents locked status
14–9	RSV	Reserved bits. Not used.
8–4	DPM4–DPM0	DSP PLL constant M DPM4 – DPM0 0 0 0 0 0 M = 1 0 0 0 0 1 M = 1 0 0 0 1 0 M = 2 : : 1 0 1 0 0 M = 20 1 0 1 0 1 Reserved : : 1 1 1 1 1 Reserved
3–2	RSV	Reserved bits. Not used.
1–0	DPN1–DPN0	DSP PLL constant N DPN1 DPN0 0 0 N = 4 0 1 N = 1 1 0 N = 2 1 1 N = 3

Table 3–9. PLL SDRAM Control (PLLSDR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SLMTR	RSV	RSV	RSV	RSV	RSV	RSV	SPM4	SPM3	SPM2	SPM1	SPM0	RSV	RSV	SPN1	SPN0
R/W	R	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W
Default	0	—	—	—	—	—	—	0	0	0	0	0	—	—	0	1

Table 3–10. PLL SDRAM Control (PLLSDR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	SLMTR	SDRAM clock lock monitor A 1 represents locked status
14–9	RSV	Reserved bits. Not used.
8–4	SPM4–SPM0	SDRAM PLL constant M SPM4 – SPM0 0 0 0 0 0 M = 1 0 0 0 0 1 M = 1 0 0 0 1 0 M = 2 : : 1 0 1 0 0 M = 20 1 0 1 0 1 Reserved : : 1 1 1 1 1 Reserved
3–2	RSV	Reserved bits. Not used.
1–0	SPN1–SPN0	SDRAM PLL constant N SPN1 SPN0 0 0 N = 4 0 1 N = 1 1 0 N = 2 1 1 N = 3

3.4.1.7 Clock Distribution

On/off of clock supply can be controlled individually for each of the 28 modules in the DSC24. Table 3–11 shows each module name and the type of clock that can be used by that module. To reduce power consumption, clock on/off must be controlled according to the operating mode. Table 3–11 also shows the status of each module at reset.

Table 3–11. Clock Distribution

REGISTER NAME	MODULE	USED CLOCK	STATUS AFTER RESET
CERAM	DSP expansion memory	CLK_DSP	Off
C5409	DSP core and internal memory	CLK_DSP	On
CIMG	Image buffer	CLK_ARM, CLK_SDR, CLK_DSP	Off
CIMX	IMX	CLK_DSP	Off
CVLC	VLC	CLK_DSP	Off
CHPIB	HPI bridge	CLK_ARM	On
CSDRC	SDRAM controller	CLK_ARM, CLK_SDR	Off
CGIO	GIO	CLK_ARM	Off
CINTC	Interrupt controller	CLK_ARM	On
CACORE	ARM core	CLK_ARM	On in internal CPU mode (default) Off if the external host mode is used
CAIM	ARM internal memory	CLK_ARM	ON in internal CPU mode (default) Off if the external host mode is used

Table 3–11. Clock Distribution (Continued)

REGISTER NAME	MODULE	USED CLOCK	STATUS AFTER RESET
CUSB	USB	CLK_USB, CLK_ARM	Off
CUAT1	UART1	CLK_ARM	Off
CUAT0	UART0	CLK_ARM	Off
CTMR3	TIMER3	CLK_ARM, CLK_XI	Off
CTMR2	TIMER2	CLK_ARM, CLK_XI	Off
CTMR1	TIMER1	CLK_ARM, CLK_XI	Off
CTMR0	TIMER0	CLK_ARM, CLK_XI	Off
CWDT	Watchdog timer	CLK_ARM, CLK_XI	Off
CBRST	Burst mode codec	CLK_ARM, CLK_CCD, CLK_SDR	Off
COSD	OSD	CLK_ARM, CLK_CCD, CLK_SDR	Off
CBUSC	Bus controller	CLK_ARM, CLK_SDR	Off in internal CPU mode (default) On if the external host mode is used
CSIF2	Serial I/F 2	CLK_ARM	Off
CSIF1	Serial I/F 1	CLK_ARM	On
CSIF0	Serial I/F 0	CLK_ARM	Off
CVIF	Video interface	CLK_ARM, CLK_CCD, CLK_SDR	Off
CHSTIF	External CPU I/F	CLK_ARM	Off in internal CPU mode (default) On if the external host mode is used
CEMIF	External memory I/F	CLK_ARM	On in internal CPU mode (default) Off if the external host mode is used

The MOD1, MOD2, and MOD3 registers control the input clock of each module.

Table 3–12. Modules Clock Enable 1 (MOD1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	CERAM	C5409	CIMG	CIMX	CVLC	CHPIB	CSDRC	CGIO	CINTC	CACOR	CAIM
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	1	0	0	0	1	0	0	1	0/1†	0/1†

† Initial value is 0 in external host mode and 1 in an internal CPU mode (normal mode of operation).

Table 3–13. Modules Clock Enable 1 (MOD1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10	CERAM	C5409 expansion RAM clock 0: Disabled 1: Enabled
9	C5409	C5409 clock 0: Disabled 1: Enabled
8	CIMG	Image buffer clock 0: Disabled 1: Enabled See bit 2 of CCAI register within the DSP I/O space
7	CIMX	IMX clock 0: Disabled 1: Enabled See bit 1 of CCAI register within the DSP I/O space
6	CVLC	VLC clock 0: Disabled 1: Enabled See bit 1 of CCAI register within the DSP I/O space
5	CHPIB	HPIB clock 0: Disabled 1: Enabled
4	CSDRC	SDRAM controller clock 0: Disabled 1: Enabled
3	CGIO	GIO clock 0: Disabled 1: Enabled
2	CINTC	Interrupt controller clock 0: Disabled 1: Enabled
1	CACORE	ARM core clock 0: Disabled 1: Enabled
0	CAIM	ARM internal memory clock 0: Disabled 1: Enabled

Table 3–14. Modules Clock Enable 2 (MOD2) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	CUSB	CUAT1	CUA10	CTMR3	CTMR2	CTMR1	CTMR0	CWDT
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–15. Modules Clock Enable 2 (MOD2) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7	CUSB	USB clock 0: Disabled 1: Enabled
6	CUAT1	UART1 clock 0: Disabled 1: Enabled
5	CUAT0	UART0 clock 0: Disabled 1: Enabled
4	CTMR3	TIMER3 clock 0: Disabled 1: Enabled
3	CTMR2	TIMER2 clock 0: Disabled 1: Enabled
2	CTMR1	TIMER1 clock 0: Disabled 1: Enabled
1	CTMR0	TIMER0 clock 0: Disabled 1: Enabled
0	CWDT	Watchdog timer clock 0: Disabled 1: Enabled

Table 3–16. Modules Clock Enable 3 (MOD3) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	BAINV	BCINV	CBRST	COSD	CBUSC	CSIF2	CSIF1	CSIF0	CVIF	CHSTIF	CEMIF	RSV
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
Default	—	—	—	—	0	0	0	0	0/1†	0	0	0	0	1/0‡	0/1†	0

† Initial value is 0 in external host mode and 1 in an internal CPU mode (normal mode of operation)
‡ Initial value is 1 in external host mode and 0 in an internal CPU mode (normal mode of operation)

Table 3–17. Modules Clock Enable 3 (MOD3) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11	BAINV	Burst compression ARM clock inversion bit 0: Uninverted 1: Inverted
10	BCINV	Burst compression CCD clock inversion bit 0: Uninverted 1: Inverted
9	CBRST	Burst compression clock 0: Disabled 1: Enabled
8	COSD	OSD clock 0: Disabled 1: Enabled
7	CBUSC	Bus controller clock 0: Disabled 1: Enabled
6	CSIF2	Serial interface 2 clock 0: Disabled 1: Enabled
5	CSIF1	Serial interface 1 clock 0: Disabled 1: Enabled
4	CSIF0	Serial interface 0 clock 0: Disabled 1: Enabled
3	CVIF	Video interface clock 0: Disabled 1: Enabled
2	CHSTIF	External host interface clock 0: Disabled 1: Enabled
1	CEMIF	External memory interface clock 0: Disabled 1: Enabled
0	RSV	Reserved bit. Not used.

Figure 3–6 shows what selectors the MODx registers affect.

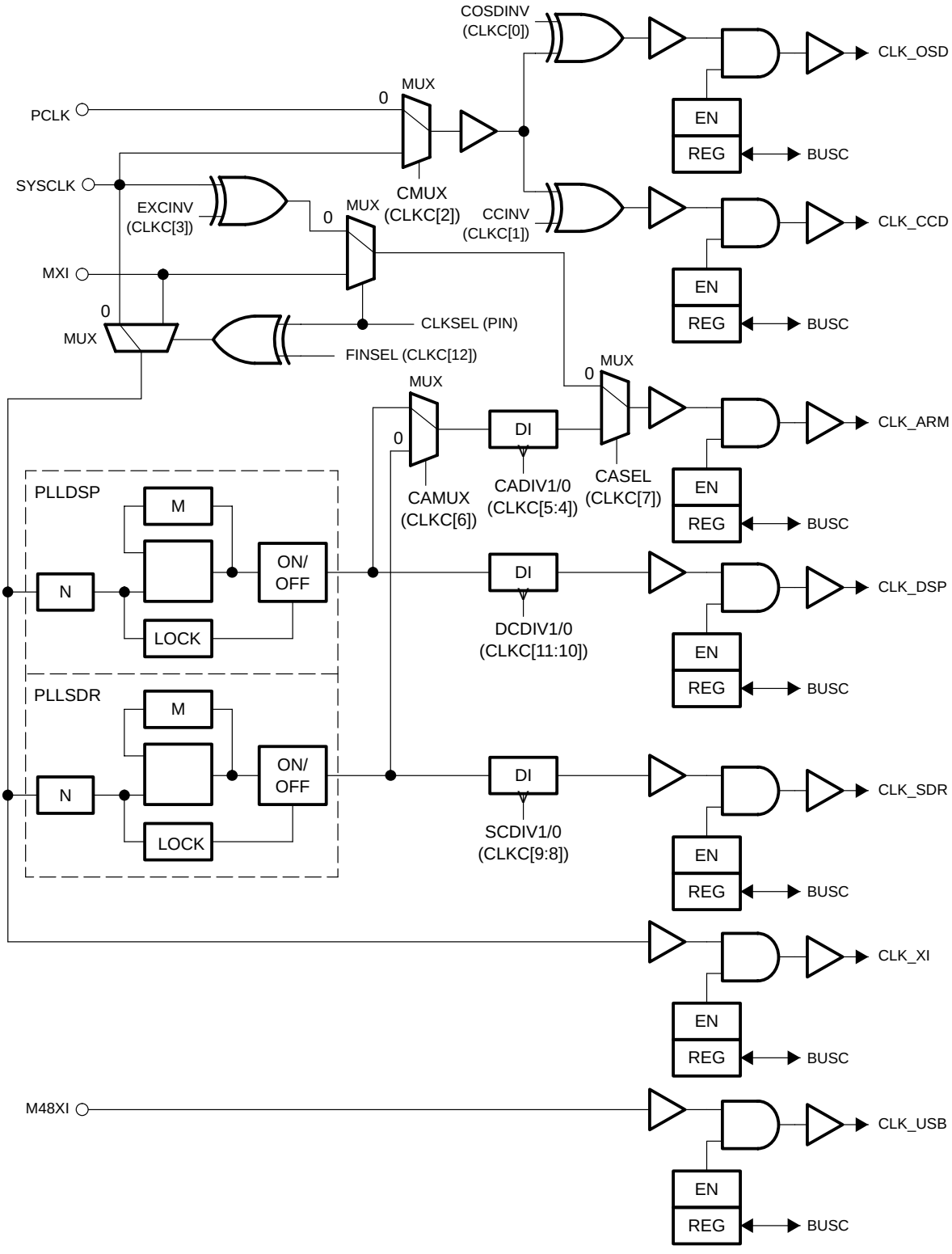


Figure 3–6. Clock Controller Block Diagram

3.4.1.8 Clock outputs

There are a couple of pins that can be used to output internal clock signals.

When the device mode 0 is used, the VCLK pin is connected to the CLK_CCD output of the clock controller module. The frequency of the VCLK signal is usually identical to the one of the PCLK input pin (but delayed). This pin has either a different function or is not available when any of the other device modes are used.

Two general input/output pins (GIO) can also be configured as clock outputs.

The GIO16 can be used to generate one of the following clocks:

- Arm_clk / ((Div1+1) x 2) with Div1 in the range 0~255
- CLK_XI clock
- CLK_XI clock divided by 2

The GIO17 can be used to generate one of the following clocks:

- Arm_clk / ((Div2+1) x 2) with Div2 in the range 0~255
- CLK_USB clock
- CLK_USB clock divided by 2

See the *GIO* chapter for information on how to configure GIO16 and GIO17.

3.4.2 Timers (x4)

The TMS320DSC24 MCU subsystem has four timers. The output of each timer is input into an interrupt. Timer 0 is connected to interrupt 0, timer 1 is connected to interrupt 1, timer 2 is connected to interrupt 2, and timer 3 is connected to interrupt 3. When the timer count reaches 0, interrupt occurs.

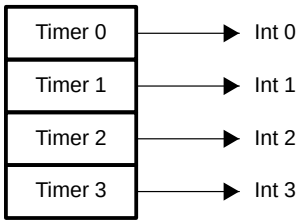


Figure 3–7. ARM Timer Interrupts

Each of the general-purpose timers runs in one of two modes. In one-shot mode, an interrupt only occurs once and then the timer must be explicitly reset to begin the timing operation again. In free-run mode, when the timer generates an interrupt, the timer counter is automatically reloaded to start the count operation again.

Either an ARM clock or external clock can be selected as the clock source of the timer. The actual count operation is divided by the value set in the TMPRSCLx register (10 bits). The count value is set in the TMVALx register (16 bits). Also, when a timer is set to one-shot mode, writing a 1 in the TMTRGx register starts the timer. In this section, x can take on values from 0 through 3, indicating timer 0 through timer 3, respectively.

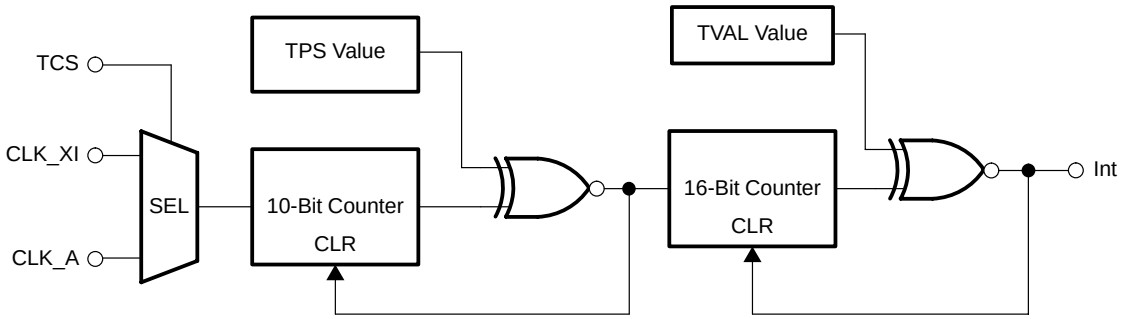


Figure 3–8. Timer Block Diagram

The control registers for the four timers are described in the following tables.

Table 3–18. Timer Registers

Offset	Timer 0 Address	Timer 1 Address	Timer 2 Address	Timer 3 Address	Register Name	
0x00	0x0003:0000	0x0003:0080	0x0003:0100	0x0003:0180	TMMDx	Timer mode
0x01	0x0003:0002	0x0003:0082	0x0003:0102	0x0003:0182	TMCLKx	Timer clock select
0x02	0x0003:0004	0x0003:0084	0x0003:0104	0x0003:0184	TMPSx	Timer prescaler divide value
0x03	0x0003:0006	0x0003:0086	0x0003:0106	0x0003:0186	TMVALx	Timer maximum counter value
0x04	0x0003:0008	0x0003:0088	0x0003:0108	0x0003:0188	TMTRGx	Timer trigger
0x05	0x0003:000A	0x0003:008A	0x0003:010A	0x0003:018A	TMCNTx	Timer current count value

Table 3–19. Timer Mode (TMMDx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	TSTx5	TSTx4	TSTx0	TSTx2	TSTx1	TMSx0	TMSx1	TMSx0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–20. Timer Mode (TMMDx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7–2	TSTx5–TSTx0	Test bits. Set to 0.
1–0	TMSx1–TMSx0	Timer mode select 0 0: Stop. When this mode is selected, the timer continues to count up until the set value is reached. When this happens an interrupt ocurs and the counter stops (used to stop from free-run mode). 0 1: One-shot mode. Count is started when 1 is written in one shot trigger bit. When the set value is reached an interrupt occurs and count operation is stopped. 1 0: Free run mode. Count is repeated up to set value. Each time set value is reached, interrupt occurs. 1 1: Reserved

Table 3–21. Timer Clock Select (TMCLKx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	TCSx
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–22. Timer Clock Select (TMCLKx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	TCSx	Timer clock select 0: Current ARM operation clock 1: External MXO/MXI or SYSCLK input clock

Table 3–23. Timer Prescaler Divide Value (TMPSx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	TPSx9	TPSx8	TPSx7	TPSx6	TPSx5	TPSx4	TPSx3	TPSx2	TPSx1	TPSx0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	X	X	X	X	X	X	X	X	X	X

Table 3–24. Timer Prescaler Divide Value (TMPSx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	TPSx7–TPSx0	Timer prescaler value. TMCLKx is divided by this value. The division ratio is the set value + 1.

Table 3–25. Timer Maximum Counter Value (TMVALx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	TVx15	TVx14	TVx13	TVx12	TVx11	TVx10	TVx9	TVx8	TVx7	TVx6	TVx5	TVx4	TVx3	TVx2	TVx1	TVx0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 3–26. Timer Maximum Counter Value (TMVALx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	TVx15–TVx0	Timer-count value. The count value is the set value + 1. Interrupt occurs at clock cycle x TMPSx x TMVALx cycle.

Table 3–27. Timer Trigger (TMTRGx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	TTRGx
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–28. Timer Trigger (TMTRGx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	TTRGx	One-shot trigger. When the timer is set to one-shot mode, the timer-count operation starts when 1 is written.

Table 3–29. Timer Current Count Value (TMCNTx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	CVx15	CVx14	CVx13	CVx12	CVx11	CVx10	CVx9	CVx8	CVx7	CVx6	CVx5	CVx4	CVx3	CVx2	CVx1	CVx0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 3–30. Timer Current Count Value (TMCNTx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	CVx15–CVx0	Indicates current timer value

3.4.3 Watchdog Timer

The watchdog timer (WDT) is intended to stop a system from running out of control. The WDT is a check and balance mechanism the user sets up so if the system does not go through the proper execution paths, the timer does not reset, and an interrupt occurs. The MCU handles all interrupts and determines that this is a WDT interrupt and restarts the application.

The five registers that control the WDT are listed in Table 3–31.

Table 3–31. Watchdog Timer Registers

Offset	Address	Register Name	
0x0	0x0003:0500	WDTMD	Watchdog timer mode
0x1	0x0003:0502	WDTRST	Watchdog timer reset
0x2	0x0003:0504	WDTPRSCL	Watchdog timer prescaler
0x3	0x0003:0506	WDTVAL	Watchdog timer count value
0x4	0x0003:0508	WDTEXRST	Watchdog timer external reset

At reset, the WDT is disabled. To use it, the user must configure the prescaler register prior to enabling the WDT.

When the DSC24 is used in mode 3, the WDT outputs a signal for resetting an external device. Figure 3–9 shows how the reset pin, the WDT, the MCU reset signal, and the reset output pin (MRST) are connected to each other.

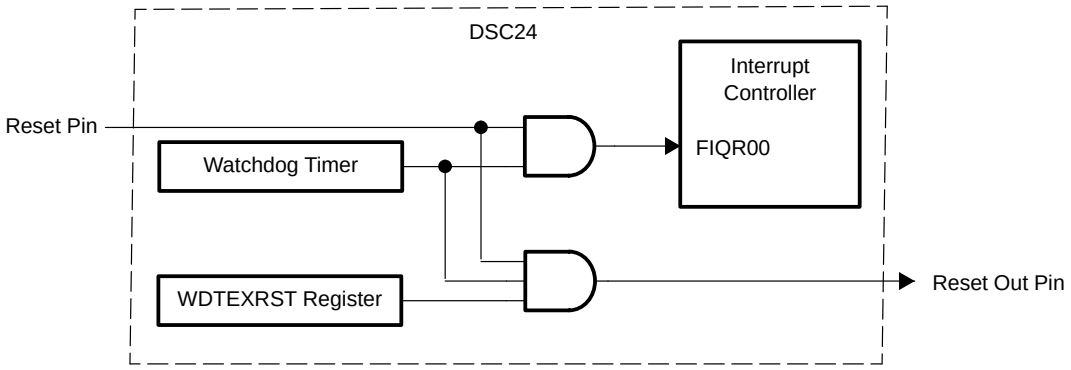


Figure 3–9. WDT Reset Diagram

This peripheral always outputs a signal to the MRST pin. When the counter reaches zero, this pin is pulled down.

The WDT interrupt signal can be connected or disconnected from the fast interrupt request (FIQ) port of the ARM depending on the WERST bit configuration.

The different registers that control the WDT are described below.

Table 3–32. Watchdog Timer Mode (WDTMD) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	OS	RSV	WDTEN
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	W	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–33. Watchdog Timer Mode (WDTMD) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved bits. Not used.
2	OS	Output select 0: Interrupt the MCU 1: interrupt and reset the MCU
1	RSV	Reserved bit. Not used.
0	WDTEN	Watchdog timer enable bit 0: Disabled 1: Enabled

Table 3–34. Watchdog Timer Reset (WDTRST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WDRT
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–35. Watchdog Timer Reset (WDTRST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	WDTRST	Watchdog timer reset. When 1 is written, the watchdog timer is reset.

Table 3–36. Watchdog Timer Prescaler (WDTPRSCL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	WPS9	WPS8	WPS7	WPS6	WPS5	WPS4	WPS3	WPS2	WPS1	WPS0
R/W	—	—	—	—	—	—	W	W	W	W	W	W	W	W	W	W
Default	—	—	—	—	—	—	X	X	X	X	X	X	X	X	X	X

Table 3–37. Watchdog Timer Prescaler (WDTPRSCL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	WPS9–WPS0	Watchdog timer prescaler. The clock is divided by the set value + 1 and clock count is performed.

Table 3–38. Watchdog Timer Divisor (WDTVAL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	WTV15	WTV14	WTV13	WTV12	WTV11	WTV10	WTV9	WTV8	WTV7	WTV6	WTV5	WTV4	WTV3	WTV2	WTV1	WTV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Table 3–39. Watchdog Timer Divisor (WDTVAL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	WTV15–WTV0	Watchdog timer count value. The set value + 1 is counted and interrupt or reset is generated in MCU. The watchdog timer detection time is set by a combination with WDTPRSCL. Detection time [ms] = (WDTPRSCL x WDTDIV)/(External clock input [MHz])

Table 3–40. Watchdog Timer External Reset (WDTEXRST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WERST
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	1

Table 3–41. Watchdog Timer External Reset (WDTEXRST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	WERST	Watchdog timer-reset signal. 0: Outputs reset signal only to external reset output pin (VDOUT) 1: Outputs reset signal to MCU and external reset output pin (VDOUT)

3.4.4 Interrupt Controller

The ARM core can accept only two interrupts: fast-interrupt request (FIQ) and interrupt request (IRQ). The FIQ is connected to the WDT only. The IRQ is connected to the interrupt controller (INTC).

The INTC is constructed of various interrupt status registers and of an interrupt mask control register.

Table 3–42. Interrupt Controller Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0580	FIQR	Fast interrupt
0x01	0x0003:0582	IRQ0R	Interrupt register 0
0x02	0x0003:0584	IRQ1R	Interrupt register 1
0x10	0x0003:05A0	EFIQR	Enable FIQ
0x11	0x0003:05A2	EIRQ0R	Enable IQR 0
0x12	0x0003:05A4	EIRQ1R	Enable IQR 1
0x20	0x0003:05C0	INTIDR	Interrupt ID register
0x21	0x0003:05C2	INTRAW	Interrupts RAW

3.4.4.1 Fast Interrupt Request (FIQ)

The FIQ is connected to the WDT only, so, only two bits are used to control this interrupt. WDIEN is used to mask or unmask the WDT interrupt. FIRQ00 is used as a flag. Regardless of the status of WDIEN, any interrupt generated by the WDT module resets the FIQR00 bit to 0. To clear this bit, a 1 must be written in its location.

Figure 3–10 shows the WDT interrupt path.

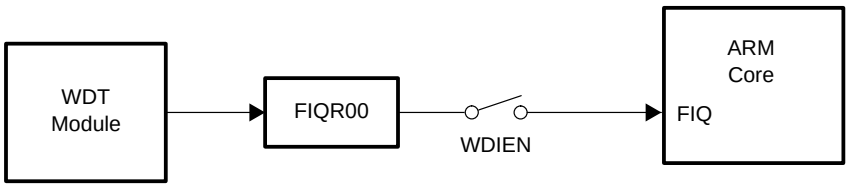


Figure 3–10. FIQ Structure

The two control registers are described in Table 3–43 through Table 3–46.

Table 3–43. Fast Interrupt (FIQR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	FIRQ00
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	1

Table 3–44. Fast Interrupt (FIQR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	FIRQ00	Watchdog timer interrupt cause. When interrupt is generated from the watchdog timer, it becomes 0. When 1 is written, it is cleared.

Table 3–45. Enable FIQ (EFIQR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WDIEN
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–46. Enable FIQ (EFIQR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	WDIEN	Watchdog timer interrupt permission bit. If interrupt occurs in the state where 1 has been written in this bit, the interrupt signal FIQ to ARM is made active. 0: Disabled 1: Enabled

3.4.4.2 Peripherals Interrupt Requests

The INTC multiplexes the interruption signals coming from the different modules and external interrupt pins and then informs the ARM core by using the IRQ signal. All interrupt request signals from the peripherals are active low pulse and are stored within the flag registers of INTC only once. If an interrupt occurs while an identical one is pending, the interrupt request is lost.

In the interrupt controller, since each interrupt cause is output simply by OR, the interrupt priority order must be implemented by software on the MCU side. Figure 3–11 shows a schematic diagram of the interrupt controller.

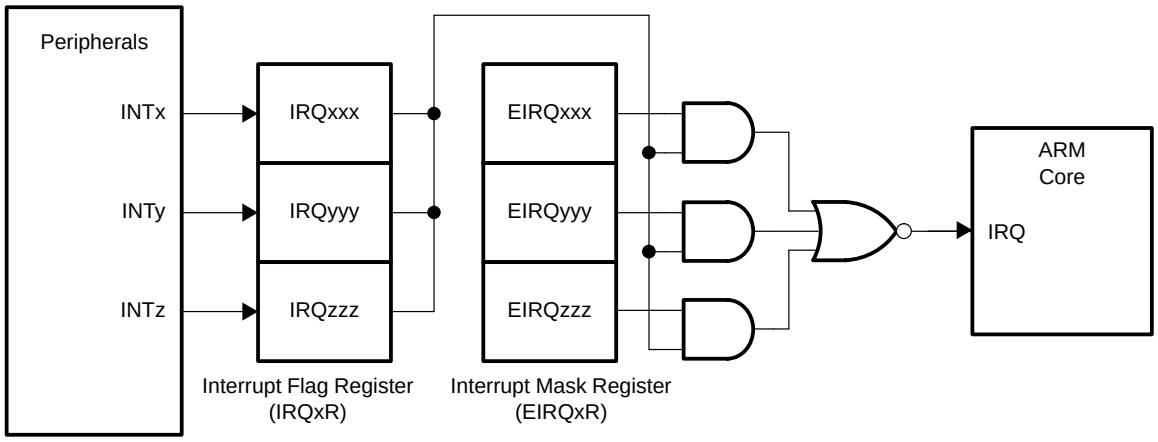


Figure 3–11. Schematic Diagram of Interrupt/Controller

The assignment of each interrupt is given in Table 3–47.

Table 3–47. ARM Interrupt Causes

PRIORITY	INTERRUPT ID	REGISTER FLAG	INTERRUPT CAUSE	INTERRUPT NAME
Low	0x1A	IRQ110	External interrupt 7	XINT7
	0x19	IRQ109	External interrupt 6	XINT6
	0x18	IRQ108	External interrupt 5	XINT5
	0x17	IRQ107	External interrupt 4	XINT4
	0x16	IRQ106	External interrupt 3	XINT3
	0x15	IRQ105	External interrupt 2	XINT2
	0x14	IRQ104	External interrupt 1	XINT1
	0x13	IRQ103	External interrupt 0	XINT0
	0x12	IRQ102	External memory I/F 1	EMIFINT1
	0x11	IRQ101	External memory I/F 0	EMIFINT0
	0x10	IRQ100	Burst compression end	BCINT
	0x0F	IRQ015	Burst decompression end	BDINT
	0x0E	IRQ014	USB	USBINT
	0x0D	IRQ013	UART1	UART1INT
	0x0C	IRQ012	UART0	UART0INT
	0x0B	IRQ011	DSP HINT	DSPHINT
	0x0A	IRQ010	Serial I/F 2	SIF2INT
	0x09	IRQ009	Serial I/F 1	SIF1INT
	0x08	IRQ008	Serial I/F 0	SIF0INT
	0x07	IRQ007	OSD	OSDINT
	0x06	IRQ006	Reserved	
	0x05	IRQ005	Video I/F VD1	VD1INT
	0x04	IRQ004	Video I/F VD0	VD0INT
	0x03	IRQ003	Timer 3	T3INT
	0x02	IRQ002	Timer 2	T2INT
	0x01	IRQ001	Timer 1	T1INT
High	0x00	IRQ000	Timer 0	T0INT

Since interrupt signals to the MCU are output at level, any pending interrupt flag must be cleared prior to unmasking the corresponding interrupt.

The interrupt controller also has a function that helps to speed up the interrupt processing routine. It is possible to know whether any interrupt occurred using a register that displays the currently generated interrupt ID number (INTIDR[4:0]).

Table 3–48 shows the interrupt ID numbers. For example, if OSD interrupt occurs, the number 7 is displayed in the ID register. If multiple interrupts occur, the lowest ID number is displayed first (e.g., the one with the highest priority). When that interrupt is cleared, the next smallest ID number is displayed. For example, if timer 2 interrupt and video I/F VD1 interrupt occurs, 2 is first displayed in the ID register, and when the timer 2 interrupt status bit is cleared, the number 5 is then displayed. By using this feature, the time it takes to perform an interrupt status bit search by the software after an interrupt occurs is reduced, and the interrupt process is speeded up. The INTRAW register allows the user to specify if they want the masked interrupts to be reflected in the INTIDR register or not.

The interrupt control registers are described below. For details of interrupt, see the section about each module.

Table 3–48. Interrupt Register 0 (IRQ0R) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IRQ015	IRQ014	IRQ013	IRQ012	IRQ011	IRQ010	IRQ009	IRQ008	IRQ007	IRQ006	IRQ005	IRQ004	IRQ003	IRQ002	IRQ001	IRQ000
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 3–49. Interrupt Register 0 (IRQ0R) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	IRQ015–IRQ000	Interrupt 1 cause bits. If interrupt occurs, the corresponding bit becomes 0. When 1 is written in this bit, it is cleared. The interrupt cause corresponding to each bit is as follows. IRQ015 Burst decompression end IRQ014 USB IRQ013 UART1 IRQ012 UART0 IRQ011 DSP HINT IRQ010 Serial interface 2 IRQ009 Serial interface 1 IRQ008 Serial interface 0 IRQ007 OSD IRQ006 Reserved IRQ005 Video interface VD1 IRQ004 Video interface VD0 IRQ003 Timer 3 IRQ002 Timer 2 IRQ001 Timer 1 IRQ000 Timer 0

Table 3–50. Interrupt Register 1 (IRQ1R) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	IRQ110	IRQ109	IRQ108	IRQ107	IRQ106	IRQ105	IRQ104	IRQ103	IRQ102	IRQ101	IRQ100
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	1	1	1	1	1	1	1	1	1	1	1

Table 3–51. Interrupt Register 1 (IRQ1R) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	IRQ110–IRQ100	Interrupt 0 cause bits. If interrupt occurs, the corresponding bit becomes 0. When 1 is written in this bit, it is cleared. The interrupt cause corresponding to each bit is as follows. IRQ110 External interrupt 7 IRQ109 External interrupt 6 IRQ108 External interrupt 5 IRQ107 External interrupt 4 IRQ106 External interrupt 3 IRQ105 External interrupt 2 IRQ104 External interrupt 1 IRQ103 External interrupt 0 IRQ102 External memory I/F 1 IRQ101 External memory I/F 0 IRQ100 Burst compression end

Table 3–52. Enable IRQ 0 (EIRQ0R) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EIRQ015	EIRQ014	EIRQ013	EIRQ012	EIRQ011	EIRQ010	EIRQ009	EIRQ008	EIRQ007	EIRQ006	EIRQ005	EIRQ004	EIRQ003	EIRQ002	EIRQ001	EIRQ000
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–53. Enable IRQ 0 (EIRQ0R) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	EIRQ015–EIRQ000	Interrupt 1 permission bits. Equivalent to each bit of IRQ0R. Permits or prohibits each interrupt. Interrupt to the ARM does not occur in prohibited status, but each bit of IRQ0R reflects the status of the interrupt. 0: Prohibited 1: Permitted

Table 3–54. Enable IRQ 1 (EIRQ1R) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	EIRQ110	EIRQ109	EIRQ108	EIRQ107	EIRQ106	EIRQ105	EIRQ104	EIRQ103	EIRQ102	EIRQ101	EIRQ100
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–55. Enable IRQ 1 (EIRQ1R) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	EIRQ110–EIRQ100	Interrupt 1 permission bits. Equivalent to each bit of IRQ1R. Permits or prohibits each interrupt. Interrupt to the ARM does not occur in prohibited status, but each bit of IRQ1R reflects the status of the interrupt. 0: Prohibited 1: Permitted

Table 3–56. Interrupt ID Register (INTIDR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	INTFN5	INTFN4	INTFN3	INTFN2	INTFN1	INTFN0	RSV	RSV	RSV	INTID4	INTID3	INTID2	INTID1	INTID0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
Default	—	—	0	0	0	0	0	0	—	—	—	0	0	0	0	0

Table 3–57. Interrupt ID Register (INTIDR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–14	RSV	Reserved bits. Not used.
13–8	INTFN5–INTFN0	Number of unprocessed interrupt services
7–5	RSV	Reserved bits. Not used.
4–0	INTID4–INTID0	ID of the pending interrupt with the highest priority (lowest ID)

Table 3–58. Interrupts RAW (INTRAW) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRAW1	IRAW0
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0

Table 3–59. Interrupts RAW (INTRAW) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–2	RSV	Reserved bits. Not used.
1	IRAW1	Sets whether or not masked interrupts are counted in INTFN5–INTFN0 count value of INTIDR register. 0: Not counted 1: Counted
0	IRAE0	Sets whether or not masked interrupts are reflected in INTID5–INTID0 ID value of INTIDR register. 0: Not reflected 1: Reflected

3.4.4.3 External Interrupts

Since interrupts 7–0 are shared with pins GIO 7–0, they must be set as interrupts in the registers on the GIO side. The interrupt control registers samples any new interrupt on the rising edge of the clk_arm clock. The interrupt signal must be at least two clk_arm cycles long to be recognized.

3.4.5 Bus Controller

3.4.5.1 Description

The ARM bus controller serves as the interface between the address/data bus of the ARM7TDMI core and the rest of the DSC24 system. These bus controller functions are transparent to the user. However, there are some discrete registers in the bus controller that provide bit and byte reversal functions as well as a read-only register containing the DSC24 device revision number. This section gives information on these registers and describes the way the bus controller interfaces with peripherals and the external memory controller.

3.4.5.2 Registers

The bus control register (BUSC) has four registers associated with it.

Table 3–60. Data Access Mode

OFFSET	ADDRESS	REGISTER NAME	
0x0	0x0003:0900	ECR	Endian converter register
0x1	0x0003:0902	EBYTER	Endian byte converted register
0x2	0x0003:0904	EBITR	Endian bit converted register
0x3	0x0003:0906	REVR	Device revision number

3.4.5.3 Memory Map

The MCU can access all peripheral registers, RAM, and ROM area. The bus controller is responsible to automatically generate the number of wait states required according to the peripheral module accessed by the MCU. A description of each of the memory blocks and their respective characteristics are given in the *MCU Subsystem Memory* section.

3.4.5.4 Access Modes

In the MCU subsystem, words, half words, or bytes may be transferred between the processor and the memory. The size of the transaction taking place is determined by the internal MAS[1:0] signals. These are encoded as follows:

Table 3–61. Data Access Mode

MAS [1:0]	DATA MODE	BIT WIDTH
00	Byte	8
01	Half word	16
10	Word	32
11	Reserved	—

The type of memory access used (8-, 16-, or 32-bit access) depends on the assembly instruction used.

The ARM7 processor always produces a byte address, but instructions are either words (4 bytes) or half words (2 bytes), and data can be any size. Note that when word instructions are fetched from memory, address bits A[1:0] are undefined and when half word instructions are fetched, A[0] is undefined.

When a data read of byte or half word size is performed (e.g., LDRB instruction), the memory system may safely ignore the fact that the request is for a sub-word sized quantity and present the whole word. The ARM7 core always correctly extracts the addressed byte or half word from the data.

When a byte or half word write occurs (e.g., STRH instruction), the ARM7 core broadcasts the byte or half word across the whole of the bus. The memory decodes A[1:0] to enable writing only to the addressed byte or half word.

See the *TMS470 Assembly Language Tools User's Guide*, literature number SPNU118 for more details concerning the access modes.

3.4.5.5 Handshaking Wait-State Control and Wait-State Generation

When the MCU accesses some of the memories, additional wait cycles are required. Additional wait sates are added through a handshaking protocol.

When the MCU requests a half word or a word data access to the memories, a wait state is inserted automatically in the access cycle.

The USB peripheral registers, the HPI bridge interface, and some external memory accesses cannot be achieved within one cycle of ARM clock. For these accesses, an internal signal informs the MCU core to stretch indefinitely the wait states to allow access to slow peripherals or memories.

The requirement of handshaking access means that each time one of the block mentioned above is accessed, one wait state is implicitly inserted automatically at the beginning of the access.

3.4.5.6 Endian Conversion

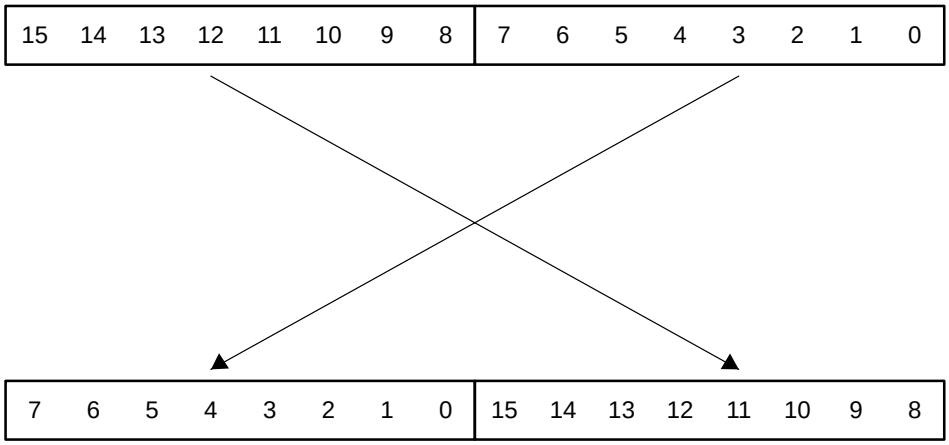
The user can modify the endianess of a piece of data by using the endian convert register (ECR).

Writing data to be converted into the ECR register once, allows the user to obtain either data that is converted by byte unit and by bit unit.

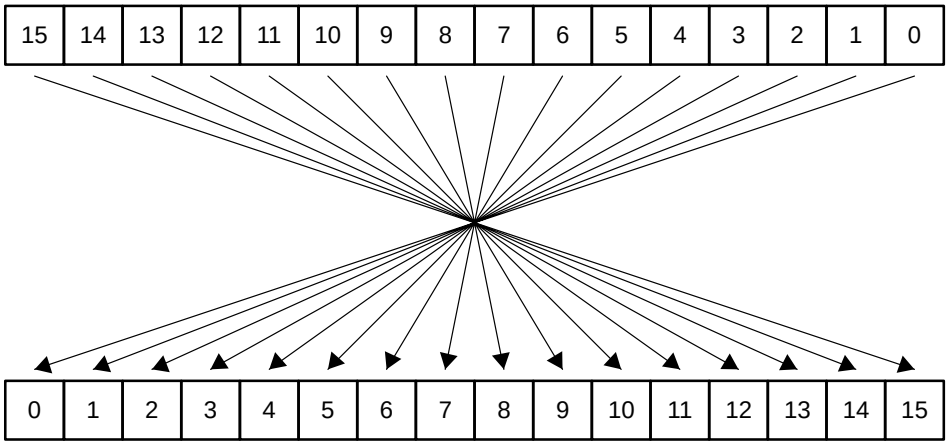
The converted data is stored to the endian byte converted register (EBYTER) and endian bit converted register (EBITR).

Data is stored within the MCU memory in a little endian format.

The following is a representation of the endian conversion.



(a) Byte Conversion



(b) Bit Conversion

Figure 3–12. Endian Conversion

Table 3–62. Endian Convert (ECR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ECR15	ECR14	ECR13	ECR12	ECR11	ECR10	ECR9	ECR8	ECR7	ECR6	ECR5	ECR4	ECR3	ECR2	ECR1	ECR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–63. Endian Convert (ECR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	ECR15–ECR0	Endian conversion register. Converts endian of written data. Results are read from the EBYTER register and EBITR register.

Table 3–64. Endian Byte Reversed (EBYTER) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EBY15	EBY14	EBY13	EBY12	EBY11	EBY10	EBY9	EBY8	EBY7	EBY6	EBY5	EBY4	EBY3	EBY2	EBY1	EBY0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–65. Endian Byte Reversed (EBYTER) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	EBY15–EBY0	Reads data written in the ECR register as data in which the upper byte and lower byte are swapped.

Table 3–66. Endian Bit Reversed (EBITR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EBI15	EBI14	EBI13	EBI12	EBI11	EBI10	EBI9	EBI8	EBI7	EBI6	EBI5	EBI4	EBI3	EBI2	EBI1	EBI0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–67. Endian Bit Reversed (EBITR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	EBI15–EBI0	Reads each bit of data written in the ECR register as data in which the upper byte and lower byte are swapped

3.4.5.7 Device Revision Number

One of the registers implemented within the ARM BUS controller is a read only register that allows the user to know the silicon revision number of the device.

Table 3–68. Device Revision (REVR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	REV7	REV6	REV5	REV4	REV3	REV2	REV1	REV0
R/W	—	—	—	—	—	—	—	—	R	R	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Table 3–69. Device Revision (REVR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved
7–0	REV7–REV0	Represents device version

The revision number is represented with one decimal digit. For example, REV=0x10 corresponds to silicon version 1.0.

3.5 I/O Peripherals

3.5.1 GIO

The DSC24 has a maximum of 32 general-purpose parallel ports. Using the GIO ports, the logic levels of the ports of an externally assigned device are read and can be controlled. These pins are used to control various external devices such as AD, DA, or motor.

The addresses of the 11 registers that control the GIO ports are given in Table 3–70. In the DSC24, 32 GIO pins can be used in device operating mode 3. In other operating modes, only 21 pins from GIO0 to GIO20 can be used. In that case, the control bits of unused pins are reserved.

Table 3–70. GIO Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0600	DIR0	GIO port direction 0
0x01	0x0003:0602	DIR1	GIO port direction 1
0x02	0x0003:0604	INV0	GIO port inversion 0
0x03	0x0003:0606	INV1	GIO port inversion 1
0x04	0x0003:0608	BITSET0	GIO bit set 0
0x05	0x0003:060A	BITSET1	GIO bit set 1
0x06	0x0003:060C	BITCLR0	GIO bit clear 0
0x07	0x0003:060E	BITCLR1	GIO bit clear 1
0x08	0x0003:0610	IRQPORT	GIO interrupt port
0x09	0x0003:0612	FSEL	Function select
0x0A	0x0003:0614	BITRATE	Bit rate

3.5.1.1 Input/Output Settings

The GIO ports can each be individually set to input or output ports by the DIR0 and DIR1 registers. If set to an input port, the MCU knows the status of the GIO port by reading the register BITSET0 and BITSET1 in the GIO module. Also, the INV0 and INV1 registers allow the user to reverse the polarity of input.

Table 3–71. GIO Port Direction 0 (DIR0) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DIR15	DIR14	DIR13	DIR12	DIR11	DIR10	DIR9	DIR8	DIR7	DIR6	DIR5	DIR4	DIR3	DIR2	DIR1	DIR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 3–72. GIO Port Direction 0 (DIR0) Register Bit/Field Description

BIT	REGISTER NAME	DESCRIPTION
15–0	DIR15–DIR00	Port direct bits Sets the input/output direction of each port. DIR15 sets pin GIO15 and DIR00 sets pin GIO0. 0: Output 1: Input

Table 3–73. GIO Port Direction 1 (DIR1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DIR31	DIR30	DIR29	DIR28	DIR27	DIR26	DIR25	DIR24	DIR23	DIR22	DIR21	DIR20	DIR19	DIR18	DIR17	DIR16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 3–74. GIO Port Direction 1 (DIR1) Register Bit/Field Description

BIT	REGISTER NAME	DESCRIPTION
15–0	DIR31–DIR16	Port direct bits Sets the input/output direction of each port. DIR31 sets pin GIO31 and DIR16 sets pin GIO16. 0: Output 1: Input

Table 3–75. GIO Port Inversion 0 (INV0) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	INV15	INV14	INV13	INV12	INV11	INV10	INV9	INV8	INV7	INV6	INV5	INV4	INV3	INV2	INV1	INV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–76. GIO Port Inversion 0 (INV0) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	INV15–INV00	Port inversion bits Inverts the polarity of each input port. INV15 sets pin GIO15 and INV00 sets pin GIO0. 0: Uninverted 1: Inverted

Table 3–77. GIO Port Inversion 1 (INV1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	INV31	INV30	INV29	INV28	INV27	INV26	INV25	INV24	INV23	INV22	INV21	INV20	INV19	INV18	INV17	INV16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–78. GIO Port Inversion 1 (INV1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	INV31–INV16	Port inversion bits Inverts the polarity of each input port. INV31 sets pin GIO31 and INV16 sets pin GIO16. 0: Uninverted 1: Inverted

In the GIO module, the output is controlled using a set register and a clear register (BITSET0, BITSET1, BITCLR0, and BITCLR1). When a GIO port is set to output, 1 is output when 1 is written in the relevant bit in the set register, and 0 is output when 1 is written in the relevant bit of the clear register. It is possible to control only the bits the user wishes to control without affecting the output status of other GIO ports. When configured as output, the GIO ports are not affected by the INV0 and INV1 registers, their action is ignored.

Table 3–79. GIO Bit Set 0 (BITSET0) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–80. GIO Bit Set 0 (BITSET0) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	BS15–BS00	GIO bit set 0 When the GIO pin has been set to output and a 1 is written in the bit, the equivalent GIO pin becomes 1. BS15 is equivalent to GIO15 and BS00 is equivalent to GIO0. When the GIO pin has been set to input, the logic level of the GIO pin is read from the equivalent bit. If set to inverted by INV0 or INV1, if GIO is set to input, the inverted value is read.

Table 3–81. GIO Bit Set 1 (BITSET1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BS31	BS30	BS29	BS28	BS27	BS26	BS25	BS24	BS23	BS22	BS21	BS20	BS19	BS18	BS17	BS16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–82. GIO Bit Set 1 (BITSET1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	BS31–BS16	GIO bit set 1 Same function as the GIO bit set to 0. BS31 is equivalent to GIO31 and BS16 is equivalent to GIO16.

Table 3–83. GIO Bit Clear 0 (BITCLR0) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–84. GIO Bit Clear 0 (BITCLR0) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	BC15–BC00	GIO bit clear If GIO has been set to output and a 1 is written in the equivalent bit, the GIO pin is cleared to 0. BC15 is equivalent to pin GIO15 and BC00 is equivalent to pin GIO0. When read, the current port status can be read, similar to the BITSET0 register.

Table 3–85. GIO Bit Clear 1 (BITCLR1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BC31	BC30	BC29	BC28	BC27	BC26	BC25	BC24	BC23	BC22	BC21	BC20	BC19	BC18	BC17	BC16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–86. GIO Bit Clear 1 (BITCLR1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	BC31–BC16	GIO bit clear If GIO has been set to output and a 1 is written in the equivalent bit, the GIO pin is cleared to 0. BC31 is equivalent to pin GIO31 and BC16 is equivalent to pin GIO16. When read, the current port status can be read, similar to the BITSET1 register.

3.5.1.2 External Interrupt Settings

A GIO port can be used as an external interrupt port to the MCU of the DSC24 from an external device. GIO0 through GIO7 can be used as external interrupt ports. A maximum of eight external interrupts can be specified by settings in registers in the GIO module. These interrupt signals are supplied to the interrupt controller. Either rise or fall is selected as the interrupt trigger, but the pulse width must be at least two clocks of the DSC24 internal ARM clock.

Prior to using a GIO port as an external interrupt, the appropriate bit in the IRQPORT register must be set. It is also necessary to configure the GIO port as input. If the INV0x bit in the INV0 register is set to 1, the interrupt is triggered by a rising edge of the external signal; if INV0x bit is set to 0, the detection is done on a falling edge.

When configured as external interrupts, the corresponding bits in the registers BITCLR and BITSET are still updated with the pin status.

Table 3–87. GIO Interrupt Port (IRQPORT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRP7	IRP6	IRP5	IRP4	IRP3	IRP2	IRP1	IRP0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–88. GIO Interrupt Port (IRQPORT) Register Bit/Field Description

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7–0	IRP7–IRP0	Interrupt port. GIO7–GIO0 are used as external interrupt ports. IRP7 sets GIO7 and IRP0 sets GIO0. 0: Ordinary GIO 1: External interrupt To use as an external interrupt, set the input/output setting of the DIR0 register to input. Interrupt detects pulse of at least one cycle of the ARM clock of 0 when input is uninverted by the INV register; interrupt detects pulse of at least one cycle of the ARM clock of 1 when input is inverted by the INV register.

3.5.1.3 GIO Selection Settings

The GIO ports can be used as dedicated function ports rather than general-purpose input/output ports. Dedicated ports are selected by setting the proper bits in the FSEL and IRQPORT registers. Table 3–89 shows the GIO ports and signals that can be selected.

After a reset, all GIO ports are defined as uninverted general-purpose inputs.

Table 3–89. GIO Port Selection Output

PIN NAME	SEL.BIT	I/O (Secondary Function)	Function Selected	
			FSELx=0 or IRPx=0 (Primary)	FSELx=1 or IRPx=1 (Secondary)
GIO00/INT0	IRP0	I	GIO #00	External interrupt #0
GIO01/INT1	IRP1	I	GIO #01	External interrupt #1
GIO02/INT2	IRP2	I	GIO #02	External interrupt #2
GIO03/INT3	IRP3	I	GIO #03	External interrupt #3
GIO04/INT4	IRP4	I	GIO #04	External interrupt #4
GIO05/INT5	IRP5	I	GIO #05	External interrupt #5
GIO06/INT6	IRP6	I	GIO #06	External interrupt #6
GIO07/INT7	IRP7	I	GIO #07	External interrupt #7
GIO08/DSP_BFSR1	FSEL0	I	GIO #08	DSP frame sync pulse input
GIO09/DSP_BCLKR1	FSEL1	I	GIO #09	DSP serial transfer clock input
GIO10/DSP_BDR1	FSEL2	I	GIO #10	DSP serial data input
GIO11/DSP_BFSX1	FSEL3	I/O	GIO #11	DSP frame sync pulse
GIO12/DSP_BCLKX1	FSEL4	I/O	GIO #12	DSP serial transfer clock
GIO13/DSP_BDX1	FSEL5	O	GIO #13	DSP serial data output
GIO14/DSP_XF	FSEL6	O	GIO #14	DSP external flag output
GIO15/ATTACH	FSEL7	I	GIO #15	USB Attach detection
GIO16/CLKOUT1	FSEL9:8	O	GIO #16	Clock output #1
GIO17/CLKOUT2	FSEL11:10	O	GIO #17	Clock output #2
GIO18/SDEN0	FSEL12	O	GIO #18	Transfer enable signal of serial I/F #0
GIO19/SDEN1	FSEL13	O	GIO #19	Transfer enable signal of serial I/F #1
GIO20/MIRQ	FSEL14	O	GIO #20	MCU interrupt output. Same signal as the IRQ output of the interrupt controller. Active low level output

The GIO16 and GIO17 ports can be used as clock output. The clock frequencies that can be output are the clk_usb (M48XI) clock or its half, the clk_xi (MXI) clock or its half, and a divided value of the ARM clock. The clock output frequencies values are defined by the FSEL and BITRATE registers.

Table 3–90. Function Select (FSEL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	FSEL14	FSEL13	FSEL12	FSEL11	FSEL10	FSEL9	FSEL8	FSEL7	FSEL6	FSEL5	FSEL4	FSEL3	FSEL2	FSEL1	FSEL0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–91. Function Select (FSEL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved bit. Not used.
14	FSEL14	Pin GIO20 function setting 0: GIO20 1: MIRQ
13	FSEL13	Pin GIO19 function setting 0: GIO19 1: SDEN2
12	FSEL12	Pin GIO18 function setting 0: GIO18 1: SDEN1
11–10	FSEL11 FSEL10	Pin GIO17 function setting FSEL11 FSEL10 0 0 GIO17 0 1 GIO divided clock output 2 1 0 M48XI clock output 1 1 M48XI clock output divided by 2 When FSEL11:10 = 01, the frequency of the clock output is determined by the BITRATE register.
9–8	FSEL9 FSEL8	Pin GIO16 function setting FSEL9 FSEL8 0 0 GIO16 0 1 GIO divided clock output 1 1 0 MXI clock output 1 1 MXI clock output divided by 2 When FSEL9:8 = 01, the frequency of the clock output is determined by the BITRATE register.
7	FSEL7	Pin GIO15 function setting 0: GIO15 1: ATTACH
6	FSEL6	Pin GIO14 function setting 0: GIO14 1: XF
5	FSEL5	Pin GIO13 function setting 0: GIO13 1: BDX1
4	FSEL4	Pin GIO12 function setting 0: GIO12 1: BCLKX1
3	FSEL3	Pin GIO11 function setting 0: GIO11 1: FSX1
2	FSEL2	Pin GIO10 function setting 0: GIO10 1: BDR1
1	FSEL1	Pin GIO9 function setting 0: GIO9 1: BCLKR1
0	FSEL0	Pin GIO8 function setting 0: GIO8 1: BFSR1

Table 3–92. Bit Rate (BITRATE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BR27	BR26	BR25	BR24	BR23	BR22	BR21	BR20	BR17	BR16	BR15	BR14	BR13	BR12	BR11	BR10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–93. Bit Rate (BITRATE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	BR27–BR20	Transfer rate setting bits Sets the division ratio of clock output of FSEL11–FSEL10. Determines the transfer rate when pin GIO17 is set to GIO divided clock output 2. Bit rate GIO = ARM clock/ ((register value + 1) x 2)
7–0	BR17–BR10	Transfer rate setting bits Sets the division ratio of clock output of FSEL9–FSEL8. Determines the transfer rate when pin GIO16 is set to GIO divided clock output 1. Bit rate = ARM clock/ ((register value + 1) x 2)

3.5.2 Serial Port Interface

Depending on the device operating mode selected, the DSC24 has a two-channel (device operating modes 0, 2, or 3) or a 3-channel (device operating mode 1) clock synchronized serial port interface (SPI). The transfer unit is 8 bits each time, and it is a simple three-line serial consisting of a clock line, output data line, and input data line. It has a DMA transfer function with SDRAM for transfer of large quantities of data. The SPI always supplies the serial clock. The SPI can only be used in a master mode.

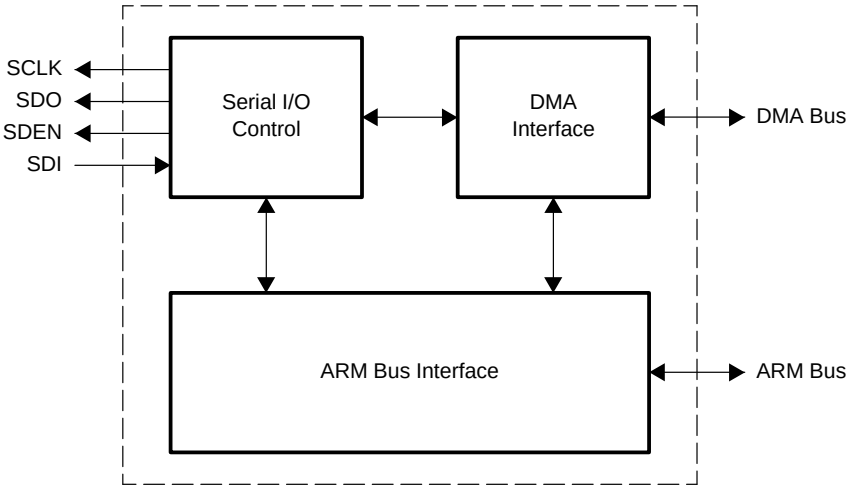


Figure 3–13. Serial Port Interface

Nine registers control each of the serial ports and its DMA. Their respective locations within the MCU memory map are given in Table 3–94.

Table 3–94. Serial Port Registers

OFFSET	ADDRESS FOR SERIAL PORT 0	ADDRESS FOR SERIAL PORT 1	ADDRESS FOR SERIAL PORT 2 (MODE 1 ONLY)	REGISTER NAME	
0x00	0x0003:0200	0x0003:0280	0x0003:0300	TXDATAx	Transmit data
0x01	0x0003:0202	0x0003:0282	0x0003:0302	RXDATAx	Receive data
0x02	0x0003:0204	0x0003:0284	0x0003:0304	SIOENx	Serial interface output enable
0x03	0x0003:0206	0x0003:0286	0x0003:0306	SIOMODEx	Serial interface output mode
0x04	0x0003:0208	0x0003:0288	0x0003:0308	DMATRGx	DMA trigger
0x05	0x0003:020A	0x0003:028A	0x0003:030A	DMAMODEx	DMA transfer mode
0x06	0x0003:020C	0x0003:028C	0x0003:030C	DMASTAx_LO	DMA transfer start address (lower 16 bits)
0x07	0x0003:020E	0x0003:028E	0x0003:030E	DMASTAx_HI	DMA transfer start address (upper 11 bits)
0x08	0x0003:0210	0x0003:0290	0x0003:0310	DMASTATx	DMA status

3.5.2.1 Transmission and Reception Buffers

Each serial port has an 8-bit buffer on each of the transmitting and receiving sides. The data set in the transmission buffer is output in synchronization with the falling edge of SCLK.

Figure 3–14 shows the 3-line serial interface-timing chart. When data is written into the transmission buffer (8 bits), serial data is output from the SDO terminal. Since data is loaded into the reception buffer (8 bits) from the SDI terminal at the same time as that, some value must be written into the transmission buffer in the case of data reception. Only SCLK/SDO/SDI have dedicated terminals pin of the serial ports 0 and 1. The SDEN (transfer enable output) signals for these two serial ports are multiplexed with some general-purpose input output pins (GIO18/GIO19).

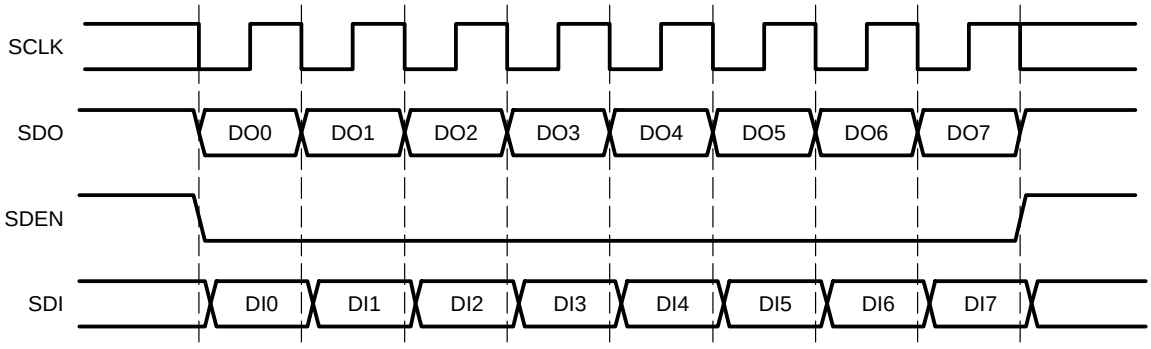


Figure 3–14. Data Transmission Diagram

These buffers correspond to the TXDATA and RXDATA registers. The XMIT bit of the RXDATA register is used by the system to indicate if a transmission or reception is in progress or not. When writing into the transmit data buffer, the transfer flag XMIT is automatically asserted. The transfer flag is cleared automatically at the end of the transmission or reception.

The user must check the status of XMIT before writing into the transmit register. If a transmission is in progress, the piece of data the user writes into TXDATA is ignored and lost. When a reception is in progress (XMIT=1), the data read from RXDATA is always the previous one. Any new piece of data that arrives into the serial port updates the RXDATA register. If the previous data has not been read before, it is lost.

Table 3–95. Transmit Data (TXDATAx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	TXDx7	TXDx6	TXDx5	TXDx4	TXDx3	TXDx2	TXDx1	TXDx0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–96. Transmit Data (TXDATAx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7:0	TXDx7–TXDx0	Transmission data register When data is set in this register the transmission/reception is started. To write into this register, bit 10 of the SIOENx register must be set to 1. When read, the written value is read.

Table 3–97. Receive Data (RXDATAx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	XMITx	RXDx7	RXDx6	RXDx5	RXDx4	RXDx3	RXDx2	RXDx1	RXDx0
R/W	—	—	—	—	—	—	—	—	R	R	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–98. Receive Data (RXDATAx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	XMITx	Transmission/reception flag bit 0: Transmission/reception complete 1: Transmission/reception in progress
7:0	RXDx7–RXDx0	Reception data register. When data is set in the TXDATAx register and transmission/reception is complete, the reception data is read.

3.5.2.2 Baud Rate

The maximum speed that can be achieved with any of the serial ports is half of the current ARM clock. The serial interface output mode register (SIOMODE) lets the user choose the bit rate they want.

The bit rate is defined with the following formula.

Bit rate = (current ARM operation clock) / ((set value + 1) x 2)

Table 3–99. Serial Interface Output Mode (SIOMODEx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	SCLKMx	MSB	RSV	BRx7	BRx6	BRx5	BRx4	BRx3	BRx2	BRx1	BRx0
R/W	—	—	—	—	—	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	—	0	0	0	0	0	0	0	0

Table 3–100. Serial Interface Output Mode (SIOMODEx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10	SCLKMx	Serial clock mode. Sets polarity of clock when there is no transfer. 0: No-transfer clock level 1 1: No-transfer clock level 0
9	MSBx	LSB first/MSB first bit 0: LSB first 1: MSB first
8	RSV	Reserved bit. Not used.
7:0	BRx7–BRx0	Transfer rate setting bits. The bit rate is: Bit rate = (current ARM operation clock) / ((set value + 1) x 2)

3.5.2.3 Serial Port Modes

The serial clock mode bit (SCLKM) of the SIOMODE register is used to set the polarity of the clock pin when the serial port is in idle mode. By default, this bit is set to 0 and the SCLK clock signal stays high when no data transfer is in progress.

The SIOEN bit is used to enable or disable the serial port. When this bit is set to 0 (default value), the serial port is disabled. If a transmission or reception is in progress, writing to this bit takes effect only after the current transfer has been completed.

When the serial port is disabled, any new data written into the TXDATA register does not initiate any new transmission, any new data received is ignored and the RXDATA register keeps is previous value. When in this mode, the pins SDO and SDEN output the logic level 1. The status of the SCLK pin is dependent on the SCLKM bit.

The MSB control bit of the SIOMODE register allows the user to choose between the LSB and the MSB of the data being placed first. This bit affects the transmission and the reception.

Table 3–101. Serial Interface Output Enable (SIOENx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	SIOENx
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–102. Serial Interface Output Enable (SIOENx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	SIOENx	Transmission permission bit. When set to 1, transmission is permitted. When prohibited, data cannot be written into the TXDATAx register. 0: Transmission prohibited 1: Transmission permitted

3.5.2.4 Interrupts

When a transmission or reception is completed, an interrupt can be sent to the ARM. The interrupts IRQ8, IRQ9, and IRQ10 are connected to the serial port 0, serial port 1, and serial port 2 respectively. The interrupt is enabled or disabled by setting registers in the interrupt controller.

3.5.2.5 Transfer Using the Direct Memory Access

The serial I/F module is connected to SDRAM by a dedicated bus. DMA data transfer to and from external devices is possible without going through the MCU. The MCU executes the DMA transfer by controlling the DMA controller in the serial I/F module. The SDRAM transfer start address, total number of transferred bytes, and the transfer direction are set. When DMA transfer is complete, an interrupt is generated to ARM (IRQ8, IRQ9, or IRQ10 depending on the serial port used). The number of transfer bytes from SDRAM has a minimum unit of 4 bytes. The number of bytes transmitted over the serial port line has to be a multiple of 2. For the DMA transfer to be enabled, both the bit 0 of SIOEN register in serial interface controller and SDMAS bit of the SDMODE register in SDRAM controller must be set to 1. Endian conversion is possible at the time of DMA transfer. There are three conversion modes, as shown in Figure 3–15 and Figure 3–16. The endian conversion is configured with the PRM1:0 bits.

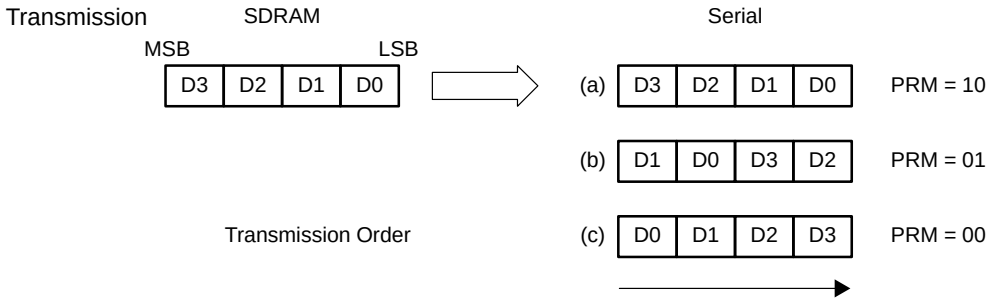


Figure 3–15. Endian Conversion During Transmission

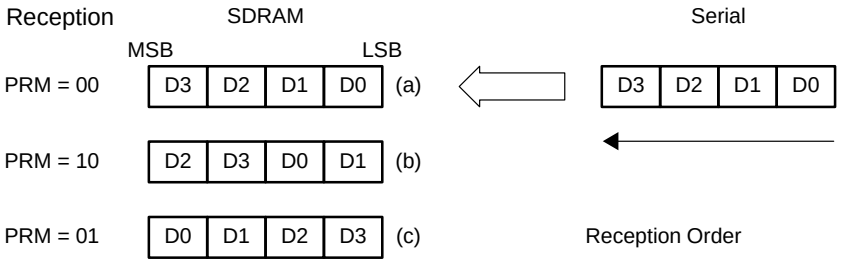


Figure 3–16. Endian Conversion During Reception

Only an even number of bytes can be transmitted. The start address in SDRAM has to be aligned on an even address. If the start address or number of bytes to be accessed in the SDRAM is not a multiple of four, the serial transmission or reception is modified accordingly. Table 3–103 and Table 3–104 shows how the bytes are handled in the different situations.

Table 3–103. SDRAM Access During Transmission

PERM1:0	SDRAM START ADDRESS	NUMBER OF BYTES	SERIAL TRANSMISSION ORDER
00	4n	4m	D0 D1 D2 D3
	4n	4m+2	D0 D1
	4n+2	4m	–
	4n+2	4m+2	D2 D3
01	4n	4m	D3 D2 D1 D0
	4n	4m+2	D1 D0
	4n+2	4m	–
	4n+2	4m+2	D3 D2
10	4n	4m	D1 D0 D3 D2
	4n	4m+2	D1 D0
	4n+2	4m	–
	4n+2	4m+2	D3 D2

Data order in SDRAM is D3 D2 D1 D0 (MSB=D3, LSB=D0)

Table 3–104. SDRAM Access During Reception

PERM1:0	SDRAM START ADDRESS	NUMBER OF BYTES	DATA SAVED IN DMA
00	4n	4m	D3 D2 D1 D0
	4n	4m+2	D1 D0 D1 D0
	4n+2	4m	–
	4n+2	4m+2	D1 D0 D1 D0
01	4n	4m	D0 D1 D2 D3
	4n	4m+2	D0 D1 D0 D1
	4n+2	4m	–
	4n+2	4m+2	D0 D1 D0 D1
10	4n	4m	D2 D3 D0 D1
	4n	4m+2	D0 D1 D0 D1
	4n+2	4m	–
	4n+2	4m+2	D0 D1 D0 D1

Data order during reception is D0 D1 D2 D3

3.5.2.6 DMA Control Registers

Table 3–105. DMA Trigger (DMATRGx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BRKx	TRGx
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0

Table 3–106. DMA Trigger (DMATRGx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–2	RSV	Reserved bits. Not used.
1	BRKx	DMA transfer forced termination bit When 1 is set during DMA transfer, the DMA transfer is forcibly terminated and this bit is automatically cleared to 0. Writing 0 has no meaning. If set at the same time as TRGx during transfer, BRKx is valid.
0	TRGx	DMA transfer start bit When set to 1, the DMA transfer is started. After transfer ends, this bit is automatically cleared to 0. Writing 0 has no meaning. If 1 is set during transfer, it has no meaning. When there is no transfer, if set at the same time as BRKx, TRGx is invalid.

Table 3–107. DMA Transfer Mode (DMAMODEx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	DMAx	PRMx1	PRMx0	DIRx	TXSZx10	TXSZx9	TXSZx8	TXSZx7	TXSZx6	TXSZx5	TXSZx4	TXSZx3	TXSZx2	TXSZx1	TXSZx0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–108. DMA Transfer Mode (DMAMODEx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved bit. Not used.
14	DMAMx	DMA mode Set to 1 when DMA is used. When this bit is 1 and DMAPx is 1, the DMA transfer is permitted.
13–12	PRMx1–PRMx0	DMA transfer byte order See the *Transfer Using The Direct Memory Access* section.
11	DIRx	DMA transfer direction 0: SDRAM from serial interface 1: Serial interface from SDRAM
10–0	TXSZx10–TXSZx0	Number of transfer bytes Only even numbers can be specified. (TXSZx0=0)

Table 3–109. Transfer Start Address (Lower 16 Bits) (DMASTAx_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	STADx15	STADx14	STADx13	STADx12	STADx11	STADx10	STADx9	STADx8	STADx7	STADx6	STADx5	STADx4	STADx3	STADx2	STADx1	STADx0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–110. DMA Transfer Start Address (Lower 16 Bits) (DMASTAx_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	STADx15–STADx0	Lower 16 bits of DMA start address of SDRAM Only even numbers can be specified. (STADx0=0)

Table 3–111. DMA Transfer Start Address (Upper 11 Bits) (DMASTAx_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	STADx26	STADx25	STADx24	STADx23	STADx22	STADx21	STADx20	STADx19	STAD18	STADx17	STADx16
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–112. DMA Transfer Start Address (Upper 11 Bits) (DMASTAx_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	STADx26–STADx16	Upper 16 bits of DMA start address of SDRAM

Table 3–113. DMA Status (DMASTATx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RUNx	DMAPx	RSV	RSV	RSV	RESTx10	RESTx9	RESTx8	RESTx7	RESTx6	RESTx5	RESTx4	RESTx3	RESTx2	RESTx1	RESTx0
R/W	R	R	—	—	—	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–114. DMA Status (DMASTATx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RUNx	DMA transfer flag 0: Transfer end 1: Transfer in progress
14	DMAPx	DMA permission status 0: DMA transfer prohibited 1: DMA transfer permitted
13–11	RSV	Reserved bits. Not used.
10–0	RESTx10–RESTx0	Indicates remaining number of DMA transfer bytes

3.5.3 DMA Transfer Procedure

Below is an example of the DMA transfer procedure.

The serial interface is set to DMA mode, furthermore the transfer start address of the transfer original SDRAM and the number of transfer bytes are set.

The DMA transfer starts by setting the start bit (TRGx) in the register DMATRGx.

With the DMA transfer starts, the first piece of data is read from SDRAM 4 bytes at a time. The data is written in the SPI transfer register 1 byte at a time. The data transmitted is synchronized with the falling edge of the serial port clock.

The next 4 bytes are read from the SDRAM and transmitted.

When the requested amount of bytes to be sent has been reached, an interruption occurs.

Because the DMA transfer request occurs every 4 bytes, the serial transfer can look discontinued. See Figure 3–17.

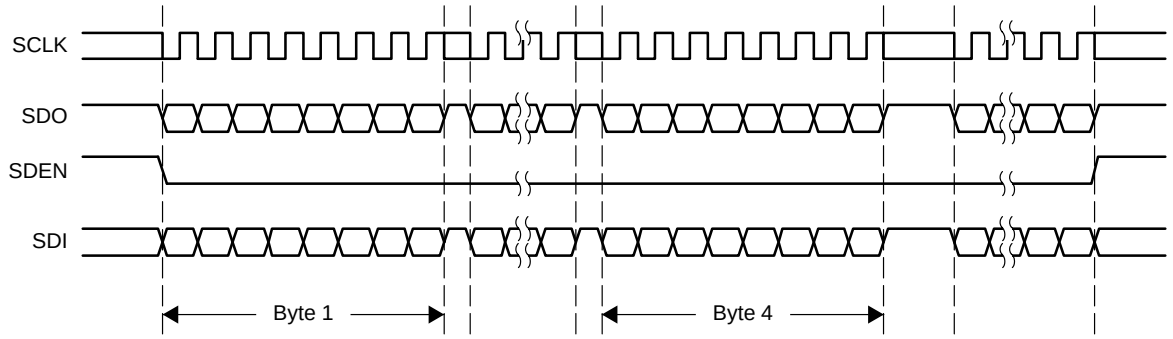


Figure 3–17. Timing Chart at Time of DMA Transfer

Each serial I/F module has its own DMA controller, but since only one module can access the SDRAM at a time, more than one DMA transfer cannot occur at a time. If several serial interfaces request a DMA at the same time, the one with the smallest instance number gets the priority. If a DMA transfer has been preempted by another one from the serial interface module, it is necessary to re-enable the DMA channel by writing a 1 into the DMAMx bit of the DMAMODE register.

Before starting a DMA transfer, make sure the DMA controller of the serial port is connected to the SDRAM controller. The bit SDMAS (bit 13) of the SDMODE register must be set to 1.

3.5.4 UART

The DSC24 supports a two-channel two-line (RX, TX) start-stop synchronization serial port (UART). Between the two channels, one channel (UART0) supports a full set of UART communications when the device is used in mode 0 or mode 3. In this case, five lines are available (RX, TX, CTS, DSR, RTS).

A 32-byte FIFO is built in for each of the transmitting sides and receiving sides and interrupts are generated due to causes such as errors or various states of the FIFO. Data is 8 or 7 bits. Even, odd, or no-parity bits can be selected, and 1- or 2-stop bits can be selected. Transmission/reception timing uses the ARM clock. The maximum baud rate achievable is 1/16 of the ARM clock (in bps).

The UART module detects parity errors and overrun errors. Interrupt to the ARM is generated when any of the following events occur.

- Amount of data in receiving-side FIFO exceeds set value
- Amount of data in transmitting-side FIFO exceeds set value
- Word received has an error
- Change occurs in DSR or CTS input when UART0 in mode 0 or mode 3 is used
- The time-out period is reached

The ARM controls the UART module by accessing the following internal registers.

- Data transmitting-side/receiving-side registers (FIFO)
- Bit rate register
- Mode register
- Receiving-side FIFO control register
- Transmitting-side FIFO control register
- Line control register
- Status register

Table 3–115 gives their respective addresses.

Table 3–115. UART Control Registers

OFFSET	REGISTER ADDRESS		REGISTER NAME	
	UART0	UART1		
0x00	0x0003:0380	0x0003:0400	DTRRx	Data transmission/reception
0x01	0x0003:0382	0x0003:0402	BRSRx	Bit rate set
0x02	0x0003:0384	0x0003:0404	MSRx	Mode set
0x03	0x0003:0386	0x0003:0406	RFCRx	Reception FIFO control
0x04	0x0003:0388	0x0003:0408	TFCRx	Transmission FIFO control
0x05	0x0003:038A	0x0003:040A	LCRx	Line control
0x06	0x0003:038C	0x0003:040C	SRx	Status register

The BRSRx register determines the speed of the UART port. The UART is clocked by the internal signal CLK_ARM divided by 16.

$$\text{UART speed} = \frac{\text{ARM\_CLK}}{16(\text{BRSR value} + 1)}$$

Table 3–116. Bit Rate Set (BRSRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	BRx15	BRx14	BRx13	BRx12	BRx11	BRx10	BRx9	BRx8	BRx7	BRx6	BRx5	BRx4	BRx3	BRx2	BRx1	BRx0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–117. Bit Rate Set (BRSRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	BRx15–BRx0	Bit rate setting register Sets bit rate of data transfer. See the formula above.

Each UART has two FIFOs, one 32-byte for transmit and one 32-byte for receive. However, there is only one register to access them. The DTRRx register is used to access the transmit and receive FIFOs. When this register is read, the oldest data stored in reception FIFO is returned. When the user writes into it, the data is stored into the transmission FIFO register and then transferred. The user must make sure the transmit FIFO is not full before writing into this register.

When reading this register, bits 12 to 8 indicate if the word read is valid or not. The data is valid only when all these bits are equal to 0.

Table 3–118. Data Transmission/Reception (DTRRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RVFx	BFx	FEx	ORFx	PEFx	DTRx7	DTRx6	DTRx5	DTRx4	DTRx3	DTRx2	DTRx1	DTRx0
R/W	—	—	—	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–119. Data Transmission/Reception (DTRRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–13	RSV	Reserved bits. Not used.
12	RVFx	Received word valid flag Indicates whether the received word that was read is valid or invalid. 0: Received word invalid 1: Received word valid
11	BFx	Break flag 0: No break 1: Break is detected
10	Fex	Framing error 0: No framing error 1: Framing error
9	ORFx	Overrun flag 0: No overrun 1: Overrun
8	PEFx	Parity error flag 0: No parity error 1: Parity error
7–0	DTRx7–DTRx0	Transmission/reception data register When reading, the oldest data stored in the reception FIFO is read. The status of this data can be read at the same time in each bit. When writing into this register, data is stored in the transmission FIFO and data is transferred. When writing, it must be confirmed that FIFO is not full before writing is performed.

The configuration of the UART is defined by the mode set register (MSRx). The PSBx and PEBx bits select the parity used during a communication. When enabled, one bit is appended at the end of the data. The CLSx bit selects the size of the word transmitted or received. When only 7 bits are used, the DTRx7 bit of the DTRRx register is ignored during a write and always reads 0 during a read.

The RFTIEx bit is selected if an interrupt is generated when the reception FIFO is above the trigger level.

The TFTIEx bit enables an interrupt to be generated when the transmission FIFO is below trigger level.

The REIEx bit enables the interrupt generated when a received word that has parity error, overrun, frame error, or a break is present in the reception FIFO.

The LSIEx bit is available only for UART0 in mode 0 and mode 3. This bit enables the interrupt generated when a change occurs on the DSR or CTS input pin (when CTS transmission control is off).

The TOICx1:0 bits are used to configure the time-out interrupt. An interrupt is generated in status where at least one word of received data is stored in the reception FIFO, if new data has not been transmitted even when the set time

has elapsed. This interrupt is used to provide a trigger of reception data loading to the MCU when, at the end of data transfer, the reception data up to the reception FIFO trigger level has not been stored in the reception FIFO.

The CTS transmission control bit (CTSCx) enables or disables the checking of the CTS signal. This bit is only valid for UART0 in mode 0 and 3. When the CTS transmission control is disabled, the UART ignores the CTS input signal.

The RTS reception control bit refers to the trigger level of the reception FIFO and enables RTS on/off control function. This bit is valid only for UART0 in mode 0 and 3.

Table 3–120. Mode Set (MSRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RFTIEx	TFTIEx	REIEx	LSIEx	TOICx1	TOICx0	RSV	RSV	RSV	CTSCx	RTSCx	PEBx	PSBx	SBLSx	RSV	CLSx
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—	—	R/W	R/W	R/W	R/W	R/W	—	R/W
Default	0	0	0	0	0	0	—	—	—	0	0	0	0	0	—	0

Table 3–121. Mode Set (MSRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RFTIEx	Reception FIFO trigger interrupt 0: Disabled 1: Enabled
14	TFTIEx	Transmission FIFO trigger interrupt 0: Disabled 1: Enabled
13	REIEx	Reception data error interrupt enable 0: Disabled 1: Enabled
12	LSIEx / RSV	If UART0 in mode 0 or 3 this bit is: Line status change interrupt enable 0: Disabled 1: Enabled Otherwise, it is a reserved bit.
11–10	TOICx1–TOICx0	Time out interrupt enable TOICx1 TOICx0 0 0 Time out interrupt disabled 0 1 3 word time 1 0 7 word time 1 1 15 word time
9–7	RSV	Reserved bits. Not used.
6	CTSCx / RSV	If UART0 in mode 0 or 3 this bit is: CTS transmission control. 0: Disabled 1: Enabled Otherwise, it is a reserved bit.
5	RTSCx / RSV	If UART0 in mode 0 or 3 this bit is: RTS reception FIFO control. 0: Disabled (RTS pin stays in a high level) 1: Enabled Otherwise, it is a reserved bit.
4	PEBx	Parity enable bit 0: Disabled (no parity) 1: Enabled
3	PSBx	Parity select bit 0: Even 1: Odd
2	SBLSx	Stop bit length select 0: 1 bit 1: 2 bits
1	RSV	Reserved bit. Not used.
0	CLSx	Character length select 0: 8 bits 1: 7 bits

Table 3–122. Reception FIFO Control (RFCRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RFCBx	RDEFx	RSV	RSV	RSV	RTLx2	RTLx1	RTLx0	RSV	RSV	RWx5	RWx4	RWx3	RWx2	RWx1	RWx0
R/W	R/W	R/W	—	—	—	R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	—	—	—	0	0	0	—	—	0	0	0	0	0	0

Table 3–123. Reception FIFO Control (RFCRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RFCBx	Reception FIFO clear bit When 1 is written the reception FIFO is cleared. When clear is finished, the bit returns to 0.
14	RDEFx	Reception data error detection flag If there is data in which a reception error occurred in the reception FIFO, it is set to 1. When there is no longer any data in which a reception error occurred in the reception FIFO, it is cleared to 0. Same bit as bit 9 of SRx register. When 1 is written into this bit, the reception block is reset (FIFO cleared, RDEF bit becomes 0), and after reset is finished it is reset to 0.
13–11	RSV	Reserved bits. Not used.
10–8	RTLx2–RTLx0	Reception FIFO trigger level setting Sets trigger level of reception FIFO trigger interrupt. RTLx2 RTLx1 RTLx0 1 1 1 Reserved 1 1 0 Reserved 1 0 1 32 1 0 0 24 0 1 1 16 0 1 0 8 0 0 1 4 0 0 0 1
7–6	RSV	Reserved bits. Not used.
5–0	RW5–RW0	Received word counter Indicates number of received words stored in the reception FIFO.

Table 3–124. Transmission FIFO Control (TFCRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	TFCBx	RSV	RSV	RSV	RSV	TTLx2	TTLx1	TTLx0	RSV	RSV	TWx5	TWx4	TWx3	TWx2	TWx1	TWx0
R/W	R/W	—	—	—	—	R/W	R/W	R/W	—	—	R	R	R	R	R	R
Default	0	—	—	—	—	0	0	0	—	—	0	0	0	0	0	0

Table 3–125. Transmission FIFO Control (TFCRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	TFCBx	Transmission FIFO clear bit When set to 1 the transmission FIFO is cleared. After clear is finished, it returns to 0.
14–11	RSV	Reserved bits. Not used.
10–8	TTLx2–TTLx0	Transmission FIFO trigger level setting Sets trigger level of the transmission FIFO trigger interrupt. RTLx2 RTLx1 RTLx0 1 1 1 Reserved 1 1 0 Reserved 1 0 1 32 1 0 0 24 0 1 1 16 0 1 0 8 0 0 1 4 0 0 0 1
7–6	RSV	Reserved bits. Not used.
5–0	TW5–TW0	Transmitted word counter Indicates number of received words stored in the transmission FIFO.

Table 3–126. Line Control (LCRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	UTSTx	RSV	RSV	RSV	RSV	RSV	BOCx	DSRx	RSV	RSV	CTSx	RSV	RTSx	RSV	RSV
R/W	—	R/W	—	—	—	—	—	R/W	R	—	—	R	—	R/W	—	—
Default	—	0	—	—	—	—	—	0	0	—		0	—	0	—	—

Table 3–127. Line Control (LCRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved bit. Not used.
14	UTSTx	UART test bit This bit needs to be set to 0
13–9	RSV	Reserved bits. Not used.
8	BOCx	Break output control Controls TX output 0: Ordinary data 1: Forcibly sets TX to 0
7	DSRx	Data set ready Displays current DSR input pin value This bit is reserved for UART1 or UART0 in a 2-pin mode
6–5	RSV	Reserved bits. Not used.
4	CTSx	Clear to send Displays current CTS input pin value. This bit is reserved for UART1 or UART0 in a 2-pin mode.
3	RSV	Reserved bit. Not used.
2	RTSx	Request to send Set the status of the RTS output pin. When 1 is written into this register the RTS pin becomes low. This bit is reserved when a 2-pin UART is used.
1–0	RSV	Reserved bits. Not used.

Table 3–128. Status Register (SRx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DSRSx	RSV	RSV	CTSSx	TFTIx	RFTIx	RFERx	TOIFx	RSV	RSV	RSV	RSV	RSV	RFNFx	TFEFx	TREFx
R/W	—	—	—	—	R	R	R	R	—	—	—	—	—	R	R	R
Default	—	—	—	—	0	1	0	0	—	—	—	—	—	0	1	1

Table 3–129. Status Register (SRx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	DSRSx	Data set ready status Indicates that there has been a change in DSR input. When this register is read, it is cleared. 0: No change in DSR 1: Change in DSR
14–13	RSV	Reserved bits. Not used.
12	CTSSx	Clear to send status Indicates that there has been a change in CTS input. When this register is read, it is cleared. 0: No change in CTS 1: Change in CTS
11	RTFIx	Reception FIFO trigger indicator 0: Data in reception FIFO less than trigger level defined by RTLx2:0 1: Data in reception FIFO above trigger level
10	TFTIx	Transmission FIFO trigger indicator 0: Data in transmission FIFO above trigger level defined by TTLx2:0 1: Data in transmission FIFO less than trigger level
9	RFERx	Reception data error flag If there is data in the reception FIFO in which a reception error occurred, it is set to 1. When there is no longer any data in which a reception error occurred in the reception FIFO, it is cleared to 0. Same bit as bit 14 of the RFCRx register.
8	TOIFx	Time out interrupt flag 1 indicates that time out interrupt occurred. This register is cleared to 0 when read.
7–3	RSV	Reserved bits. Not used.
2	RFNFx	Reception FIFO not empty flag Indicates that there is at least one received word of data in reception FIFO. 0: No data in reception FIFO 1: Data in reception FIFO
1	TFEFx	Transmission FIFO empty flag Indicates that transmission FIFO is empty. 0: Transmission FIFO is not empty 1: Transmission FIFO is empty
0	TREF	Transmission register empty flag Set to 1 when both transmission FIFO and transmission shift out register are empty. 0: There is data in either transmission FIFO or shift out register 1: Transmission FIFO and shift out register are both empty

3.5.5 USB

This section describes the USB module that is found in the DSC24. The features of the USB module are as follows:

- Compliant with USB specification revision 1.1
- Involve UDC (USB device controller), the protocol engine by Phoenix
- Support only for full-speed device
- Built-in USB transceiver
- Built-in 2K RAM for endpoint FIFO
- Programmable MaxPktSize
- Understanding and decoding of standard requests except for GetDescriptor, SetDescriptor, and SynchFrame
- Support of device remote wake up feature

3.5.5.1 Block Diagram

Figure 3–18 shows the block diagram of the USB module. This module has a built-in transceiver and its own 2K bytes of RAM for the FIFOs. The module has a total of six FIFOs, each associated with an endpoint. The structure of each FIFO is the same except for the direction and the buffer size. The maximum buffer size is 256 bytes for EP0 and EP2 while it is 2048 bytes for EP1 and EP3. Each FIFO buffer is allocated onto a unified RAM. The user programs the base address and buffer size.

The MCU accesses FIFO read/write port registers to read from or write to FIFOs. There are no DMA capabilities implemented on this chip.

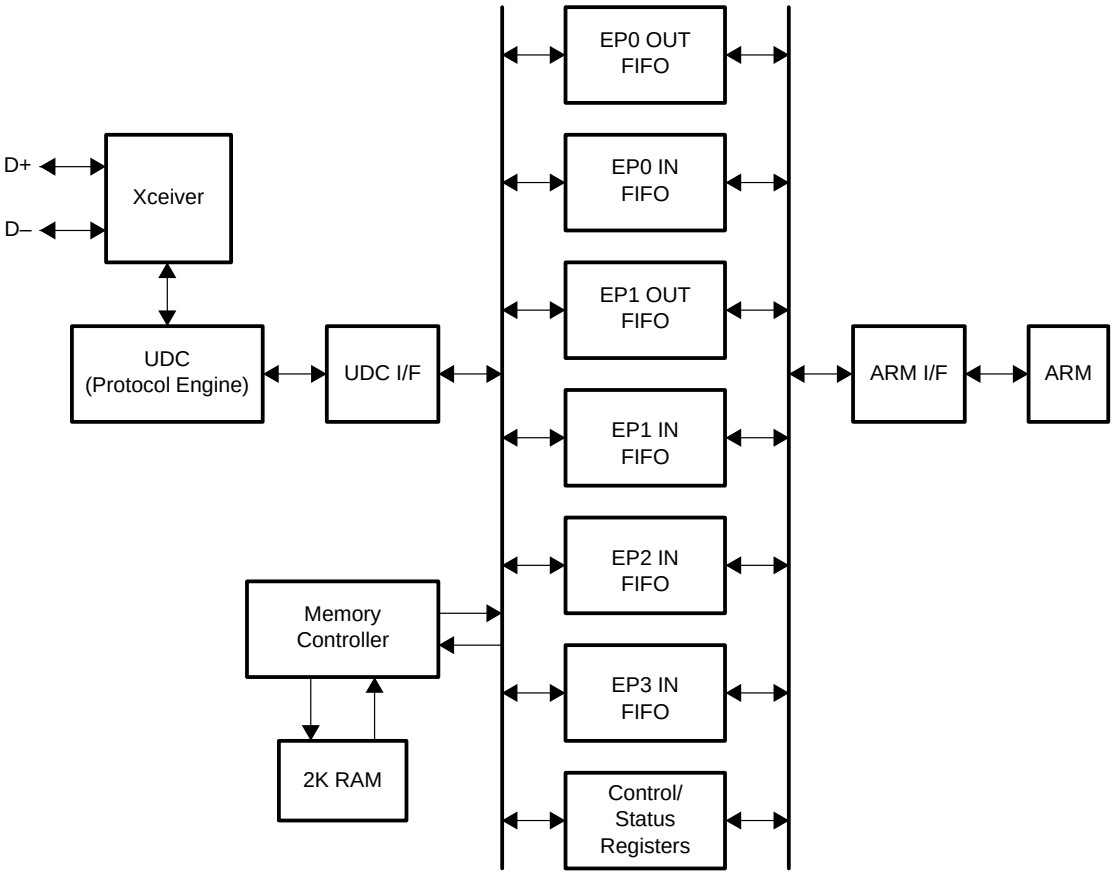


Figure 3–18. USB Module Block Diagram

3.5.5.2 Initialization Flow

The USB initialization must include the following points:

- Set the size of the FIFOs.
- Set the base address within the dedicated RAM of the FIFOs.
- Set the threshold value used to generate an interrupt.
- Enable the interrupt that will be used.
- Perform a soft reset by writing a 1 in the USBRST register.
- Initialize the USB core by writing a succession of configuration bytes into the USBCFG register.

When the MCU accesses a peripheral memory mapped register, it takes only 1 cycle. However, in the case of any USB register, it takes 2+n cycles for the MCU to read or write the register. The value n is variable and depends on the current access condition.

There are fifty registers associated with the USB. Table 3–130 shows their name and location within the memory.

Table 3–130. USB Control Registers

OFFSET	REGISTER ADDRESS	REGISTER NAME	
0x00	0x0003:0480	EP0OAD	Endpoint 0 out buffer address
0x01	0x0003:0482	EP0OSZ	Endpoint 0 out buffer size
0x02	0x0003:0484	EP0OCTL	Endpoint 0 out buffer control
0x03	0x0003:0486	EP0OIRQSZ	Endpoint 0 out interrupt size
0x05	0x0003:048A	EP0ORDT	Endpoint 0 out read data port
0x06	0x0003:048C	EP0OREST	Endpoint 0 out reminder
0x07	0x0003:048E	EP0OST	Endpoint 0 out status
0x08	0x0003:0490	EP0IAD	Endpoint 0 in buffer address
0x09	0x0003:0492	EP0ISZ	Endpoint 0 in buffer size
0x0A	0x0003:0494	EP0ICTL	Endpoint 0 in control
0x0B	0x0003:0496	EP0IIRQSZ	Endpoint 0 in interrupt size
0x0C	0x0003:0498	EP0IWDT	Endpoint 0 in write data port
0x0E	0x0003:049C	EP0IREST	Endpoint 0 in remainder
0x0F	0x0003:049E	EP0IST	Endpoint 0 in status
0x010	0x0003:04A0	EP1OAD	Endpoint 1 out buffer address
0x011	0x0003:04A2	EP1OSZ	Endpoint 1 out buffer size
0x012	0x0003:04A4	EP1OCTL	Endpoint 1 out control
0x013	0x0003:04A6	EP1OIRQSZ	Endpoint 1 out interrupt size
0x015	0x0003:04AA	EP1ORDT	Endpoint 1 out read data port
0x016	0x0003:04AC	EP1OREST	Endpoint 1 out remainder
0x017	0x0003:04AE	EP1OST	Endpoint 1 out status
0x018	0x0003:04B0	EP1IAD	Endpoint 1 in buffer address
0x019	0x0003:04B2	EP1ISZ	Endpoint 1 in buffer size
0x01A	0x0003:04B4	EP1ICTL	Endpoint 1 in control
0x01B	0x0003:04B6	EP1IIRQSZ	Endpoint 1 in interrupt size
0x01C	0x0003:04B8	EP1IWDT	Endpoint 1 in write data port
0x01E	0x0003:04BC	EP1IREST	Endpoint 1 in remainder
0x01F	0x0003:04BE	EP1IST	Endpoint 1 in status
0x020	0x0003:04C0	EP2IAD	Endpoint 2 in buffer address
0x021	0x0003:04C2	EP2ISZ	Endpoint 2 in buffer size
0x022	0x0003:04C4	EP2ICTL	Endpoint 2 in control
0x023	0x0003:04C6	EP2IIRQSZ	Endpoint 2 in interrupt size
0x024	0x0003:04C8	EP2IWDT	Endpoint 2 in write data port
0x026	0x0003:04CC	EP2IREST	Endpoint 2 in remainder
0x027	0x0003:04CE	EP2IST	Endpoint 2 in status
0x028	0x0003:04D0	EP3IAD	Endpoint 3 in buffer address
0x029	0x0003:04D2	EP3ISZ	Endpoint 3 in buffer size
0x02A	0x0003:04D4	EP3ICTL	Endpoint 3 in control
0x02B	0x0003:04D6	EP3IIRQSZ	Endpoint 3 in interrupt size
0x02C	0x0003:04D8	EP3IWDT	Endpoint 3 in write data port
0x02E	0x0003:04DC	EP3IREST	Endpoint 3 in remainder
0x02F	0x0003:04DE	EP3IST	Endpoint 3 in status
0x030	0x0003:04E0	USBRST	USB reset
0x031	0x0003:04E2	USBRSM	USB resume
0x032	0x0003:04E4	USBINTEN	USB interrupt enable
0x033	0x0003:04E6	USBCFC	USB configuration

Table 3–130. USB Control Registers (Continued)

OFFSET	REGISTER ADDRESS	REGISTER NAME	
0x034	0x0003:04E8	USBST	USB status
0x035	0x0003:04EA	USBINTST	USB interrupt status
0x036	0x0003:04EC	USBFRM	USB frame
0x037	0x0003:04EE	USBALT	USB alternate

All of these registers are described in the subsections below.

3.5.5.3 FIFO Size Selection

The size of each FIFO is set in the EPxxSZ register.

Table 3–131. EP0 and EP1 Out Buffer Size (EPxOSZ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	OxSZ10	OxSZ9	OxSZ8	OxSZ7	OxSZ6	OxSZ5	OxSZ4	OxSZ3	OxSZ2	OxSZ1	OxSZ0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–132. EP0 and EP1 Out Buffer Size (EPxOSZ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	OxSZ10–OxSZ0	EPx OUT buffer size Specify the buffer size of EPx OUT FIFO. The actual buffer size equals to this register value added 1. The maximum FIFO size of EP1 OUT is 2048; the one for EP0 OUT is 256. Bits 10 to 8 are ignored and should be set to zero for EP0.

Table 3–133. EP0, EP1, EP2, and EP3 In Buffer Size (EPxISZ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	IxSZ10	IxSZ9	IxSZ8	IxSZ7	IxSZ6	IxSZ5	IxSZ4	IxSZ3	IxSZ2	IxSZ1	IxSZ0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–134. EP0, EP1, EP2, and EP3 In Buffer Size (EPxISZ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	IxSZ10–IxSZ0	EPx IN buffer size Specify the buffer size of EPx IN FIFO. The actual buffer size equals to this register value added 1. The maximum FIFO size of EP1 IN and EP3 IN is 2048, the one for EP0 IN and EP2 IN is 256. Bits 10 to 8 are ignored and should be set to zero for EP0 and EP2.

3.5.5.4 FIFO Base Address Selection

The RAM base address of each FIFO buffer is specified in the EPxxAD register. This address must be set so as not to be duplicated among the FIFOs. Since the built-in RAM is 2K bytes, this address must be assigned within an area of 2K (0000h–07FFh).

Table 3–135. EP0 and EP1 Out Buffer Address (EPxOAD) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	OxAD10	OxAD9	OxAD8	OxAD7	OxAD6	OxAD5	OxAD4	OxAD3	OxAD2	OxAD1	OxAD0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–136. EP0 and EP1 Out Buffer Address (EPxOAD) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved
10–0	OxAD10–OxAD0	EPx OUT RAM base address Specify the base address of RAM allocated for the EPx OUT FIFO.

Table 3–137. EP0, EP1, EP2, and EP3 In Buffer Address (EPxIAD) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	IxAD10	IxAD9	IxAD8	IxAD7	IxAD6	IxAD5	IxAD4	IxAD3	IxAD2	IxAD1	IxAD0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–138. EP0, EP1, EP2, and EP3 In Buffer Address (EPxIAD) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved
10–0	IxAD10–IxAD0	EPx IN RAM base address Specify the base address of RAM allocated for the EPx IN FIFO.

3.5.5.5 Interrupt Threshold

The threshold value of the FIFO data counter at which interrupt occurs is set for each FIFO in the EPxxIRQSZ register.

In IN endpoint (except ISO) FIFOs, when transfer ends by ACK, interrupt occurs if the number of bytes of data in the FIFO is less than the value set here. In ISO endpoints, when an IN transaction ends, interrupt occurs if the number of bytes of data in the FIFO is less than or equal to the value set here.

In OUT endpoints, when an OUT transaction ends by ACK, interrupt occurs if the number of bytes of data in the FIFO is more than or equal to the value set here. If this value is set to 0, it is possible to tell if there was reception of a packet of any number of transfer bytes including a short/null packet.

Table 3–139. EP0 and EP1 Out Interrupt Size (EPxOIRQSZ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OxIRQ11	OxIRQ10	OxIRQ9	OxIRQ8	OxIRQ7	OxIRQ6	OxIRQ5	OxIRQ4	OxIRQ3	OxIRQ2	OxIRQ1	OxIRQ0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–140. EP0 and EP1 Out Interrupt Size (EPxOIRQSZ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved
11–0	OxIRQ11–OxIRQ0	EPx OUT threshold for Interrupt Specify the threshold value for the interrupt assertion. For EP0, bits 11 to 9 are ignored and should be set to 0.

Table 3–141. EP0, EP1, EP2, and EP3 IN Interrupt Size (EPxIIRQSZ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	IxIRQ11	IxIRQ10	IxIRQ9	IxIRQ8	IxIRQ7	IxIRQ6	IxIRQ5	IxIRQ4	IxIRQ3	IxIRQ2	IxIRQ1	IxIRQ0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–142. EP0, EP1, EP2, and EP3 IN Interrupt Size (EPxIIRQSZ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved
11–0	IxIRQ11–IxIRQ0	EPx IN threshold for interrupt Specify the threshold value for the interrupt assertion. For EP0 and EP2, bits 11 to 9 are ignored and should be set to 0.

3.5.5.6 Interrupt Control

To enable the interrupt, the USBINTEN register must be written.

Table 3–143. USB Interrupt Enable (USBINTEN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	MINCFIE	MINSPIE	MISOFIE	MISIIE	MISCIE	MIBRIE	MISETIE	MISUSIE	MIATTIE	MIEP3IIE	MIEP2IIE	MIEP1IIE	MIEP1OIE	MIEP0IIE	MIEP0OIE
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–144. USB Interrupt Enable (USBINTEN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved
14	MINCFIE	ClearFeature interrupt enable
13	MINSPIE	No SOF packet interrupt enable
12	MISOFIE	SOF interrupt enable
11	MISIIE	SetInterface interrupt enable
10	MISCIE	SetConfiguration interrupt enable
9	MIBRIE	USB bus reset interrupt enable
8	MISETIE	Setup interrupt enable
7	MISUSIE	Suspend interrupt enable
6	MIATTIE	Attach interrupt enable
5	MIEP3IIE	EP3 IN interrupt enable
4	MIEP2IIE	EP2 IN interrupt enable
3	MIEP1IIE	EP1 IN interrupt enable
2	MIEP1OIE	EP1 OUT interrupt enable
1	MIEP0IIE	EP0 IN interrupt enable
0	MIEP0OIE	EP0 OUT interrupt enable

To enable the interrupt, set each bit to one. To disable the interrupt, clear each bit to zero.

When an interrupt is generated by the USB, the source of it is determined by reading the USBINTST register. The interrupt flag is automatically cleared when this register is read.

The USB attach detect pin is multiplexed with the GIO15. See the *GIO* section for information concerning its configuration.

Table 3–145. USB Interrupt Status (USBINTST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	MINCFI	MINSPI	MISOFI	MISII	MISCI	MIBRI	MISETI	MISUSI	MIATTI	MIEP3II	MIEP2II	MIEP1II	MIEP1OI	MIEP0II	MIEP0OI
R/W	—	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–146. USB Interrupt Status (USBINTST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved
14	MINCFI	ClearFeature interrupt Interrupt asserted when ClearFeature (ENDPOINT_HALT) is received.
13	MINSPI	No SOF packet interrupt Interrupt asserted when the device does not receive the next SOF packet within 1 ms after a SOF packet is received.
12	MISOFI	SOF interrupt Interrupt asserted when a SOF packet is received.
11	MISII	SetInterface interrupt Interrupt asserted when SetInterface command is received. AlternateSetting set by HOST is read from the USBALT register.
10	MISCI	SetConfiguration interrupt Interrupt asserted when SetConfiguration command is received. Configuration set by HOST is read from bits 1–0 of the USBALT register.
9	MIBRI	USB bus reset interrupt Interrupt asserted when a USB bus reset is received.
8	MISETI	Setup interrupt Interrupt asserted when a SETUP transaction whose request is one of GetDescriptor, SetDescriptor and SynchFrame and the class/vendor requests is received. Also asserted when ClearFeature(HALT) request is received. Because SETUP data packet of the ClearFeature request is not stored in EP0 OUT FIFO, the user cannot know the receiving of ClearFeature explicitly. However, it can be known by no assertion of EP0O interrupt.
7	MISUSI	Suspend change interrupt Interrupt asserted when the USB bus is turned into the suspend state or is resumed (including the device remote wake up). The actual bus status can be known by reading bit 2 of USBST register. The USB enters the suspend state when it is idle for more than 6 ms.
6	MIATTI	Attach change interrupt Interrupt asserted when the cable is attached or detached. The actual cable status can be known by reading bit 3 of USBST register.
5	MIEP3II	EP3I interrupt Interrupt asserted when the number of bytes of data stored in FIFO is less than or equals to EP3IIRQSZ at the completion of sending a DATA0 packet.
4	MIEP2II	EP2I interrupt Interrupt asserted when the number of bytes of data stored in FIFO is less than or equals to EP2IIRQSZ at the completion of receiving of an ACK handshake.
3	MIEP1II	EP1I interrupt Interrupt asserted when the number of bytes of data stored in FIFO is less than or equals to EP1IIRQSZ at the completion of receiving of an ACK handshake.
2	MIEP1OI	EP1O interrupt Interrupt asserted when the number of bytes of data stored in FIFO is greater than or equals to EP1OIRQSZ or the received DATA packet is short or null at the completion of sending of an ACK handshake.
1	MIEP0II	EP0I interrupt Interrupt asserted when the number of bytes of data stored in FIFO is less than or equals to EP0IIRQSZ at the completion of receiving of an ACK handshake.
0	MIEP0OI	EP0O interrupt Interrupt asserted when the number of bytes of data stored in FIFO is greater than or equals to EP0OIRQSZ or the received DATA packet is short or null at the completion of sending of an ACK handshake.

3.5.5.7 Protocol Engine Reset

Writing a 1 into the MIRST bit of the USBRST register produces software reset.

Table 3–147. USB Reset (USBRST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	MIRST
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 3–148. USB Reset (USBRST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved
0	MIRST	Soft reset When set, the protocol engine (UDC) is reset. The USBST and USBINTST registers are also reset. This bit is automatically cleared. When the cable is attached, the user must set this bit.

3.5.5.8 Device Configuration

Figure 3–19 shows the configurations that the USB module supports. There are two configurations that are supported in this module. CfgA supports the control-bulk interrupt (CBI) transport defined in the mass storage class. CfgB is provided to support the configuration defined in the *Still Image Capture Device Definition* section.

The user configures the following properties at device initialization.

1. Whether or not to use each configuration
2. Each configuration number. A configuration number of 1 or 2 is assigned to CfgA and CfgB. If either is not used, set the unused configuration to 0, and set the used configuration to 1.
3. Whether or not to use each endpoint for each AlternateSetting. The device does not respond to access from the host to an endpoint that is set to unused.
4. MaxPktSize of each endpoint for each AlternateSetting

Writing the initialization data to the USBCFG register causes the above device initialization and must be done whenever the cable is attached. The format of the actual initialization data is described in the next subsection.

The user cannot change the number of interfaces, the interface value, and the maximum number of AlternateSettings of each interface, endpoint number, and endpoint type.

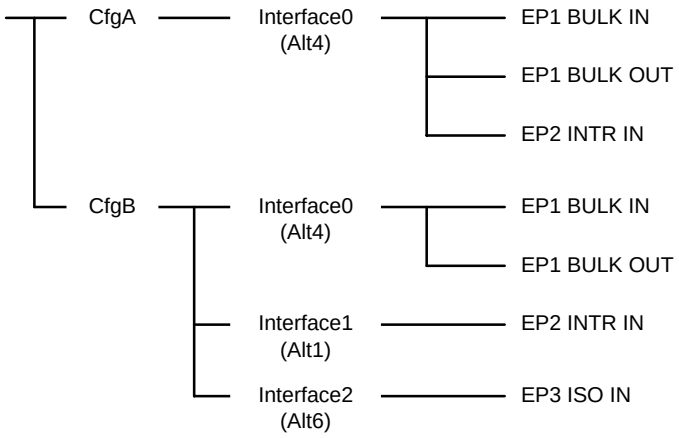


Figure 3–19. Device Configuration

Table 3–149. Summary of Possible Configurations

EP #	DIR	TYPE	MAX FIFO SIZE
EP0	IN/OUT	CONTROL	256 bytes
EP1	IN/OUT	BULK	2048 bytes
EP2	IN	INTR	256 bytes
EP3	IN	ISO	2048 bytes

Each endpoint has a dedicated FIFO for IN and for OUT and the endpoint is accessed from the MCU via these FIFOs. The main USB module has 2K bytes of RAM built in and each FIFO has a dedicated buffer within this memory area. The user sets the RAM base address and buffer size for each FIFO in the memory size. For EP3 IN, a maximum FIFO size of 2048 can be set (other FIFOs are a maximum of 256).

3.5.5.9 Device Initialization

The length of initialization data is 20 bytes. The user must write the 20 bytes from #0 to #19 into the USBCFG register.

Table 3–150. USB Configuration (USBCFC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	MICFC15	UMICFC14	MICFC13	MICFG12	MICFG11	MICFG10	MICFG9	MICFG8	MICFG7	MICFG6	MICFG5	MICFG4	MICFG3	MICFG2	MICFG1	MICFG0
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–151. USB Configuration (USBCFC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	MICFC15–MICFC0	Device initialization data write port After the cable is attached and the device is reset by USBRST, the initialization data must be written through this register. The format of the initialization data is explained below. Writing this register while the cable is detached has no effects.

The bytes that need to be written at initialization are described below in chronological order.

Table 3–152. Initialization Bytes Order

BYTE NUMBER	BYTE NAME	FUNCTION
#0	Configuration and endpoint0	Associate configuration number with CfgA and CfgB. Configures endpoint 0
#1	Intf0–Alt0–EP1	Setting of alternate setting 0 of endpoint 1 in configuration CfgA
#2	Intf0–Alt1–EP1	Setting of alternate setting 1 of endpoint 1 in configuration CfgA
#3	Intf0–Alt2–EP1	Setting of alternate setting 2 of endpoint 1 in configuration CfgA
#4	Intf0–Alt3–EP1	Setting of alternate setting 3 of endpoint 1 in configuration CfgA
#5	Intf0–Alt0–EP2	Setting of alternate setting 0 of endpoint 2 in configuration CfgA
#6	Intf0–Alt1–EP2	Setting of alternate setting 1 of endpoint 2 in configuration CfgA
#7	Intf0–Alt2–EP2	Setting of alternate setting 2 of endpoint 2 in configuration CfgA
#8	Intf0–Alt3–EP2	Setting of alternate setting 3 of endpoint 2 in configuration CfgA
#9	Intf0–Alt0–EP1	Setting of alternate setting 0 of endpoint 1 in configuration CfgB
#10	Intf0–Alt1–EP1	Setting of alternate setting 1 of endpoint 1 in configuration CfgB
#11	Intf0–Alt2–EP1	Setting of alternate setting 2 of endpoint 1 in configuration CfgB
#12	Intf0–Alt3–EP1	Setting of alternate setting 3 of endpoint 1 in configuration CfgB
#13	Intf1–Alt0–EP2	Setting of alternate setting 0 of endpoint 2 in configuration CfgB
#14	Intf2–Alt0–EP3	Setting of alternate setting 0 of endpoint 1 in configuration CfgB
#15	Intf2–Alt1–EP3	Setting of alternate setting 1 of endpoint 1 in configuration CfgB
#16	Intf2–Alt2–EP3	Setting of alternate setting 2 of endpoint 1 in configuration CfgB
#17	Intf2–Alt3–EP3	Setting of alternate setting 3 of endpoint 1 in configuration CfgB
#18	Intf2–Alt4–EP3	Setting of alternate setting 4 of endpoint 1 in configuration CfgB
#19	Intf2–Alt5–EP3	Setting of alternate setting 5 of endpoint 1 in configuration CfgB

Table 3–153. Byte #0—Setting of Configuration and Endpoint0

BITS	DESCRIPTION
15–14	Configuration number of CfgA
13–12	Configuration number of CfgB
11–10	Reserved
9–0	MaxPktSize of EP0

The configuration number can be 00, 01, or 10. The value 00 indicates that the configuration is not used. The configuration numbers for CfgA and CfgB cannot be identical if not 00.

Table 3–154. Bytes 1 to 4—Setting of Endpoint1 in CfgA

BITS	DESCRIPTION
15	Enable for EP1 IN (1:enable, 0:disable)
14	Enable for EP1 OUT (1:enable, 0:disable)
13–10	Reserved
9–0	MaxPktSize of EP1

If the enable bit for an endpoint is set to 0, the functionality of the specified endpoint is disabled and the device does not respond to the transaction for the disabled endpoint.

Table 3–155. Bytes 5 to 8—Setting of Endpoint2 in CfgA

BITS	DESCRIPTION
15	Enable for EP2 IN (1:enable, 0:disable)
14–10	Reserved
9–0	MaxPktSize of EP2

Table 3–156. Bytes 9 to 12—Setting of Endpoint1 in CfgB

BITS	DESCRIPTION
15	Enable for EP1 IN (1:enable, 0:disable)
14	Enable for EP1 OUT (1:enable, 0:disable)
13–10	Reserved
9–0	MaxPktSize of EP1

Table 3–157. Byte 13—Setting of Endpoint2 in CfgB

BITS	DESCRIPTION
15	Enable for EP2 IN (1:enable, 0:disable)
14–10	Reserved
9–0	MaxPktSize of EP2

Table 3–158. Bytes 14 to 19—Setting of Endpoint3 in CfgB

BITS	DESCRIPTION
15	Enable for EP3 IN (1:enable, 0:disable)
14–10	Reserved
9–0	MaxPktSize of EP3

3.5.5.10 Power Management

The USB module requires a 48-MHz clock as the main clock. This clock and its divided clocks drive all of the internal logic. If the USB is not used in such a case that the cable is detached or the USB bus is suspended, the user may stop these clocks for power management. The USB module provides the clock control logic, which can stop most of the internal clocks except for detection of attach or suspend interrupt condition. The user stops the clocks by setting the bit 1 (MICKOF) of the USBRSM register. By clearing this bit, the clocks are resumed. The user does not have to recover the clocks manually when the cable is attached or the USB bus is resumed, because the clocks are resumed automatically in such cases.

The clock controller module in the DSC24 also provides clock control logic for power management. As for the USB module, one of the clocks that can be controlled in the clock controller is an original 48-MHz clock and another clock is an ARM clock for MCU interface. These clocks must not be disabled if the user wants the USB ready to use. If these clocks are disabled, the MCU cannot detect and attach an interrupt and never boots up the USB module.

The USBRSM register is also used to resume the bus when the USB is in the suspend mode. To save power in the idle state, the USB_DP and USB_DM pins should stay high and low respectively.

Table 3–159. USB Resume (USBRSM) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	MICKOF	MIRSM
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0

Table 3–160. USB Resume (USBRSM) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–2	RSV	Reserved
1	MICKOF	Clock off Internal clock off control. When set to one, the internal clock is disabled. To enable the clock, clear this bit. This clock control logic is for the case that the cable is detached or USB is in the suspend condition. Note that when the cable is attached or the device is resumed (including the remote wake up), this bit is automatically cleared, namely the clock is automatically enabled.
0	MIRSM	Resume Remote wakeup control register. When the USB bus is in the suspend condition, the device resumes the bus by setting this register to one. It is not until the DEVICE_REMOTE_WAKEUP feature is enabled by the SetFeature command from the HOST, that the remote wake up function is available. At power-on reset, this function is disabled and setting this bit has no effects. When the user sets this bit, do not set the MICKOF bit (bit 1) at the same time.

3.5.5.11 Data Transfer

The data received is retrieved from the in port FIFO by reading the EPxORDT register.

Table 3–161. EP0 and EP1 Out Read Data Port (EPxORDT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	OxRDT7	OxRDT6	OxRDT5	OxRDT4	OxRDT3	OxRDT2	OxRDT1	OxRDT0
R/W	—	—	—	—	—	—	—	—	R	R	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–162. EP0 and EP1 Out Read Data Port (EPxORDT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved
7–0	OxRDT7–ORDT0	EPx OUT read port Read port of EPx OUT FIFO. The data field of the received data packet is stored in the FIFO. This is a read only register.

The data to be sent to the HOST through the IN FIFO can be written to the EPxIWDT register.

Table 3–163. EP0, EP1, EP2, and EP3 In Write Data Port (EPxIWDT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IxWDT7	IxWDT6	IxWDT5	IxWDT4	IxWDT3	IxWDT2	IxWDT1	IxWDT0
R/W	—	—	—	—	—	—	—	—	W	W	W	W	W	W	W	W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 3–164. EP0, EP1, EP2, and EP3 In Write Data Port (EPxIWDT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved
7–0	IxWDT7–IxWDT0	EPx IN write port Write port of EP0 IN FIFO. The data field of the data packet to be sent to HOST should be stored in the FIFO by writing this register. This is a write only register.

The number of bytes available in each buffer can be known by reading the EPxxREST register.

Table 3–165. EP0 and EP1 Out Remainder (EPxOREST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OxRST11	OxRST10	OxRST9	OxRST8	OxRST7	OxRST6	OxRST5	OxRST4	OxRST3	OxRST2	OxRST1	OxRST0
R/W	—	—	—	—	R	R	R	R	R	R	R	R	R	R	R	R
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–166. EP0 and EP1 Out Remainder (EPxOREST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved
11–0	OxRST11–OxRST0	EPx OUT available byte size Indicate the available byte size for read from the EPx OUT FIFO. For EP0, bits 11 to 9 should be ignored.

Table 3–167. EP0, EP1, EP2, and EP3 In Remainder (EPxIREST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	IxRST11	IxRST10	IxRST9	IxRST8	IxRST7	IxRST6	IxRST5	IxRST4	IxRST3	IxRST2	IxRST1	IxRST0
R/W	—	—	—	—	R	R	R	R	R	R	R	R	R	R	R	R
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 3–168. EP0, EP1, EP2, and EP3 In Remainder (EPxIREST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved
11–0	IxRST11–IXRST0	EPx IN available byte size. The available byte size for write to EP0 IN FIFO for write. For EP0 and EP2, bits 11 to 9 should be ignored.

3.5.5.12 USB Status

A set of read only registers is used to know the actual status of the USB module. The MISUSTS bit of the USBST register is used to determine if a USB cable is present or not.

Table 3–169. USB Status (USBST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	MIRMTF	MISUSTS	MISPD	MISETUP	MIDID
R/W	—	—	—	—	—	—	—	—	—	—	—	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0

Table 3–170. USB Status (USBST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–5	RSV	Reserved
4	MIRMTF	RemoteWakeupFeature status. Set when the device receives SetFeature (DEVICE_REMOTE_WAKEUP) request. Clear when the device receives the ClearFeature (DEVICE_ REMOTE_WAKEUP) request.
3	MISUSTS	Attach/detach Cable status. Set when the cable is attached, cleared when detached. Since the attach interrupt is asserted when the event that the cable is detached or attached is occurred, the user should read this register to know the actually the cable status.
2	MISPD	Suspend Suspend status of USB bus. Set when the bus is in the suspend status. When the cable is detached or attached, this bit is reset.
1	MISETUP	Setup Setup status. Set when the received SETUP transaction requires decoding operation. When the successive IN or OUT token is received, this bit is automatically cleared. Since the device automatically processes the standard requests except from the GetDescriptor, SetDescriptor, and SynchFrame, this bit is not set for these requests. The data is not stored in EP0 OUT FIFO when the device receives these requests. This bit is reset when the cable is detached or attached.
0	MIDID	Device initialize done The flag for device initialization done. Set when 20 bytes of the initialization data are written to the USBCFG register. Clear at soft reset by USBRST and when the cable is detached or attached.

Table 3–171. USB Frame (USBFRM) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	MIFRN10	MIFRN9	MIFRN8	MIFRN7	MIFRN6	MIFRN5	MIFRN4	MIFRN3	MIFRN2	MIFRN1	MIFRN0
R/W	—	—	—	—	—	R	R	R	R	R	R	R	R	R	R	R
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 3–172. USB Frame (USBFRM) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved
10–0	MIFRN10–MIFRN0	USB frame number The latest frame number received from HOST. Reset at USB bus reset, a soft reset by USBRST, or when the cable is attached or detached.

The register described below indicates the current configuration and alternate setting value of each interface assigned by the HOST. Reset at the USB bus reset, a soft reset by USBRST, or when the cable is attached or detached and the alternate setting values (bits 14–12, bits 10–8, bits 6–4) are reset after receiving a SetConfiguration request.

Table 3–173. USB Alternate (USBALT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	MIASI22	MIASI21	MIASI20	RSV	MIASI12	MIASI11	MIASI10	RSV	MIASI02	MIASI01	MIASI00	RSV	RSV	MICONF1	MICONF0
R/W	—	R	R	R	—	R	R	R	—	R	R	R	—	—	R	R
Default	—	0	0	0	—	0	0	0	—	0	0	0	—	—	0	0

Table 3–174. USB Alternate (USBALT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	RSV	Reserved
14–12	MIASI22–MIASI20	AlternateSetting of Interface2 AlternateSetting value set to Interface2 by the SetInterface command.
11	RSV	Reserved
10–8	MIASI12–MIASI10	AlternateSetting of Interface1 AlternateSetting value set to Interface1 by the SetInterface command.
7	RSV	Reserved
6–4	MIASI02–MIASI00	AlternateSetting of Interface0 AlternateSetting value set to Interface0 by the SetInterface command.
3–2	RSV	Reserved
1–0	MICONF1– MICONF0	Configuration Current configuration number selected by HOST.

3.5.5.13 Endpoint Control

Table 3–175. Endpoint 0 Out Buffer Control (EP0OCTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	O0STL	O0RST	O0ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–176. Endpoint 0 Out Buffer Control (EP0OCTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved.
2	O0STL	STALL bit for EP0 OUT The USB specifies the protocol stall in addition to the function stall of Endpoint by the SetFeature command. The protocol stall is when the data packet of the received setup transaction is an illegal or unsupported command, the IN/OUT transaction of the successive data, or status stage is stalled by the device. When set to one, EP0 OUT is stalled. The device returns a STALL handshake to the HOST after an OUT token is received. Since this bit is automatically cleared when the device receives the next SETUP token, the user does not have to clear this bit. Note, do not set the STALL bit and the ENABLE bit simultaneously.
1	O0RST	RESET bit for EP0 OUT FIFO When set to one, the status and the data stored in the EP0 OUT FIFO are cleared. This bit is automatically cleared and cannot be read. Writing into this register does not clear EP0OAD, EP0OSZ, and EP0OIRQSZ. When a SETUP token is received, this bit is set automatically, and the FIFO is reset to receive the DATA packet without a failure.
0	O0ENA	ENABLE bit for EP0 OUT When this bit is set, the device returns an ACK handshake after receiving a data packet from host in OUT transactions to EP0. When this bit is cleared, the device receives but ignores the DATA packet, and returns a NAK handshake. This bit is cleared by hardware when the device sends an ACK handshake to the host (ENCLRMODE = 0). The number of data sent from the host in the transaction is less than MaxPktSize when the device sends an ACK handshake to the host (ENCLRMODE = 1). The enable bit auto-clearing scheme is selected by setting the ENCLRMODE register (bit 15 of EP0OST register). As for the setup transaction, after receiving a setup token, the device always receives a DATA0 packet and returns an ACK handshake regardless of this bit.

Table 3–177. Endpoint 1 Out Control (EP1OCTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	O1STL	O1RST	O1ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–178. Endpoint 1 Out Control (EP1OCTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved.
2	O1STL	Stall bit for EP1 OUT When set, the function stall condition for EP1 OUT can be applied. On receiving an out token with setting stall bit, the internal state of EP1 OUT changes to halt and the endpoint is stalled. The device receives and ignores the data packet and sends a stall handshake. Once the endpoint is stalled, the stall condition is not cleared until the HOST clears it by a ClearFeature (halt) command even if this bit is cleared. If this bit is set with setting enable bit when ClearFeature (halt) is received, both bits are automatically cleared at that time. Setting a stall bit without setting an enable bit does not clear them automatically.
1	O1RST	Reset bit for EP1 OUT FIFO When set to one, the status and the data stored in EP1 OUT FIFO are cleared. This register is automatically cleared and cannot be read. This register does not clear EP1OAD, EP1OSZ, and EP1OIRQSZ.
0	O1ENA	ENABLE bit for EP1 OUT When this bit is set, the device returns an ACK handshake after receiving the data packet from the host in the OUT transactions to EP1. When this bit is cleared, the device receives but ignores the data packet and returns a NAK handshake. This bit is cleared by hardware when the device sends an ACK handshake to the host (ENCLRMODE = 0). The number of data sent from the host in the transaction is less than MaxPktSize when the device sends an ACK handshake to the host (ENCLRMODE = 1). The ENABLE bit control scheme is selected by setting the ENCLRMODE register (bit 15 of EP1OCTL register).

Table 3–179. Endpoint 0 In Control (EP0ICTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	I0STL	I0RST	I0ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–180. Endpoint 0 In Control (EP0ICTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved
2	I0STL	Stall bit for EP0 IN When set, the protocol stall condition for EP0 IN is activated. When an IN token to EP0 is received, the device returns a stall handshake. This bit is automatically cleared after the device receives the next setup packet. Note that this stall bit and the enable bit, bit 0 of this EP0ICTL, must not be set simultaneously.
1	I0RST	Reset bit for EP0 IN FIFO. When set to one, the status and the data stored in EP0 OUT FIFO are cleared. This register is automatically cleared and cannot be read. This register does not clear EP0IAD, EP0ISZ, and EP0IIRQSZ.
0	I0ENA	Enable bit for EP0 IN When this bit is set, the device returns a DATA0/DATA1 packet with data in the FIFO after receiving an IN token to EP0. When this bit is cleared, the device returns a NAK handshake for IN token. This bit is automatically cleared by hardware when the device receives an ACK handshake from the host (ENCLRMODE = 0). The rest of the bytes in the FIFO, when the device receives an ACK handshake from the host, are less than MaxPktSize (ENCLRMODE = 1) The enable bit control scheme is selected by setting the ENCLRMODE register (bit 15 of EP0ICTL register). If a NAK handshake is received after sending the data packet, the data sent to the host is recovered. The FIFO status (EP0IST) and this bit are not changed from the status before the transaction.

Table 3–181. Endpoint 1 In Control (EP1ICTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	I1STL	I1RST	I1ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–182. Endpoint 1 In Control (EP1ICTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved
2	I1STL	Stall bit for EP1 IN When set, the function stall condition for EP1 IN can be applied. On receiving an IN token with setting stall bit, the internal state of the EP1 IN changes to halt and the EP1 IN is stalled. The device returns a stall handshake. Once the endpoint is stalled, the stall condition is not cleared until the host clears it by a ClearFeature (halt) command even if this bit is cleared. If this bit is set with setting enable bit when ClearFeature (halt) is received, both bits are automatically cleared at that time. Setting the stall bit without setting the enable bit does not clear them automatically.
1	I1RST	EP1 OUT FIFO RESET Same as bit 1 of EP0ICTL
0	I1ENA	EP1 OUT ENABLE Same as bit 0 of EP0ICTL

Table 3–183. Endpoint 2 In Control (EP2ICTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	I2STL	I2RST	I2ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–184. Endpoint 2 In Control (EP2ICTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved
2	I2STL	EP2 in stall Same as bit 2 of EP1ICTL
1	I2RST	EP2 in FIFO reset Same as bit 1 of EP1ICTL
0	I2ENA	EP2 in enable Same as bit 0 of EP1ICTL

Table 3–185. Endpoint 3 In Control (EP3ICTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	I3STL	I3RST	I3ENA
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 3–186. Endpoint 3 In Control (EP3ICTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved
2	I3STL	Stall bit for EP3 IN On receiving an IN token with setting stall bit, the internal state of the EP3 IN changes to halt and the endpoint is stalled. The device does not send any data packets to the host for IN tokens. Once the endpoint is stalled, the stall condition is not cleared until the host clears it by a ClearFeature (halt) command even if this bit is cleared. If this bit is set with setting enable bit when ClearFeature (halt) is received, both bits are automatically cleared at that time. Setting the stall bit without setting the enable bit does not clear them automatically.
1	I3RST	EP3 in FIFO reset Same as the bit 1 of EP0ICTL
0	I3ENA	Enable bit for EP3 IN When this bit is set, the device returns a DATA0 packet with the data in the FIFO after receiving an IN token to EP3. When this bit is cleared, the device returns a null DATA0 packet. This bit is not automatically cleared by hardware.

Table 3–187. EP0 and EP1 Out Status (EPxOST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OxEMDE	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	OxFUL	OxEMP
R/W	R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R	R
Default	0	—	—	—	—	—	—	—	—	—	—	—	—	—	0	1

Table 3–188. EP0 and EP1 Out Status (EPxOST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	OxEMDE	EPx out ENCLRMODE Automatic clear mode select bit. (See the description of bit 0 of the EP0OCTL register)
14–2	RSV	Reserved.
1	OxFUL	EPx out full Set when the FIFO is full. Cleared when the FIFO is not full.
0	OxEMP	EPx out empty Set when the FIFO is empty. Cleared when the FIFO is not empty.

Table 3–189. EP0, EP1, EP2, and EP3 In Status (EPxIST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IxEMDE	IxURFB	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IxFUL	IxEMP
R/W	R/W	R/W	—	—	—	—	—	—	—	—	—	—	—	—	R	R
Default	0	0	—	—	—	—	—	—	—	—	—	—	—	—	0	1

Table 3–190. EP0, EP1, EP2, and EP3 In Status (EPxIST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	IxEMDE	EPx IN ECMODE This bit does not exist for EP3; its location is reserved. Same as EPxOST.ECMODE
14	IxURFB	Rate feedback mode This bit only exist for EP2, its location is reserved for EP0, EP1, and EP3. When set to 1, the data toggle bits of EP2 are changed after each data packet without regard to the presence or the type of handshake packet. When 0, EP2 uses the normal data toggle sequence.
13–2	RSV	Reserved
1	IxFUL	EPx IN full
0	IxEMP	EPx IN empty

3.6 Power Saving Mode

The DSC24 has been designed as a low-power device. However, in some cases, it may be necessary to reduce the power consumption.

The following describes the different features that can be used to reduce the power consumption of their design. See the DSP subsystem chapter for information on the DSP power-down modes.

3.6.1 Various Power Saving Methods

The first method to reduce the power consumption of the MCU subsystem is to disable the clock of any module that is not used. This is achieved by changing the configuration of the MODx registers. The chapter on the clock controller gives more details on this procedure.

Also, the USB module of the DSC24 goes automatically into a power saving mode if no signal is present on its pins for a while. In order to reach that state, the USB D+ pin needs to be high, and the 48-MHz clock needs to be provided to the system.

If the USB port is not to be used, connecting the USB VDD, the D+, and the D– signals to the ground line (GND) manually disable it.

When using the device in the external CPU mode, switch off the ARM core, the ARM internal memory, and the MCU bus controller (CACORE and CAIM bits of MOD1 and CBUSC bit of MOD3) since they are not used.

3.6.2 CLKC and MODx Registers

The following sections give more details concerning some of the register bits described in the clock controller module.

3.6.2.1 OSC48 Bit of CLKC

Bit 15 (OSC48) of the clock controller register (CLKC) disables the oscillator connected to the 48-MHz input (MXI48) only. It does not affect MXI for the ARM clock. The PWDN pin does not need to be set high for this to work.

3.6.2.2 SLPMD Bit of CLKC

The SLPMD bit can be used to disable all the modules and stop all the PLLs with bypass. This bit does not modify any of the clock register values (MODx).

The user must input any clock to SYSCLK before the SLPMD bit is enabled. The user must also switch the MCU input clock from PLL output to SYSCLK by setting the CASEL bit to 0 before enabling the SLPMD bit if they do not wakeup the DSP by RESET.

The MCU is stopped by setting up this bit and all the modules that have their clock enable in the MODx registers can be awakened by pulling the INT0/GIO0 pin low.

After the PLLs are locked, switch from SYSCLK to the PLL output by setting up the CASEL bit to 1.

3.6.2.3 BPSMD Bit of CLKC

When the BPSMD register is set, the PLL is bypassed and stopped. Nevertheless, the device can still be used; the DSP and SDRAM controller still receive a clock equal to MXI or SYSCLK.

The BPSMD switch should be done when the DSP is in Idle2 mode and the SDRAM controller is disabled to avoid any problem.

If the ARM clock is provided by a PLL, the user needs to change the settings they use for the timers, serial port, or UART.

3.6.2.4 CACORE and CAIM Bits of MOD1

The CACORE or CAIM bits (#1 and 0) of the MOD1 register can be used to turn off the ARM core and its internal memory. However, only a hardware reset is able to wake them up.

3.6.2.5 CBUSC Bit of MOD3

The bus controller clock controlled by the CBUSC bit (bit # 7) in the MOD3 register cannot be stopped if the ARM core is supposed to access the ARM bus.

It is not recommended to clear this bit, as only a hardware reset is able to set it up again (to access this bit, the user needs to go through the bus controller). This bit should be set only when the external CPU mode is used.

3.6.3 PWDN Pin

When the PWDN pin is enabled (pulled up), the oscillators (MXI/O and M48XI/O) are stopped and the PLLs bypassed. When PWDN is disabled, the oscillators start even if a RESET is asserted. When the oscillators wake up, execute the sequence to wake up from the sleep mode.

Table 3–191. PWDN Pin

PWDN STATE	OSCILLATORS
Low	Running
High	Stopped

When PWDN pin is used to reduce power, it takes at least 8K cycles to recover because the PLLs needs some time to lock again. Also, the user cannot disable the oscillator if they need the PLLs.

3.6.4 Wake-Up Pin

The GIO0/INT0 pin is also used as a wake-up pin. Because this function is separated from the two others, there is no need to set any of the GIO0 or INT0 related control registers prior to using it.

The wake-up signal needs a falling edge followed by at least one low clock cycle to work.

When the clock is stopped and the device is awakended with GIO0 low for two cycles, no interrupt is generated.

When the clock is enabled, GIO0 needs two cycles low to generate an interrupt. However, GIO0 needs to be at least 10 cycles low to wake-up the device and generate an interrupt when the device is in a sleep mode.

When the device wakes up, the code restarts where it left.

3.6.5 Sleep Mode

The device is said to be in a sleep mode when all the module clocks are disabled and all the PLLs are stopped.

3.6.6 Sleep Sequence

To make the device reach that state, simply set the SLPMD in the clock controller register (CLKC). This sequence needs some additional control. The user must input any clock to SYSCLK before the SLPMD bit is enabled. Also, switch the MCU input clock from PLL output to SYSCLK by CASEL=0 before the SLPMD bit is enabled if the device has not been woke up by RESET.

3.6.7 Wake-Up Sequence From Sleep Mode

The device wakes up if a pulse is applied on the INT0/GIO0 pin. The pulse needs to be at least the width of one clock. When the device wakes up, wait $8192 \times N$ cycles for the PLLs to lock. After the PLLs are locked, switch from SYSCLK to the PLL output by switching the CASEL bit to1.

3.7 Power-Down Mode

To be in the power-down mode, in addition to the sleep mode, the device needs to have its oscillators stopped.

3.7.1 Power-Down Sequence

To go into power-down mode apply the following:

- Input any clock to SYSCLK
- Set CASEL=0 to switch the MCU input clock
- Set the SLPMD bit to stop the different internal clocks
- Pull up the PWDN pin to stop the PLLs and oscillators

3.7.2 Wake-Up Sequence From Power Down

To wake the device from a power-down state, the following needs to be done:

- Pull down the PWDN pin and wait for the oscillators output to be stable. Depending on the clock input quality, this might take at least 10 ARM clock cycles (same as a reset).
- Apply a pulse on the INT0/GIO0 pin. The pulse needs to be at least the width of one clock. The PLLs start.
- Wait 8192 $\times$ N cycles for the PLLs to lock.
- Set the CASEL bit to 1 to switch from the SYSCLK to PLL output.

4 Imaging Peripherals

This chapter describes the imaging peripherals that are found on the DSC24.

4.1 Introduction

This paragraph describes the imaging peripheral block of the DSC24. This block can be divided into three main elements:

- The video interfaced used to receive and send video data to and from the DSC24 chip.
- The burst codec used to increase the amount of data that can be stored within the SDRAM.
- The on-screen display module that allows the user to superpose some text and graphics over the video data.

Each one of these modules is described in the following sections.

4.2 Video Interface

4.2.1 Specifications and Modes of Operation

The video interface loads CbCr or CCD/CMOS raw data from an external device to the SDRAM, external memory, or burst codec engine of the DSC24. This module supplies HD and VD synchronization signals to the external timing generator or it synchronizes to the HD and VD signals provided by the external timing generator. Other functions of this block are digital clamping and decimation as well as digital video output in certain modes of operation.

This module supports sequential scanning and interlace scanning. A programmable horizontal and vertical culling pattern can be applied to incoming images. The maximum input pixel clock is 37.5 MHz. The maximum input image size is 4096 pixels by 4096 pixels.

The input format used is YCbCr data with an 8-, 12-, or 16-bit bus width. The input format also supports CCD raw data with a 10-bit resolution.

Depending on the device operating-mode used, the video interface characteristics differ. The chapter first details the difference between the modes and then gives details about each of the sub-blocks that make the video interface block.

4.2.1.1 Device in Operating Mode 0

When operating in mode 0, the device uses the video interface module to load CbCr or CCD/CMOS RAW data from an external camera module to SDRAM or external memory. This module supplies HD/VD to the external timing generator, or it synchronizes to the HD/VD provided from the external timing generator. In this mode of operation, an internal synchronization generator block generates the VDOUT and HDOUT synchronization signals for the video output.

In mode 0, a VCLK output is available. The frequency of this output is identical to the PCLK input frequency.

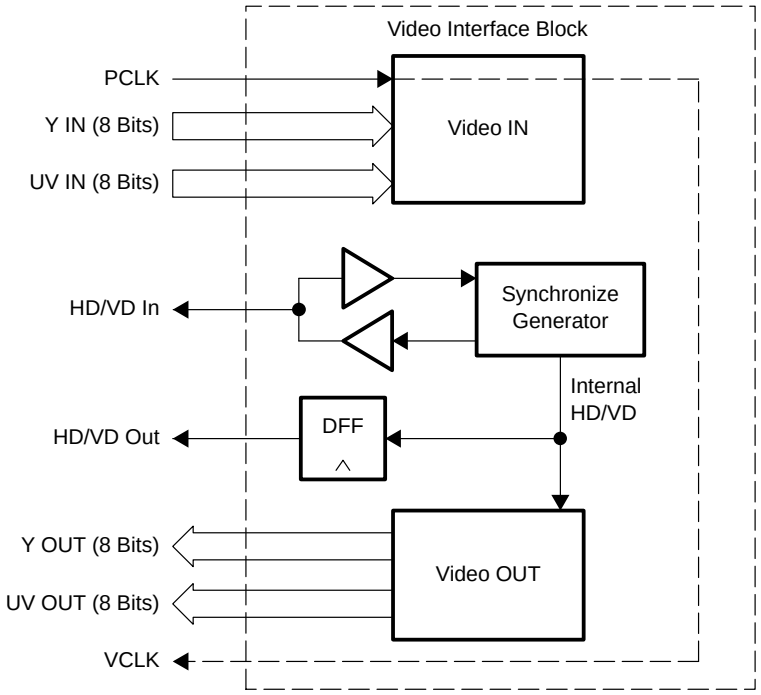


Figure 4–1. Video Interface Block Diagram in Mode 0

4.2.1.2 Device in Operating Mode 1

When used in mode 1, the DSC24 features two YCbCr input ports instead of only one. The video output port and the video input port 2 share the same pins.

In this mode, a field ID pin is available (correspond to the VCLK pin in mode 0). This pin is used to tell the system if the current frame loaded through the video-input interface is an odd or even frame.

When in mode 1, the HDOUT and VDOUT pins are not available anymore; the third serial port interface uses them instead. The video output is synchronized with the HD in and VD in signal instead.

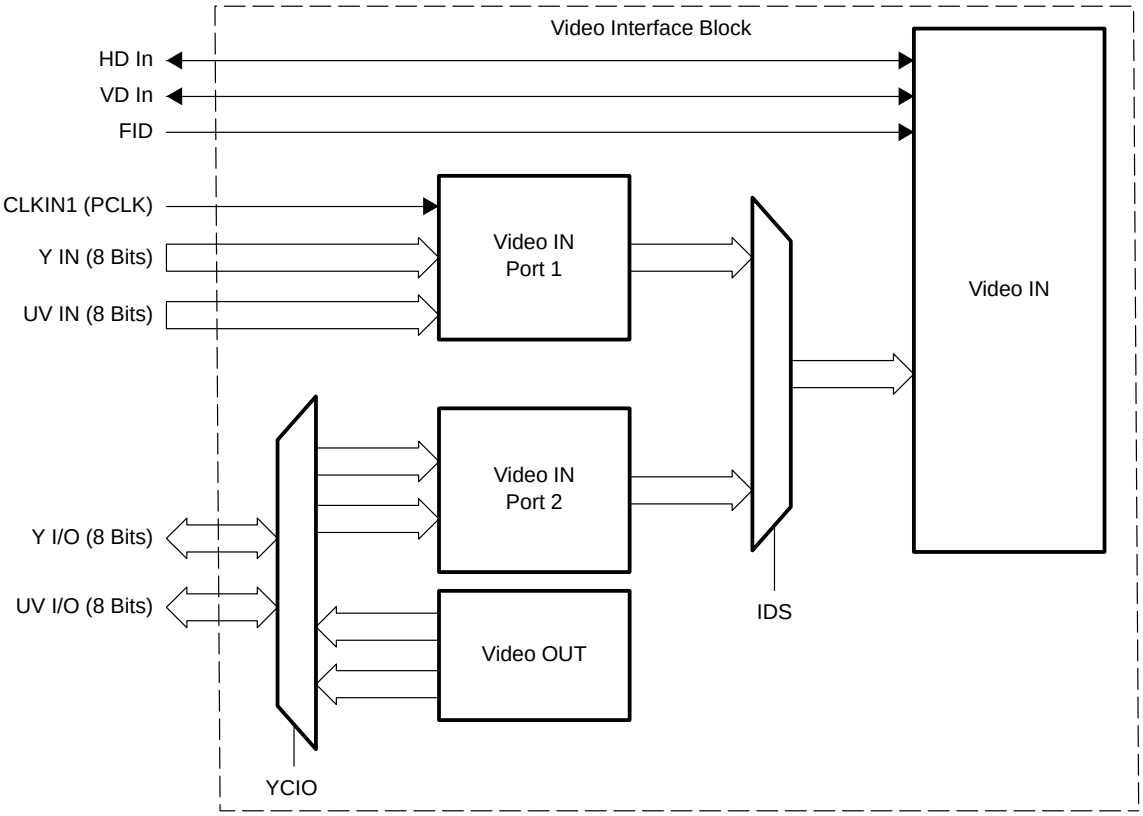


Figure 4–2. Video Interface Block Diagram in Mode 1

4.2.1.3 Device in Operating Mode 2

When the DSC24 is configured to run in mode 2, the video interface is similar to the one available in mode 0. However, the VCLK output pin is not available anymore. In addition, the HDOUT and VDOUT signals are generated directly from the HDIN and VDIN signals.

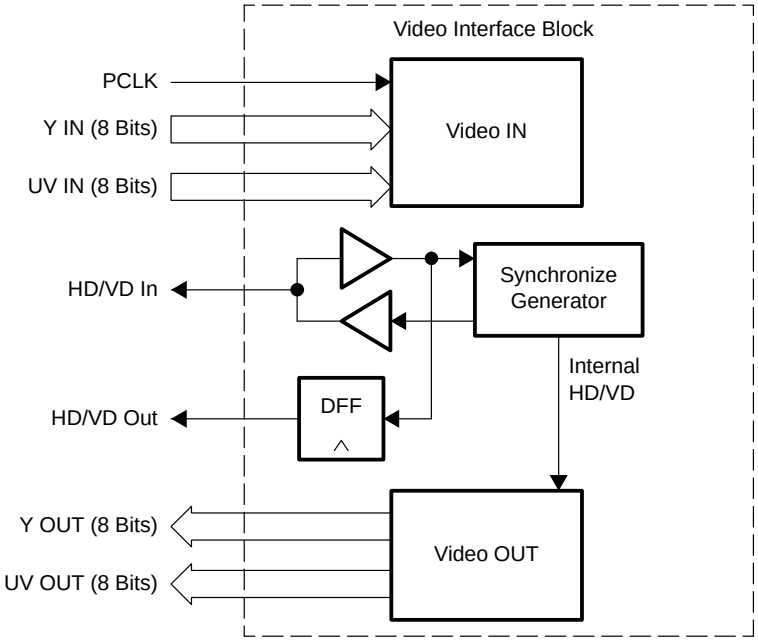


Figure 4–3. Video Interface Block Diagram in Mode 2

When used in mode 2, the device has video loop through capability. The YCIO bit in the MODESET register chooses the output source for the video. For YCIO=0, video input is looped back and directly fed to the video output pins. For YCIO=1, the video output source is the OSD module. When the video loop through is turned on, the external memory is still updated with the incoming video data as usual.

4.2.1.4 Device in Operating Mode 3

When the device is configured to be used in operating mode 3, the video output is not available anymore. Only one video input port remains. Eleven of the video output pins are used for GIO 21 to GIO31, seven are left unconnected, three are used to extend the first UART port to a 5-pin port, and the last pin is used as a watchdog timer output reset signal.

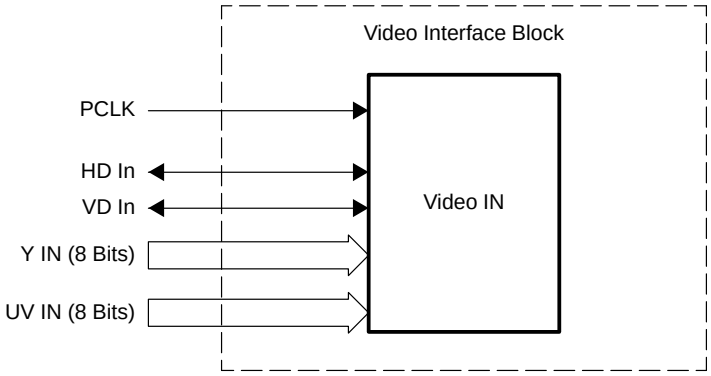


Figure 4–4. Video Interface Block Diagram in Mode 3

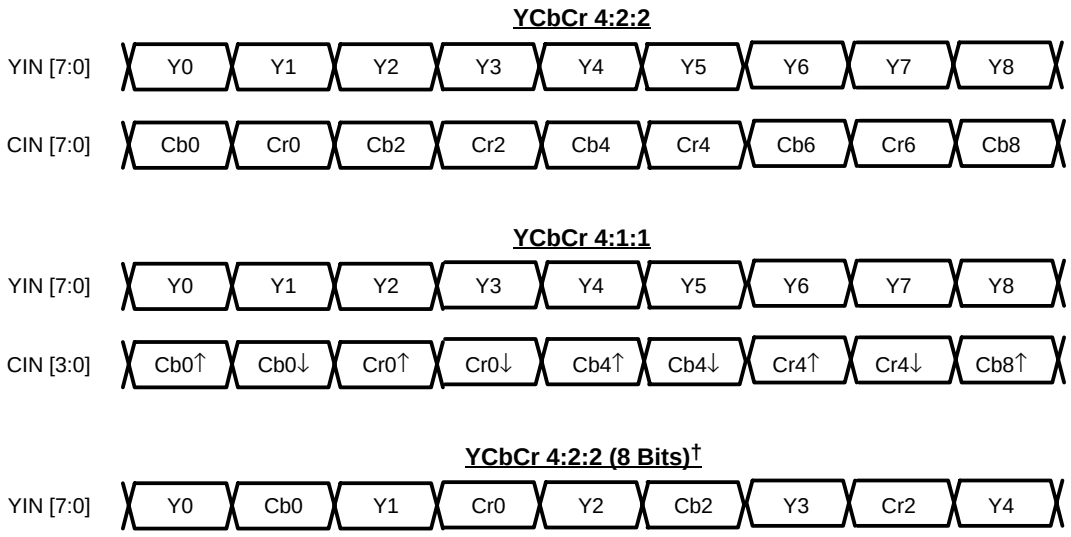
4.2.2 Video Data Input

4.2.2.1 Input Formats

The primary function of the video interface is to acquire data coming from other devices. The following formats are supported as input.

- 4:2:2 YCbCr 16 bits
- 4:1:1 YCbCr 12 bits
- 4:2:2 YCbCr 8 bits
- CCD/CMOS raw data 10 bits

The different formats are shown in Figure 4–5.

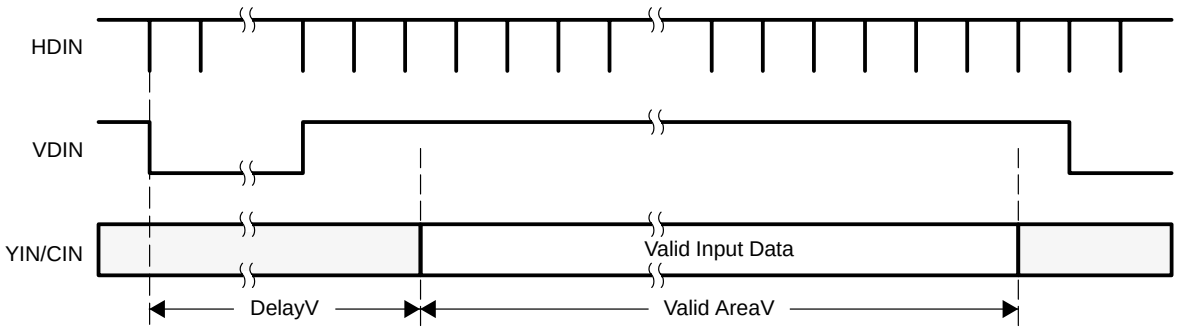


† PCLK = 2 x FCK

NOTE: Cb → Cr/Cr → Cb is programmable.

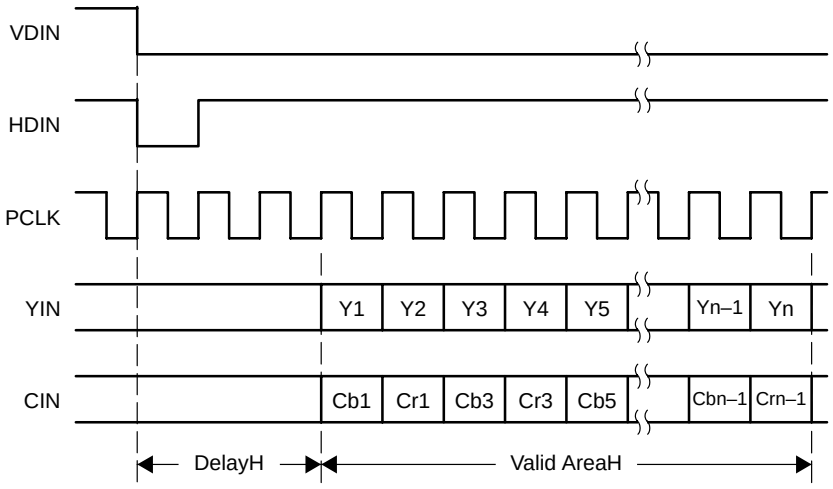
Figure 4–5. Input Video Formats

The timing diagrams for the CbCr 4:2:2 is given in Figure 4–6. The event succession is identical for the two other formats. To know the timing values, see the DSC24 data sheet, literature number SPRS195.



NOTE: DelayV, Valid AreaV are programmable.

Figure 4–6. Video Input Timing for Each Frame



NOTE: DelayH, Valid AreaH are programmable.

Figure 4–7. Video Input Timing for Each Line

When used for CCD/CMOS input, the video interface captures and processes the digital data from the sensor. The A/D converter and timing generator for the CCD sensor are external to the DSC24 chip. As seen before, the video interface can either provide HD/VD signals to an external timing generator or synchronize to HD/VD signals. The controller contains several sub-modules to complete its task.

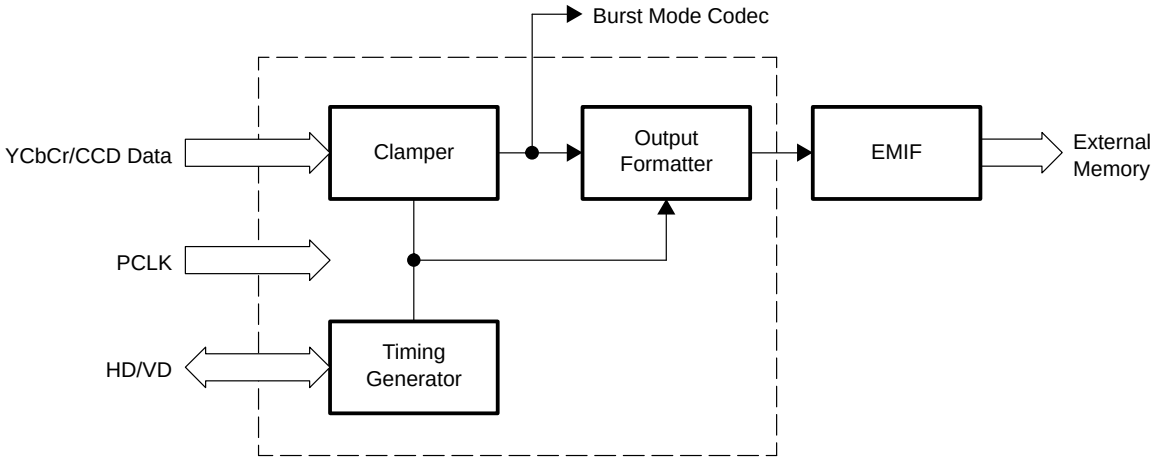


Figure 4–8. Video Input Block Diagram

When black clamping is used, it applies to all the CCD incoming data, but only on the Y information if the input is in a YCbCr format.

4.2.2.2 Digital Clamp

The first sub-module is the digital clamp block. The image data is latched at the rising or falling edge of the input pixel clock (PCLK). The averaging circuit takes an average of masked (black) pixel values from the image sensor and this value is subtracted from the image data. The user controls the position of the black pixels and the number of pixels averaged (8 or 16). Alternately, the user can select a constant black value for subtraction.

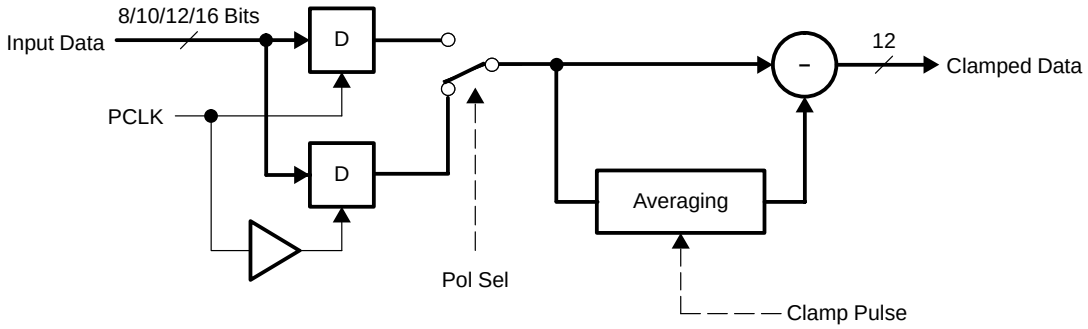


Figure 4–9. Digital Clamp Clock Diagram

When black clamping is used, it applies to all the CCD incoming data, but only on the Y information if the input is in a YCbCr format.

4.2.2.3 Frame Image

The timing generator provides the use of external sync signals (HD/VD) or internal generated timing signals. The MCU controls width, polarity, position, and direction of internal generated signals. Figure 4–10 shows various video interface register settings related to the timing. The shaded area is the physical imager size and the gray area is the valid data area. The image data in this area is stored to external memory or SDRAM.

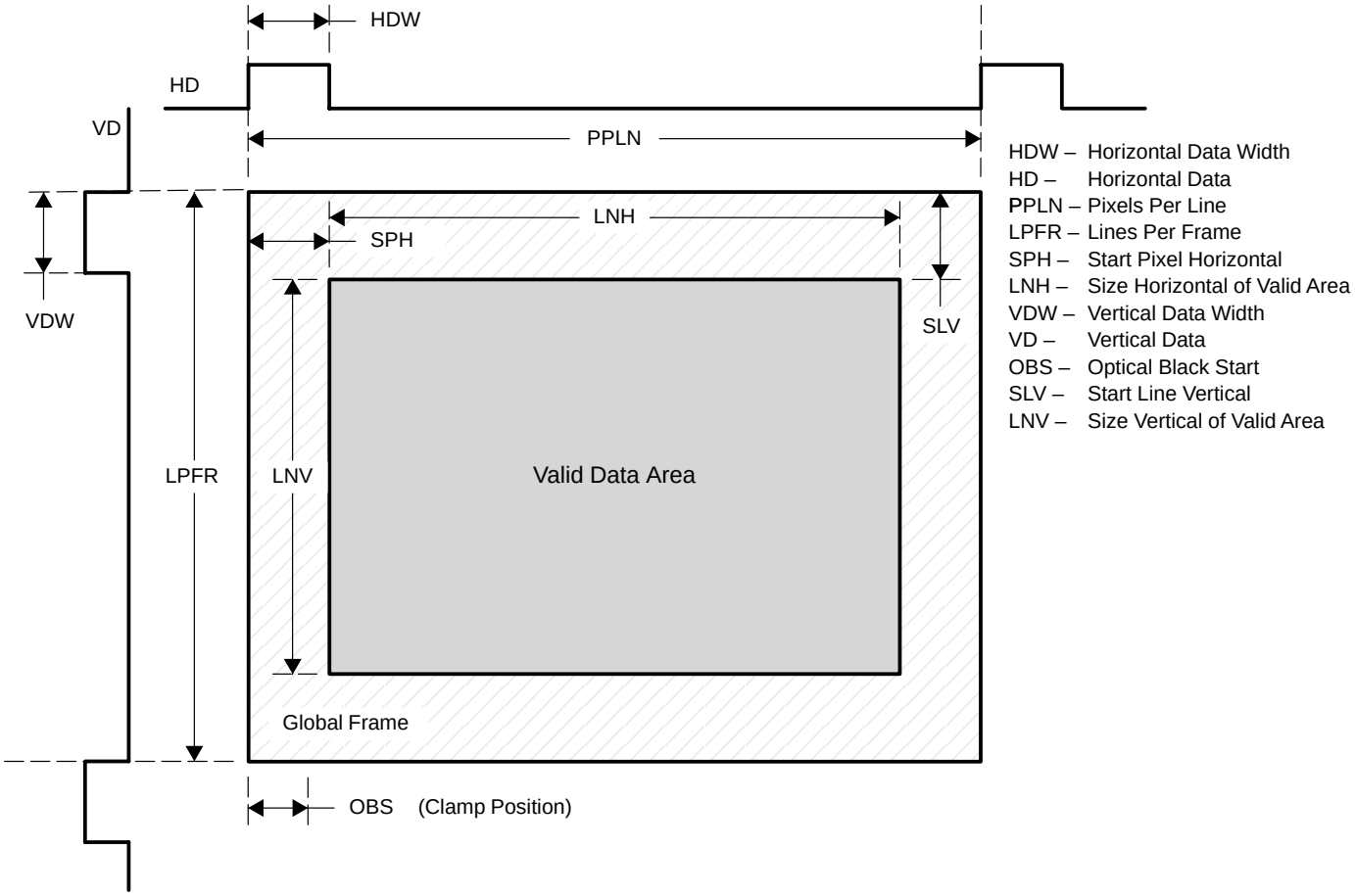


Figure 4–10. Frame Image Format

4.2.2.4 Field ID Mode

Bit 15 of the MODESET register (DVF) can be configured to select how the device handles the field information.

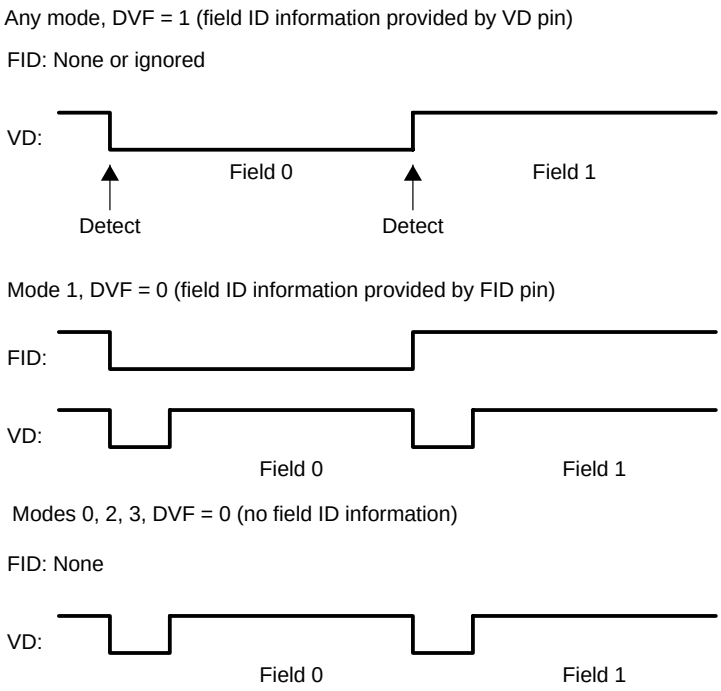


Figure 4–11. Field ID Detection

The field status can be known by reading the FSTAT bit of the SYNCEN register.

The DVF bit setting affects the video input as well as the video output ports, but only when they are configured as interlaced scan.

In operating mode 0 and mode 3, the DSC24 does not have a dedicated FID input pin. If the user wants to input an interlaced video signal, they may use a GIO pin for field status. The MCU has to read the status of the field and control the video interface module. Connecting the FID of the video decoder to the VD input pin of the DSC24 is another solution.

In operating mode 1, the FID input pin is used to indicate the field ID information. However, if the DVF bit is set to 1, the field ID is extracted from the VD signal.

When no field ID information is provided, only one field (odd or even) is used at all times.

4.2.2.5 Output Formatter

The output formatter (of the video-input block) provides options for applying an antialiasing filter, A-law compression, and a horizontal and vertical culling. The output formatter arranges the input data into 32-bit wide words for storage in external memory or SDRAM. The A-law table compresses the 10 bits of clamped CCD data to 8 bits. The final stage is a programmable decimation function along both the horizontal and vertical directions.

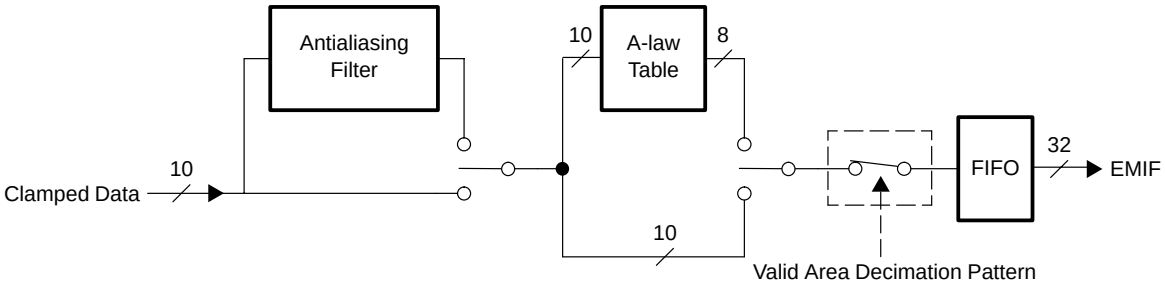


Figure 4–12. Output Formatter Block Diagram

4.2.2.6 Antialiasing Filter

The antialiasing filter consists of a simple three-tap filter. When on, 1/4 Z + 1/2 Z(–2) + 1/4 Z(–4) filter process is performed in horizontal direction on CCD data (10 bits) or Y data (8 bits).

To enable the antialiasing filter, set the TPF bit in the DECIF register. It applies on each 10-bit CCD data or on the Y data (8 bits).

4.2.2.7 A-Law Compression

The video interface provides two data paths of differing pixel bit resolutions to SDRAM. In the pass-through path, the 10 bits of sensor data from each pixel are stored as the upper bits of a 16-bit SDRAM word, with unused bit positions zero-filled. Data is stored to SDRAM 32 bits at a time, with the left-most pixel of the pair occupying the lower part of the 32-bit word. In the data compression path, the 10 bits of image data are compressed to 8 bits and four pixels are packed into each 32-bit SDRAM word. This format requires half the SDRAM capacity as the normal pass through path. The A-law table has the same characteristics of an audio codec one.

4.2.2.8 Decimation Pattern

The DECIM register controls the horizontal and vertical decimation. For each direction, the interface can be configured to save all the pixels within the SDRAM, only 1/2, 1/4, or 1/8.

The DECIM register controls this feature.

4.2.2.9 Output Data Format

The bits of data from each pixel are stored as the upper bits of a 16-bit SDRAM word, with unused bit positions zero filled.

Figure 4–13 and Figure 4–14 show the memory data format. If the input YCbCr data format is 4:2:2, it is stored as is. However, in case of 4:1:1 format, it is stored in a 4:2:2 format. In the case of CCD data input, 1 pixel is stored as 1 word and pushed to the lowest position.

YCbCr 4:2:2				10-Bit CCD			
	15 8	7 0			15 6	5 0	
n	Y0	Cb0		n	D0	0	
n+1	Y1	Cr0		n+1	D1	0	
n+2	Y2	Cb2		n+2	D2	0	
n+3	Y3	Cr2		n+3	D3	0	

YCbCr 4:1:1				8-Bit CCD			
	15 8	7 0			15 8	7 0	
n	Y0	Cb0		n	D1	D0	
n+1	Y1	Cr0		n+1	D3	D2	
n+2	Y2	Cb2		n+2	D5	D4	
n+3	Y3	Cr2		n+3	D7	D6	

Figure 4–13. Data Storage Format in 32-Bit Memory

YCbCr 4:2:2						10-Bit CCD					
	31 24	23 16	15 8	7 0			31 22	21 16	15 6	5 0	
n	Y1	Cr0	Y0	Cb0		n	D1	0	D0	0	
n+1	Y3	Cr2	Y2	Cb2		n+1	D3	0	D2	0	
n+2	Y5	Cr4	Y4	Cb4		n+2	D5	0	D4	0	
n+3	Y7	Cr6	Y6	Cb6		n+3	D7	0	D6	0	

YCbCr 4:1:1						8-Bit CCD					
	31 24	23 16	15 8	7 0			31 24	23 16	15 8	7 0	
n	Y1	Cb0	Y0	Cb0		n	D3	D2	D1	D0	
n+1	Y3	Cr0	Y2	Cr0		n+1	D7	D6	D5	D4	
n+2	Y5	Cb4	Y4	Cb4		n+2	D11	D10	D9	D8	
n+3	Y7	Cr4	Y6	Cr4		n+3	D15	D14	D13	D12	

Figure 4–14. Data Storage Format in 16-Bit Memory

4.2.2.10 External Connection

The video I/F module corresponds to 4:2:2 input and 4:1:1 input of YCbCr input data and CCD RAW data input. The relationship between the DSC24 video data input/output terminal and video data connection are as shown in Table 4–1.

Table 4–1. Video Input Data Connection

PIN NAME	YCbCr 4:2:2	YCbCr 4:1:1	YCbCr 8 Bit	CCD/CMOS
CIN7	C7 (MSB)			
CIN6	C6			
CIN5	C5			
CIN4	C4			
CIN3	C3	C3 (MSB)		
CIN2	C2	C2		
CIN1	C1	C1		CCD9 (MSB)
CIN0	C0 (LSB)	C0 (LSB)		CCD8
YIN7	Y7 (MSB)	Y7 (MSB)	YC7 (MSB)	CCD7
YIN6	Y6	Y6	YC6	CCD6
YIN5	Y5	Y5	YC5	CCD5
YIN4	Y4	Y4	YC4	CCD4
YIN3	Y3	Y3	YC3	CCD3
YIN2	Y2	Y2	YC2	CCD2
YIN1	Y1	Y1	YC1	CCD1
YIN0	Y0 (LSB)	Y0 (LSB)	YC0 (LSB)	CCD0 (LSB)

4.2.3 Video Output

4.2.3.1 Format Supported

In operating modes 0, 1, and 2, the video interface is also used to output some data to an external device. In mode 1, the output interface is multiplexed with an input port. In modes 0 and 2, 16 pins are dedicated to the Y and CbCr output, and two others to the VDOUT and HDOUT synchronization signals.

The video output sub-block takes the data from the on-screen display module and output it within one of the following format:

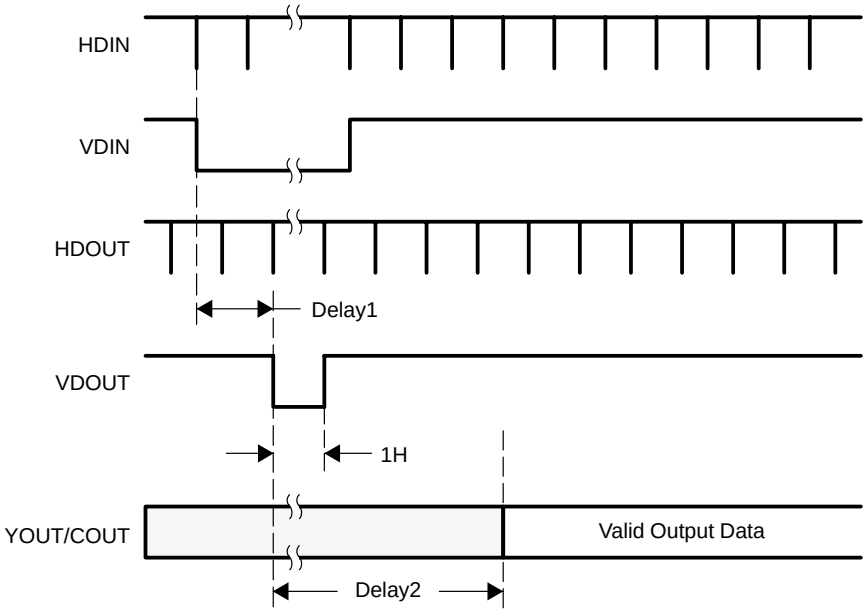
- 10-bit CCD raw data
- 4:2:2 YCbCr 16 bits

- 4:1:1 YCbCr 12 bits

When the device is used in operating mode 2, only the 16-bit YCbCr 4:2:2 format and video loop through mode are available. The loop through mode simply copies the data received on the video input pins onto the video output pins.

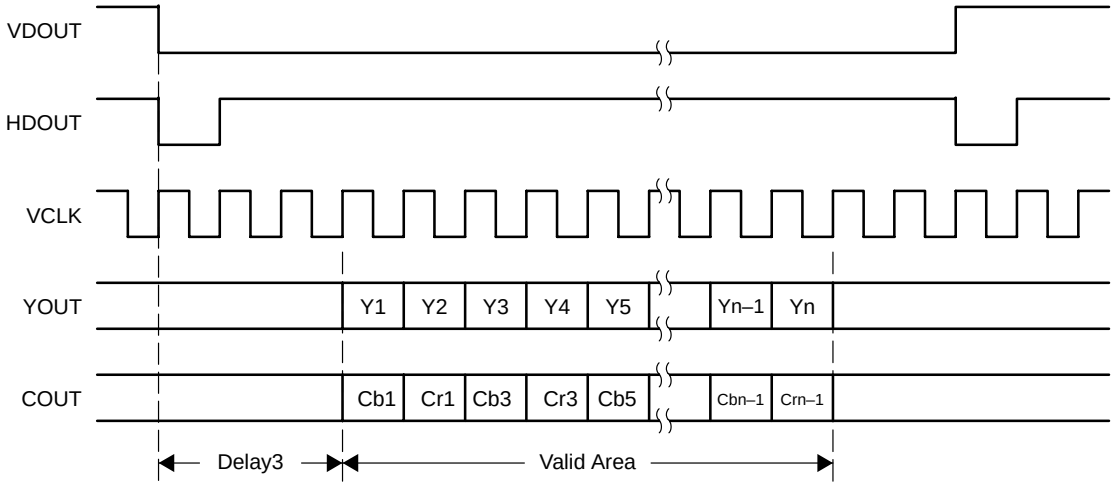
There is no special register to define the CCD output. When the content of SDRAM is CCD data, the output is CCD data.

The video interface block generates the HDOUT and VDOUT signals when available. These signals are synchronized with the HDIN and VDIN signals.



NOTE: Delay1, 2 are programmable.

Figure 4–15. Video Output Timing for Each Frame



NOTE: Delay3, Valid Area are programmable.

Figure 4–16. Video Output Timing for Each Line

The OSD module formats the video output data. The video interface only generates the HDOUT and VDOUT signals and synchronizes the data. The OSD module controls the format of the video output. See this chapter for more information.

4.2.3.2 Alternative to Video Output Interface

In any mode of operation, a different way to output the video data into an external device is to use the external memory interface (EMIF). See the *EMIF* chapter for more information.

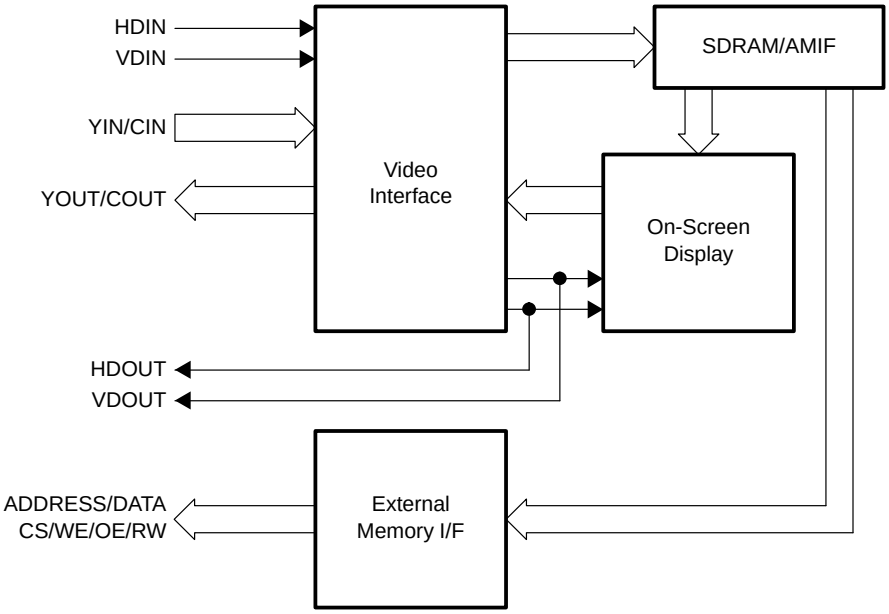


Figure 4–17. Video Output Paths

4.2.4 Interrupts

The video interface can generate two different interrupts: VD0 and VD1. The VD0 and VD1 interrupt can be set at any position in the VD interval.

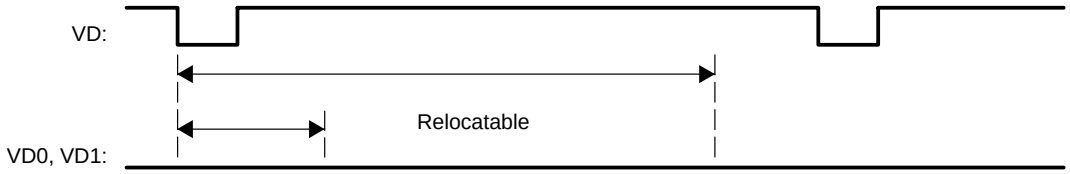


Figure 4–18. Video Interface Interrupts

4.2.5 Control Registers

There are 19 registers that control the video interface. The MCU is in charge of managing these registers to achieve the desired outcome of the video interface. These control registers allow the user to define the processing to be performed on the data—dc clamp, low-pass filter, and horizontal/vertical culling. A description of each is given in Table 4–2.

Table 4–2. Video Interface Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0780	SYNCEN	Synchronization enable
0x01	0x0003:0782	MODESET	Mode setting
0x02	0x0003:0784	CLAMP	Optical black sample/clamp
0x03	0x0003:0786	HDVDW	HD/VD width
0x04	0x0003:0788	PPLN	Pixels per line
0x05	0x0003:078A	LPFR	Lines per frame
0x06	0x0003:078C	SPH	Start position horizontal
0x07	0x0003:078E	LNH	Size of line horizontal
0x08	0x0003:0790	SLV	Start line vertical
0x09	0x0003:0792	LNV	Number of lines vertical
0x0A	0x0003:0794	DECIM	Decimation
0x0B	0x0003:0796	HSIZE	Horizontal size
0x0C	0x0003:0798	SDA_HI	SDRAM address high
0x0D	0x0003:079A	SDA_LO	SDRAM address low
0x0E	0x0003:079C	VDINT0	VD interrupt 0 timing
0x0F	0x0003:079E	VDINT1	VD interrupt 1 timing
0x010	0x0003:07A0	OSDHPOS	OSD output horizontal position
0x011	0x0003:07A2	OSDVPOS	OSD output vertical position
0x012	0x0003:07A4	FIDMODE	Field ID mode (mode 1 only)

Table 4–3. Synchronization Enable (SYNCEN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	FSTAT	RSV	RSV	RSV	RSV	RSV	RSV	RDWE	VHDEN
R/W	–	–	–	–	–	–	–	R	–	–	–	–	–	–	R/W	R/W
Default	–	–	–	–	–	–	–	0	–	–	–	–	–	–	0	0

Table 4–4. Synchronization Enable (SYNCEN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	FSTAT	Field status When in interlaced mode this bit indicates the field status 0: Odd field 1: Even field
7–2	RSV	Reserved bits. Not used.
1	RDWE	Row data write enable Starts writing of row data from external device to memory. After this bit is enabled, data is written starting from the next VD. 0: Disabled 1: Enabled
0	VHDEN	VD/HD enable Controls on/off of VD/HD output. From the point when 1 is written into this bit, the timing generator is activated, and VD/HD output is started. In case of input, VD/HD loading is started. 0: Disabled 1: Enabled

Table 4–5. Mode Setting (MODESET) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DVF	YCIO	IDS	DWDS	ULSEL	CBCR	DFRT1	DFRT0	ALAW	FMODE	DPOL	ETRG	FIP	HDP	VDP	VHDD
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–6. Mode Setting (MODESET) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	DVF	Video data format 0: Standard 1: VD is used to indicates field information (See the *Field IO Mode* section for more information)
14	YCIO	Y/C data input/output switching When the device is in mode 1, this bit is used to switch between video input port 2 and video output. 0: Input 1: Output When used in mode 2, this bit enables the video loop through: 0: Video output = Video input (Y/COUT = Y/CIN) 1: Video output = OSD output (Y/COUT = OSD output) When the video loop through mode is enabled, the memory can still be updated with video input data if the bit RDWE of the SYNCEN register is set. The YCIO bit has no effect in operating mode 0 or 3.
13	IDS	Input data switching When used in mode 1, this bit selects which video input port is used 0 : PORT 1 1 : PORT 2 This bit has no effect in any other modes.
12	DWDS	Data write direction switching Selects in which module the video data is to be sent. 0: SDRAM 1: External memory IF When the EMIF is chosen as the destination, bits 6:4 of the CCDCDSPDEST register select in which CS area the data is sent to.
11	ULSEL	C signal invert order when YCbCr 4:1:1 is input 0: C(7:4) – C(3:0) 1: C(3:0) – C(7:4)
10	CBCR	Chroma format Select the input order of the chrominance information: 0: Cb–Cr 1: Cr–Cb
9–8	DFMT1–DFMT0	Input data format DFMT1 DFMT0 Format 0 0 CbCr 4:2:2 (16 bits) 0 1 CbCr 4:1:1 (12 bits) 1 0 CbCr 4:2:2 (8 bits) 1 1 CCD/CMOS raw data (10 bits)
7	ALAW	A-law data compression: 0: Off 1: On This bit must be set with 0 if the input is not CCD or CMOS raw data.
6	FMODE	Field mode: Select the type of input data used. 0: Non-interlace 1: Interlace
5	DPOL	Data polarity: In case of CbCr input, when set to 1, MSB of C data input is inverted. 0: No change 1: MSB inversion of C signal

Table 4–6. Mode Setting (MODESET) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
4	ETRG	External trigger Selects how the video input data will be synchronized with PCLK 0: Rising edge loading 1: Falling edge loading
3	FIP	Field indicator polarity 0: Positive 1: Negative This bit inverse the polarity of the FID signal.
2	HDP	HD signal polarity. Selects HD signal polarity. 0: Positive 1: Negative
1	VDP	VD signal polarity. Selects VD signal polarity. 0: Positive 1: Negative
0	VHDD	VD/HD signal direction. Selects whether VD/HD is input or output. 0: Input 1: Output

Table 4–7. Optical Black Sample/Clamp (CLAMP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	OBL	OBEN	OBS11	OBS10	OBS9	OBS8	OBS7	OBS6	OBS5	OBS4	OBS3	OBS2	OBS1	OBS0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–8. Optical Black Sample/Clamp (CLAMP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–14	RSV	Reserved bits
13	OBL	Optical black sample length Selects how many pixels per line are used for the optical clamp 0: 8 samples 1: 16 samples
12	OBEN	Clamp enable 0: On 1: Off When on, the system subtracts from the input data the average value of the number of samples specified by bit 13 from the position set in OBS11–OBS0. This process is performed for each line. When off, the dc value set in OBS9–OBS0 is pulled from the input data.
11:0	OBS11–OBS0	Optical black sample start pixel OBEN (bit 12) On: Sets the position of OB by the distance from HD. OBEN (bit 12) Off: Sets the DC value that is subtracted from CCD.

Table 4–9. HD/VD Width (HDVDW) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	HDW7	HDW6	HDW5	HDW4	HDW3	HDW2	HDW1	HDW0	VDW7	VDW6	VDW5	VDW4	VDW3	VDW2	VDW1	VDW0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–10. HD/VD Width (HDVDW) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	HDW7–HDW0	HD width. Sets width of HD by number of PCLKs HD width = Value written + 1
7–0	VDW7–VDW0	VD width. Sets width of VD by number of lines VD width = Value written + 1

Table 4–11. Pixels Per Line (PPLN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	PPLN11	PPLN10	PPLN9	PPLN8	PPLN7	PPLN6	PPLN5	PPLN4	PPLN3	PPLN2	PPLN1	PPLN0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–12. Pixels Per Line (PPLN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	PPLN11–PPLN0	Pixels per lines Function differs depending on HD and VD setting. When HD, VD is output, sets the number of clocks of PCLK per line. PPLN value = value written + 1 When HD and VD is input, select when VD is sampled. The VD is sampled at the start of HD and delayed by the number of clocks set by bits 3:0. Bits 11:4 have no effect.

Table 4–13. Lines Per Frame (LPFR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	LPFR11	LPFR10	LPFR9	LPFR8	LPFR7	LPFR6	LPFR5	LPFR4	LPFR3	LPFR2	LPFR1	LPFR0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–14. Lines Per Frame (LPFR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	LPFR11–LPFR0	Sets number of lines of 1 frame or 1 field. The VD cycle is the value written + 1.

Table 4–15. Start Position Horizontal (SPH) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	SPH11	SPH10	SPH9	SPH8	SPH7	SPH6	SPH5	SPH4	SPH3	SPH2	SPH1	SPH0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–16. Start Position Horizontal (SPH) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	SPH11–SPH0	Position at which data loading into external memory starts is set by the position from the beginning of HD.

Table 4–17. Size of Line Horizontal (LNH) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	LNH11	LNH10	LNH9	LNH8	LNH7	LNH6	LNH5	LNH4	LNH3	LNH2	LNH1	LNH0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–18. Size of Line Horizontal (LNH) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	LNH11–LNH0	Sets number of pixels in horizontal direction of data loaded into external memory.

Table 4–19. Start Line Vertical (SLV) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	SLV11	SLV10	SLV9	SLV8	SLV7	SLV6	SLV5	SLV4	SLV3	SLV2	SLV1	SLV0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–20. Start Line Vertical (SLV) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	SLV11–SLV0	Line at which data loading into external memory starts is set by the position from the beginning of VD.

Table 4–21. Number of Lines Vertical (LNV) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	LNV11	LNV10	LNV9	LNV8	LNV7	LNV6	LNV5	LNV4	LNV3	LNV2	LNV1	LNV0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–22. Number of Lines Vertical (LNV) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	LNV11–LNV0	Sets number of lines in vertical direction of data loaded into external memory.

Table 4–23. Decimation (DECIM) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	TPF	VDECI1	VDECI0	HDECI1	HDECI0
R/W	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0

Table 4–24. Decimation (DECIM) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–5	RSV	Reserved bits. Not used.
4	TPF	3-tap filter 0: Off 1: On
3–2	VDECI1–VDECI0	Vertical decimation Sets culling ratio when data is loaded into SDRAM (vertical). VDECI1 VDECI0 Culling ratio 0 0 1/1 0 1 1/2 1 0 1/4 1 1 1/8
1–0	HDECI1–HDECI0	Horizontal decimation. Sets culling ratio when data is loaded into SDRAM (horizontal). HDECI1 HDECI0 Culling ratio 0 0 1/1 0 1 1/2 1 0 1/4 1 1 1/8

Table 4–25. Horizontal Size (HSIZE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	LS7	LS6	LS5	LS4	LS3	LS2	LS1	LS0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 4–26. Horizontal Size (HSIZE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7–0	LS7–LS0	Line size When data is loaded into external memory, sets number of pixels per line. Set in 16-pixel units.

Table 4–27. SDRAM Address High (SDA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	S21	S20	S19	S18	S17	S16
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–28. SDRAM Address High (SDA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	S21–S16	Specifies lead upper address of external memory when writing data into external memory. Byte address is the value set here multiplied by 32.

Table 4–29. SDRAM Address Low (SDA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	S15	S14	S13	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–30. SDRAM Address Low (SDA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	S15–S0	Specifies lead lower address of external memory when writing data into external memory. Byte address is the value set here multiplied by 32.

Table 4–31. VD Interrupt 0 Timing (VDINT0) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	VDI011	VDI010	VDI09	VDI08	VDI07	VDI06	VDI05	VDI04	VDI03	VDI02	VDI01	VDI00
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–32. VD Interrupt 0 Timing (VDINT0) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	VDI011–VDI00	Sets CCD VD0 interrupt timing. Sets line position in field 1 from the start of VD.

Table 4–33. VD Interrupt 1 Timing (VDINT1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	VDI111	VDI110	VDI19	VDI18	VDI17	VDI16	VDI15	VDI14	VDI13	VDI12	VDI11	VDI10
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–34. VD Interrupt 1 Timing (VDINT1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	VDI111–VDI10	Sets CCD VD1 interrupt timing. Sets line position in field 1 from start of VD.

Table 4–35. OSD Output Horizontal Position (OSDHPOS) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OSH11	OSH10	OSH9	OSH8	OSH7	OSH6	OSH5	OSH4	OSH3	OSH2	OSH1	OSH0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–36. OSD Output Horizontal Position (OSDHPOS) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	OSH11–OSH10	Specifies the delay of the HSYNC signal of OSD. This register affects the HDOUT signal.

Table 4–37. OSD Output Vertical Position (OSDVPOS) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OSV11	OSV10	OSV9	OSV8	OSV7	OSV6	OSV5	OSV4	OSV3	OSV2	OSV1	OSV0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–38. OSD Output Vertical Position (OSDVPOS) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	OSV11–OSV10	Specifies the delay of the VSYNC signal of OSD. This register affects the VDOUT signal.

Table 4–39. Field ID Mode (FIDMODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	HVP	FIDL†	FIDD†	FIDP†
R/W	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0	0

† These bits affect the operating mode 1 only.

Table 4–40. Field ID Mode (FIDMODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–4	RSV	Reserved bits. Not used
3	HVP	Reverses polarity of HD/VD to the burst compression module
2	FIDL†	Field ID loading When set to 0, FID is loaded with the timing of OSDVPOS. Loaded at the head of VD. When 1, the input FID is supplied to the OSD module.
1	FIDD†	Field ID delay – When set to 1, the FID is delayed by 3H and loaded at the head of VD.
0	FIDP†	Field ID polarity – Reverses polarity of FID to the OSD module

† These bits affect the operating mode 1 only.

4.3 Burst Mode Compression/Decompression

The DSC engine includes an improved burst compression/decompression hardware function, which allows high speed, full-resolution capture of a sequence of images. This dedicated compression and decompression hardware allows for a given amount of SDRAM capacity and increased burst capture sequence length. A sequence of raw image frames is first compressed and stored in SDRAM using the burst compression engine. Then, in off-line processing or concurrently, the image pipeline of the regular still image capture mode retrieves the CCD raw images from the SDRAM, processes them sequentially (decompression followed by image processing and JPEG compression), and finally stores them back in SDRAM as JPEG files.

4.3.1 Features

The burst codec allows real-time capture of CCD/CMOS raw data without compromising the image resolution at a speed of up to 37.5 MHz for the pixel clock. The module can be configured to use lossless or lossy compression formats. The CCD can either be an interlaced or non-interlaced sensor. Synchronization with the HD and VD signals is possible. Expansion of only partial images is possible. The expansion and compression processes can run simultaneously.

4.3.2 Data Flow

Data flow is a three-step process:

1. Raw CCD data stored in SDRAM via compression engine
2. Off-line decompression
3. Off-line image pipe processing

The compression engine takes CCD raw data from the CCD controller in progressive mode, i.e., at the full CCD resolution. The decompression engine reads the compressed bit stream from SDRAM. Reconstructed CCD raw data from the decompression engine is then available to be processed by the capture mode's image pipeline.

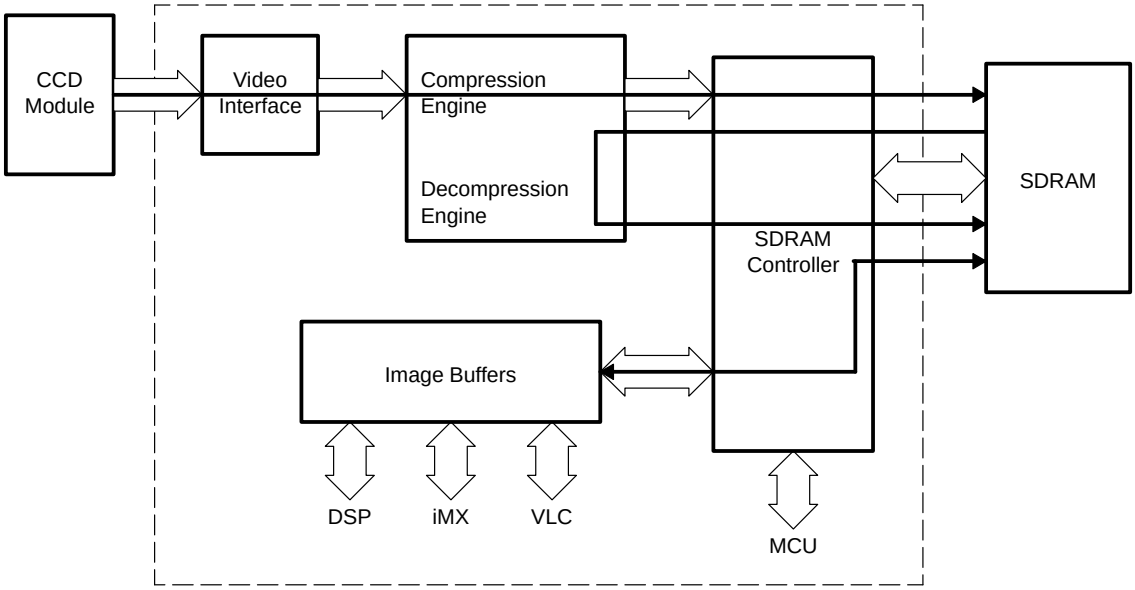


Figure 4–19. Data Flow in the Burst Mode Compression

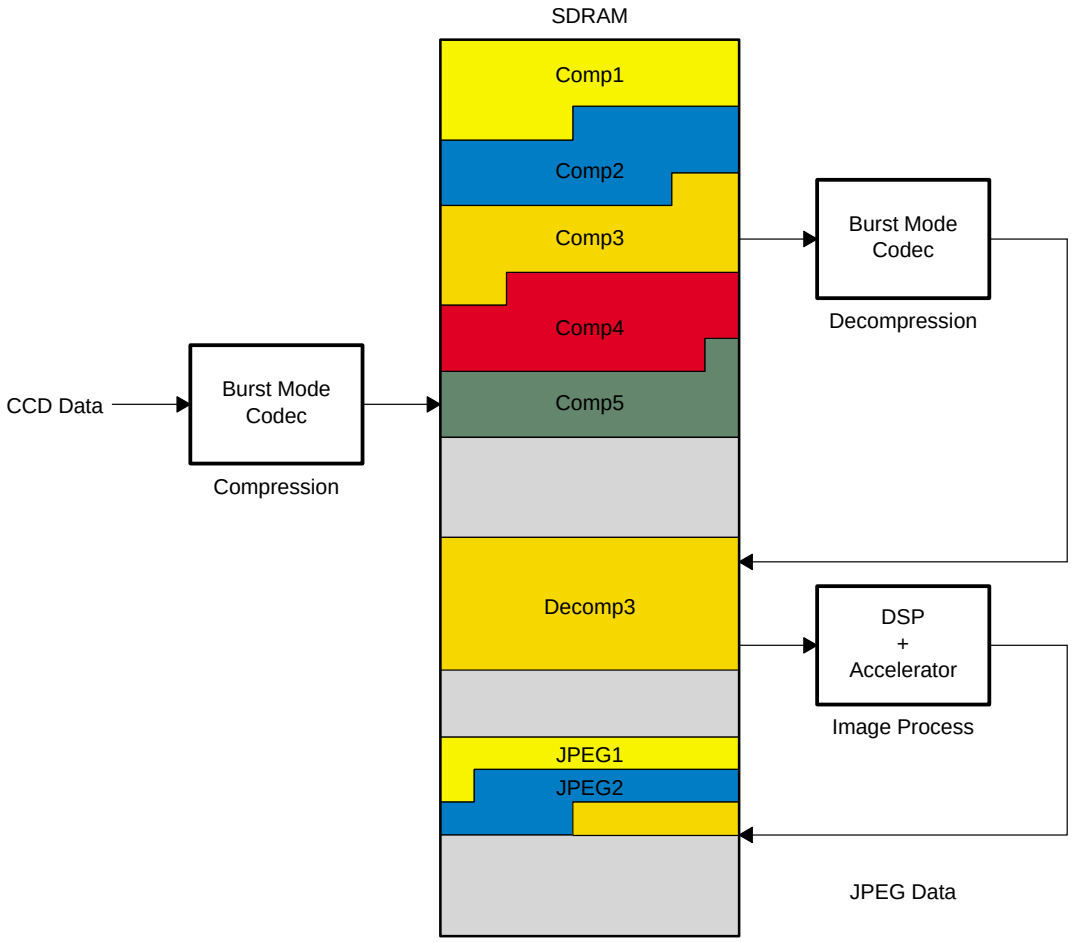


Figure 4–20. Burst Mode Memory Usage

4.3.3 Burst Mode Compression

Figure 4–21 shows the burst mode codec process. Input is 10 bits of CCD/CMOS sensor RAW data supplied from the video interface module. The raw data is first quantified and then compressed by first-order DPCM and Huffman encoding. The output bit stream is stored in SDRAM. This process is performed in real time. Each time the compression of one image is finished, an interrupt is generated to the MCU. At this time, the MCU knows the storage destination (address) of the compressed data. Since it is variable-length compression, the MCU must save the storage destination for each image.

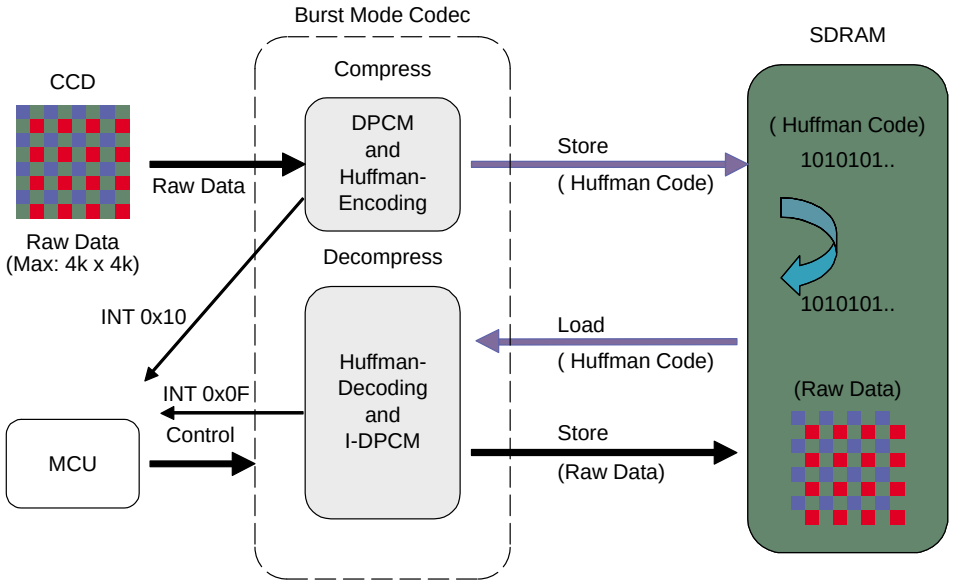


Figure 4–21. Burst Mode Codec Process

4.3.3.1 CCD Controller

The burst engine takes its data from an external CCD or CMOS sensor. The first step is to configure the CCD interface. The registers described below must be configured for this purpose.

Table 4–41. CCD Controller Registers

OFFSET	ADDRESS	REGISTER NAME	
0x02	0x0003:0804	CCDMODE	CCD mode
0x03	0x0003:0806	COLP	CCD color pattern
0x05	0x0003:080A	STPIX	Start pixel horizontal
0x06	0x0003:080C	SIZEPIX	Number of pixels horizontal
0x07	0x0003:080E	STLIN	Start line vertical
0x08	0x0003:0810	SIZELIN	Number of lines vertical

Table 4–42. CCD Mode (CCDMODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VDDET	CCDF	CCDW	FRF	RSV	INTER	PIXR2	PIXR1	PIXR0
R/W	–	–	–	–	–	–	–	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
Default	–	–	–	–	–	–	–	0	0	0	0	—	0	0	0	0

Table 4–43. CCD Mode (CCDMODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	VDDET	VD detection method: Selects the detection method of the VD signal (CCD module output) in encoding (compression). 1: VD signal status is detected without relation to HD. 0: VD signal status is detected at rising edge of HD signal.
7	CCDF	CCD field ID signal status: Sets whether or not there is reference to the FID signal (CCD module output) when interlace compression is set (INTER = 0). 0: FID signal status is not used. 1: FID signal status is used.

Table 4–43. CCD Mode (CCDMODE) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
6	RSV	Reserved. Writing to this bit has not effect
5	FRF	First field bit: Selects the first field at which encoding (compression) begins. 0: Compression starts on an even field. 1: Compression starts on an odd field. This bit is valid only when set to interlace (INTER = 0) and CCD_FID signal reference (CCDF = 1).
4	RSV	Reserved bit. Not used.
3	INTER	Selects interlace mode: 0: Non interlaced mode 1: Interlaced mode.
2	PIXR2	This bit is reserved. Always write 0
1–0	PIXR1–PIXR0	Pixel resolution: Specifies the width of compressed CCD data. The width of the data bit is set here from the MSB of the input when CCD data is used. 0: 8 bits 1: 9 bits 2: 10 bits 3: Reserved

When the encoder is used (compression) and if the interlaced mode is selected, the field selected by FRF is an even field (FIELD0).

When the decoder is used (decompression) and if the interlaced mode is selected, data stored in SDRAM after decompression is data that has been converted to the noninterlace format aligned in FIELD0/FIELD1 for each line.

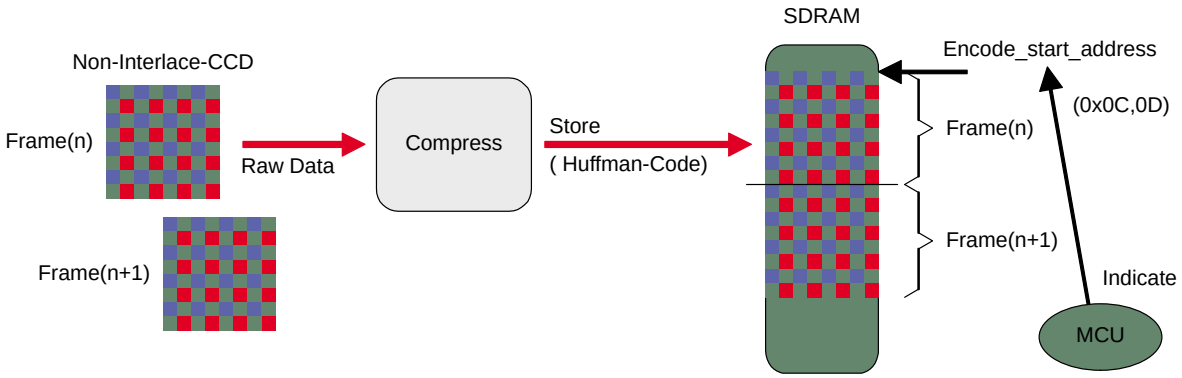


Figure 4–22. Compress Data in Noninterlaced Mode

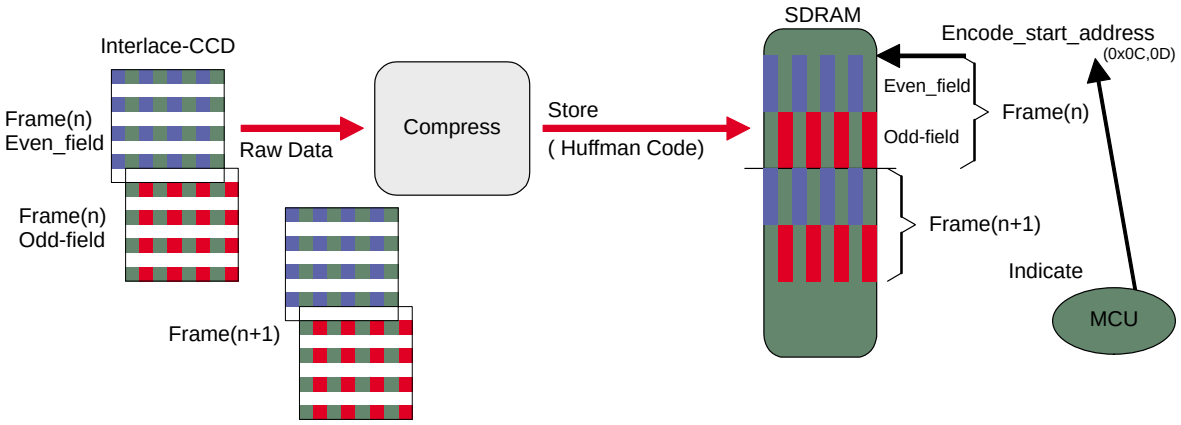


Figure 4–23. Compress Data in Interlaced Mode

Table 4–44. CCD Color Pattern (COLP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	EEO1	EEO0	OEO1	OEO0	EOO1	EOO0	OOO1	OOO0	EEE1	EEE0	OEE1	OEE0	EOE1	EOE0	OOE1	OOE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–45. CCD Color Pattern (COLP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–14	EEO1–EEO0	Specifies color pattern with even number of pixels, even number of lines, and odd number of fields. EEO1 EEO0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
13–12	OEO1–OEO0	Specifies color pattern with odd number of pixels, even number of lines, and odd number of fields. OEO1 OEO0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
11–10	EOO1–EOO0	Specifies color pattern with even number of pixels, odd number of lines, and odd number of fields. EOO1 EOO0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
9–8	OOO1–OOO0	Specifies color pattern with odd number of pixels, odd number of lines, and odd number of fields. OOO1 OOO0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
7–6	EEE1–EEE0	Specifies color pattern with even number of pixels, even number of lines, and even number of fields. EEE1 EEE0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
5–4	OEE1–OEE0	Specifies color pattern with odd number of pixels, even number of lines, and even number of fields. OEE1 OEE0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G

Table 4–45. CCD Color Pattern (COLP) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
3–2	EOE1–EOE0	Specifies color pattern with even number of pixels, odd number of lines, and even number of fields. EOE1 EOE0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G
1–0	OOE1–OOE0	Specifies color pattern with odd number of pixels, odd number of lines, and even number of fields. OOE1 OOE0 0 0 R/Ye 0 1 G/Mg 1 0 B/Cy 1 1 +/G

The CCD controller needs to be configured to the size of the image sensor. The four registers described below must be configured with the proper values. Figure 4–24 shows the correspondences between the image and the register values.

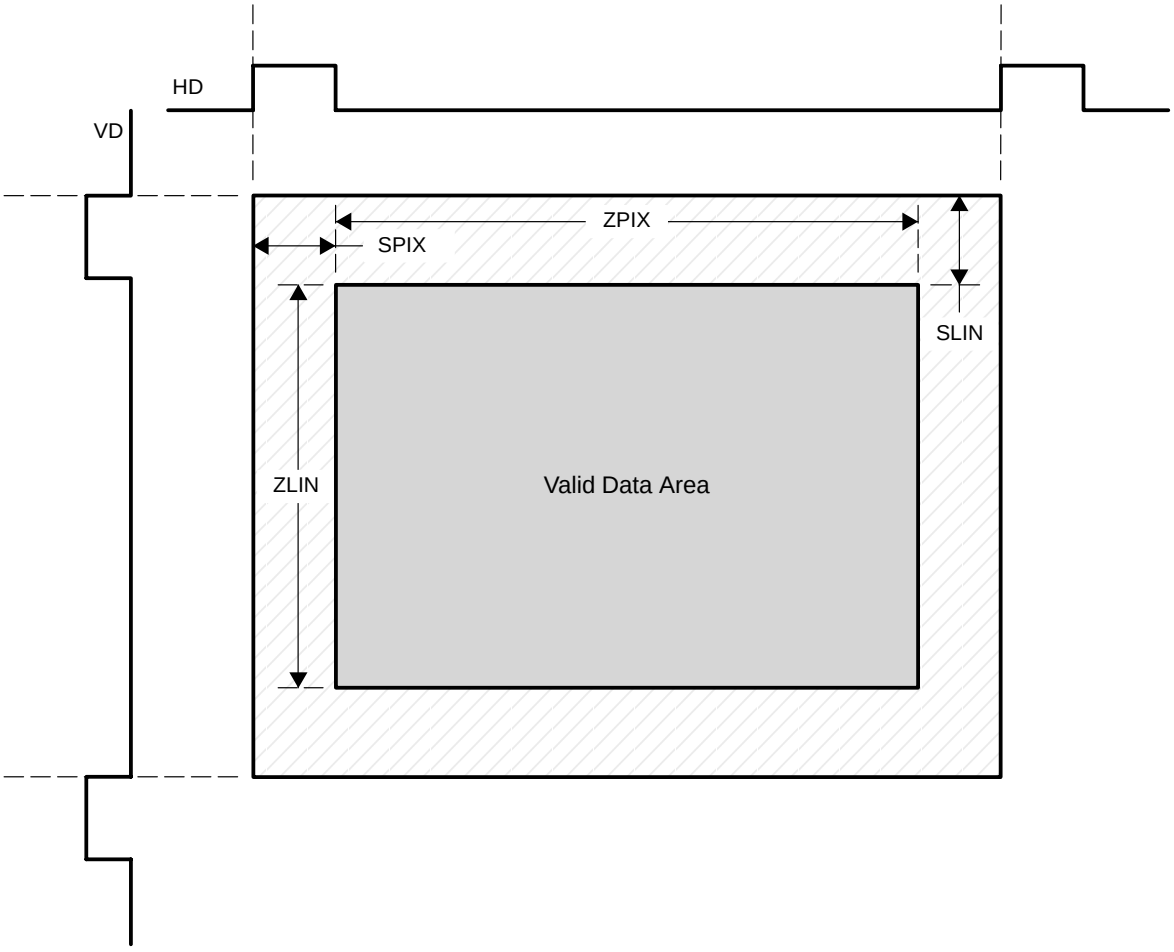


Figure 4–24. CCD Controller Configuration

Table 4–46. Start Pixel Horizontal (STPIX) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	SPIX11	SPIX10	SPIX9	SPIX8	SPIX7	SPIX6	SPIX5	SPIX4	SPIX3	SPIX2	SPIX1	SPIX0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–47. Start Pixel Horizontal (STPIX) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	SPIX11–SPIX0	Set position at which data loading into SDRAM begins by the position from HD

Table 4–48. Number of Pixels Horizontal (SIZEPIX) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	ZPIX11	ZPIX10	ZPIX9	ZPIX8	ZPIX7	ZPIX6	ZPIX5	ZPIX4	ZPIX3	ZPIX2	ZPIX1	ZPIX0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–49. Number of Pixels Horizontal (SIZEPIX) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	ZPIX11–ZPIX0	Set number of pixels per line at which data is loaded into SDRAM.

Table 4–50. Start Line Vertical (STLIN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	SLIN11	SLIN10	SLIN9	SLIN8	SLIN7	SLIN6	SLIN5	SLIN4	SLIN3	SLIN2	SLIN1	SLIN0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–51. Start Line Vertical (STLIN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	SLIN11–SLIN0	Sets position at which data loading into SDRAM begins by the position from VD.

Table 4–52. Number of Lines Vertical (SIZELIN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	ELIN11	ELIN10	ELIN9	ELIN8	ELIN7	ELIN6	ELIN5	ELIN4	ELIN3	ELIN2	ELIN1	ELIN0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–53. Number of Lines Vertical (SIZELIN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	ELIN11–ELIN0	Sets number of lines (LINE/VD) at which data is loaded into SDRAM.

4.3.3.2 Quantization

The first processing in the compression engine is quantization of the input CCD raw data. For simplicity, quantization is implemented using bit shift operation. Quantization step size equal to 1 or left-shift by bit 0 corresponds to lossless compression. The 1- and 2-bit left shifts are useful in the case of perceptually lossless compression. The 3- and 4-bit left shifts can also be used for lossy compression.

Bits 1 and 0 of the burst compression CCD mode register select the pixel resolution used.

4.3.3.3 DPCM

Figure 4–25 shows the diagram of a single-step prediction DPCM for each color component of CFA data. The DPCM scheme generates differential data based on a simple 1-step prediction. At the beginning of each line, it simply uses the last available pixel value for each color component as the prediction value.

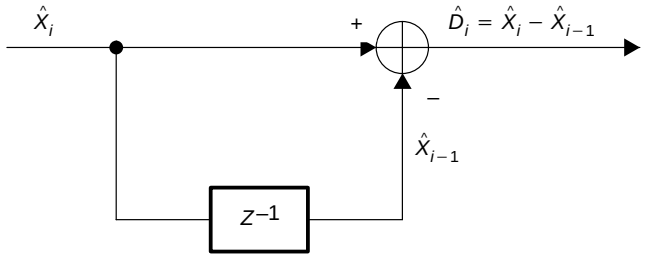


Figure 4–25. Block Diagram of Burst Mode Compression DPCM

4.3.3.4 Huffman Encoding

The differential data obtained through the DPCM is then entropy-coded by a Huffman coder. The Huffman coder used is the same as the one for entropy coding of dc coefficients in the baseline JPEG. Table 4–54 shows the fixed table used for Huffman encoding.

Table 4–54. Fixed Huffman Table (JPEG Huffman Table for Chrominance DC Coefficients)

CATEGORY (SSSS)	$\hat{D}_i$	CODE LENGTH	CODEWORD
0	0	2	00
1	–1,1	2	01
2	–3,–2,2,3	2	10
3	–7,..,–4,4,..,7	3	110
4	–15,..,–8,8,..,15	4	1110
5	–31,..,–16,16,..,31	5	11110
6	–63,..,–32,32,..,63	6	111110
7	–127,..,–64,64,..,127	7	11111110
8	–255,..,–128,128,..,128	8	111111110
9	–511,..,–256,256,..,511	9	1111111110
10	–1023,..,–512,512,..,1023	10	11111111110
11	–2047,..,–1024,1024,..,2047	11	111111111110
12	–4095,..,–2048,2048,..,4095	12	1111111111110

The Huffman table used in the compression engine is not programmable by the user.

4.3.3.5 Configuration of the Compression Engine

The burst compression memory mapped registers let the user modify some parameters of the compression engine. The CCDMode register selects the type of CCD they have (interlaced or non-interlaced) as well as the pixel resolution (see the *CCD Controller* section). The color pattern register (COLP) is used to tell the system what type of data is coming out of the image sensor (Bayer pattern for example).

You cannot select the time interval between two captures. The number of images to skip defines the interval between two captures. The encode interval register (INTVL) lets you specify this number. You can skip from 0 (continuous shot) to 15 images between two captures. This register lets the user also define the total number of images that are encoded (up to 1024). The time interval between two images (encoded or skipped) depends on the pixel clock and the sensor resolution used.

Table 4–55. Compression Engine Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0800	BURSTENC	Burst encoder control
0x04	0x0003:0808	INTVL	Encode interval
0x09	0x0003:0812	PNPLN	Pixels per line
0x0A	0x0003:0814	DECMPLN	Number of decompressed lines

Table 4–56. Encode Interval (INTVL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ENCI3	ENCI2	ENCI1	ENCI0	RSV	RSV	TPNE9	TPNE8	TPNE7	TPNE6	TPNE5	TPNE4	TPNE3	TPNE2	TPNE1	TPNE0
R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–57. Encode Interval (INTVL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	ENCI3–ENCI0	Specifies interval of encoded image data 0: Encodes image of continuous frames 12–15: Skips specified number For example, when 3 is set, it becomes OxxxOxxxOxxx
11–10	RSV	Reserved bits. Not used.
9–0	TPNE9–TPNE0	Specifies total number of pieces of encoded image data. Encoding of the number set here is performed. When set to 0, it is 1024.

The decompression engine needs to know how many lines and how many pixels were present in the original image. This information is given to the system by writing into the PNPLN and DECMPLN registers. In general, the PNPL and DLN values are identical respectively to the ZPIX and ZLIN values from the compression engine.

Table 4–58. Pixels Per Line (PNPLN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	PNPL11	PNPL10	PNPL9	PNPL8	PNPL7	PNPL6	PNPL5	PNPL4	PNPL3	PNPL2	PNPL1	PNPL0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–59. Pixels Per Line (PNPLN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	PNPL11–PNPL0	Sets number of pixels per line (during decompression). Since transfer to SDRAM is performed for every 16 pixels (8 words), the value set here must be a multiple of 16. Also, it must be at least the value set in the SIZEPIX (0003:080C) register.

The SDSAUP and SDSALO registers let the user specify where within the SDRAM they want to store the data.

Each time a compression finishes, the PICTPAUP and PICTPALO registers are loaded with the top address of the compressed image.

To prevent loss of data, two registers (ENCA_HI and ENCA_LO) let the user specify the end address for the compressed image buffer. The burst capture stops as soon as either of the following conditions are met.

- The maximum number of pictures is reached (INTVL register)
- The end of the SDRAM buffer is reached (ENCA_HI and ENCA_LO registers).

When the second event occurs, bit 8 of the burst enable register (BURSTEN) is set.

Table 4–60. Burst Encoder Control (BURSTENC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	ESTAT	RSV	RSV	RSV	RSV	RSV	RSV	RSV	DECEN
R/W	—	—	—	—	—	—	—	R	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	0	—	—	—	—	—	—	—	0

Table 4–61. Burst Encoder Control (BURSTENC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	ESTAT	Error status Becomes 1 when error occurs. An error occurs when the SDRAM write address becomes the address set in the EMCAUP (0003:0834) and ENCA_LO (0003:0836) registers.
7–1	RSV	Reserved bits. Not used.
0	DECEN	Encoder enable When 1 is set in this register, encoding begins from the next valid VD (field set in CCDMODE register) that comes and it is cleared automatically when compression ends after the number set in the INTVL register. Also, if 0 is set during the encoding process, it is forcibly ended. 0: Disabled 1: Enabled

4.3.3.6 Burst Compression Flow in Noninterlaced Mode

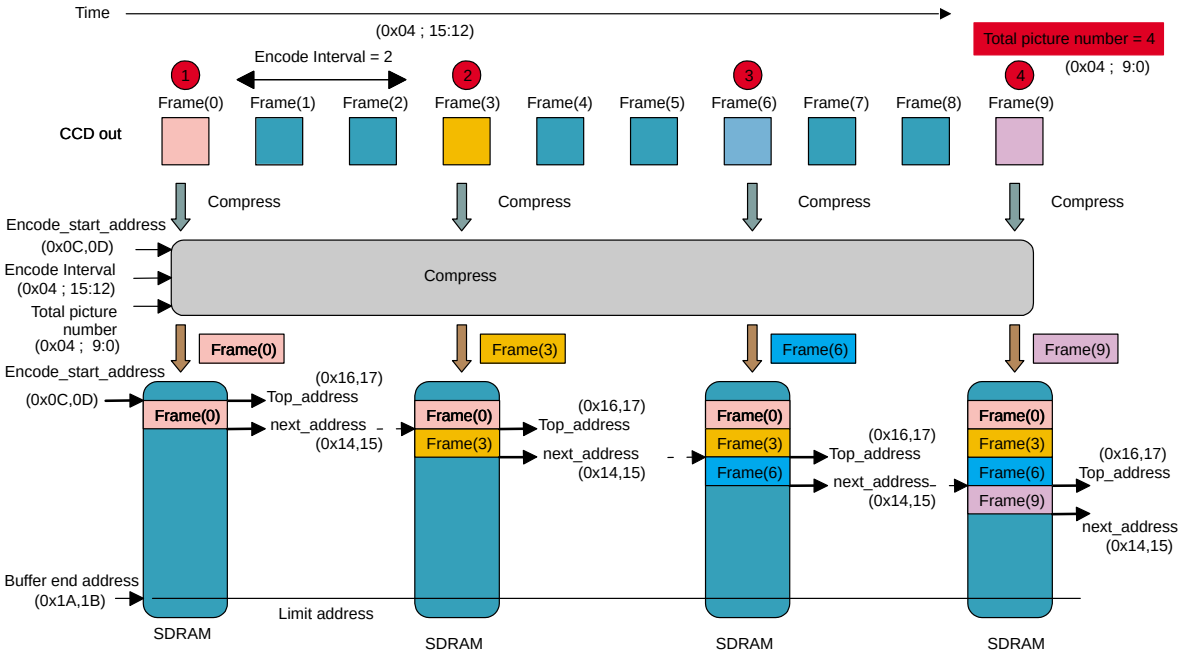


Figure 4–26. Burst Compression Flow in Noninterlaced Mode

4.3.3.7 Burst Compression Flow in Interlaced Mode

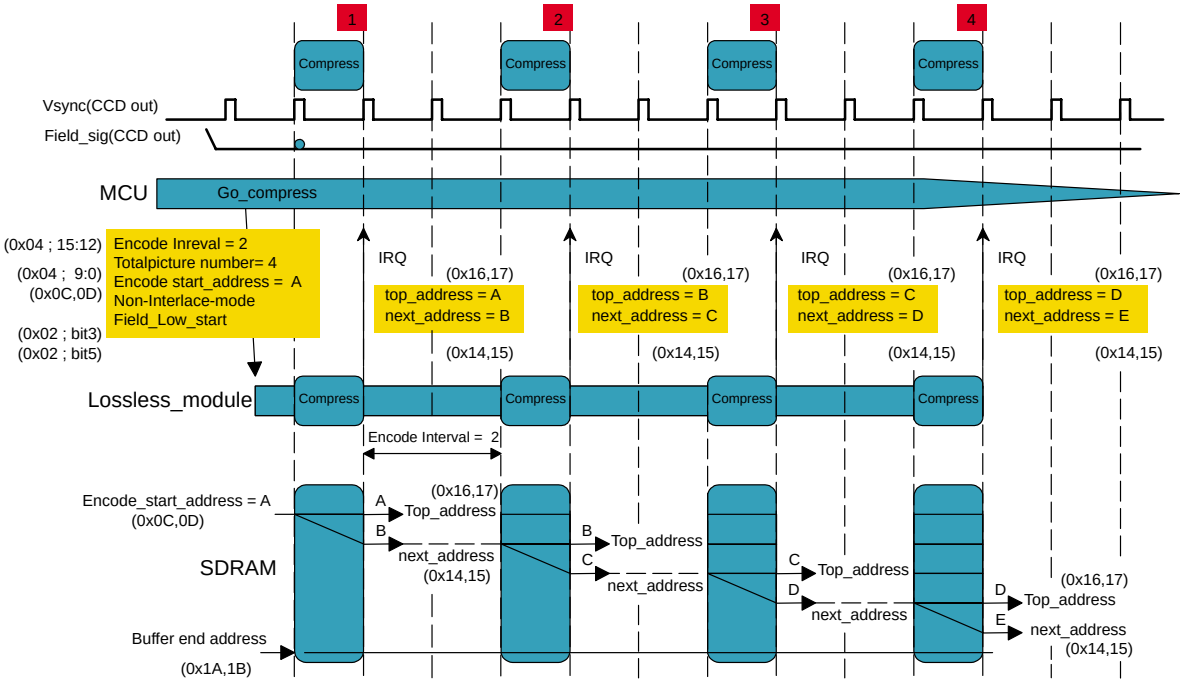


Figure 4–27. Burst Compression Interrupt in Noninterlaced Mode

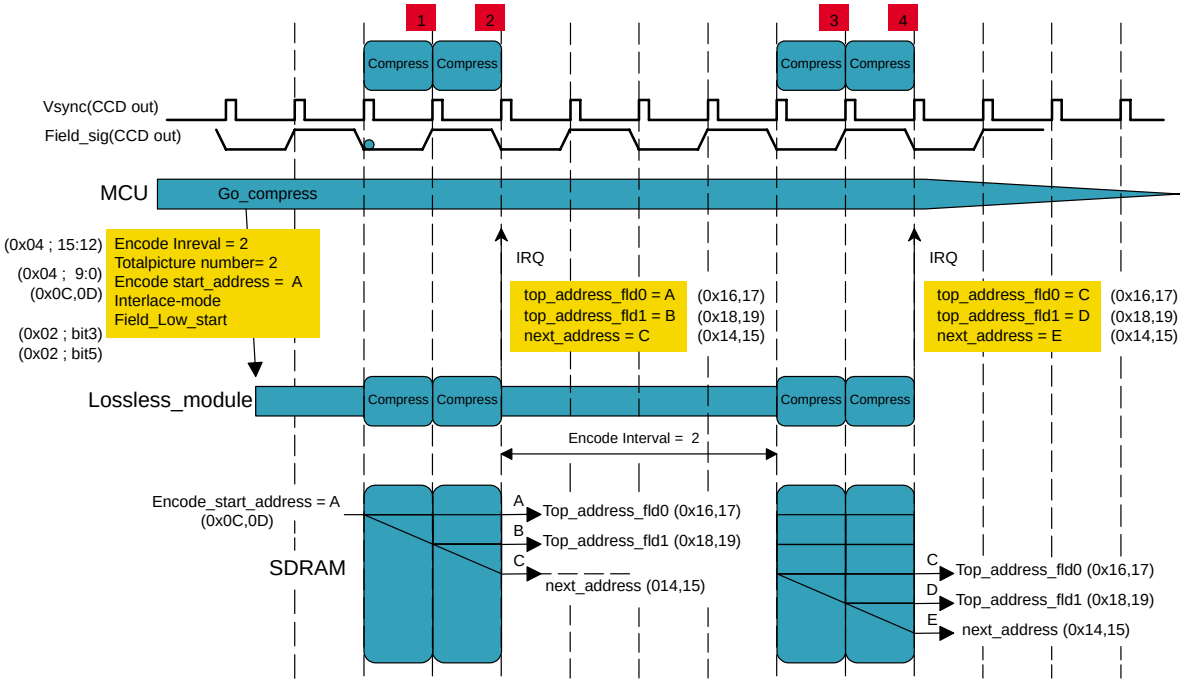


Figure 4–28. Burst Compression Interrupt in Interlaced Mode

4.3.4 Burst Capture Mode Decompression

The decompression engine has three main modules as inverse functions of the compression engine modules: Huffman decoding, inverse DPCM, and dequantization.

In the expansion process, the bit stream is loaded from SDRAM and RAW data expanded by IDPCM and Huffman decoding is stored in another SDRAM region. Expansion of one image's worth of data is performed automatically because the bit-stream data storage destination is known from the MCU. When one image's worth of data is expanded, an interrupt is generated to the MCU. Also, during expansion, it is possible to perform a partial expansion process of every few lines.

4.3.4.1 Huffman Decoding

While the Huffman decoder performs the inverse function of the Huffman encoder, it is more complex than the encoder is. The decompression is not a real-time hardware.

4.3.4.2 Inverse DPCM

Inverse DPCM recovers the quantization indices corresponding to the quantized pixel values. Its operation is depicted in the block diagram of Figure 4–29.

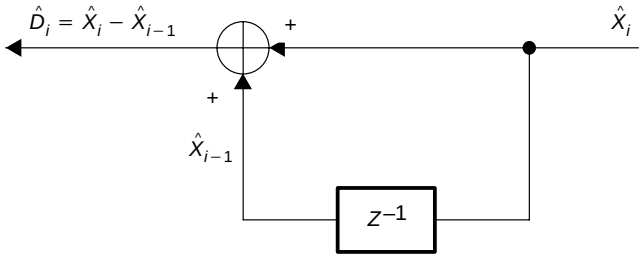


Figure 4–29. Block Diagram of Inverse DPCM

4.3.4.3 Dequantization

The dequantization reconstructs the CCD raw data at the pixel values. The dequantization module right shifts the data coming out of the inverse DPCM engine into a 10-bit data format.

4.3.4.4 Configuration of the Decompression Engine

The burst codec decompresses only one image at a time. The software needs to start the decompression engine for each image. This is done by writing a 1 into bit 0 of the BURSTEN register. This bit is cleared by the system when the decompression is finished.

Table 4–62. Decompression Engine Registers

OFFSET	ADDRESS	REGISTER NAME	
0x01	0x0003:0802	BURSTDEC	Burst decoder control
0x0A	0x0003:0814	DECMPLN	Number of decompressed lines

Table 4–63. Number of Decompressed Lines (DECMPLN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	DLN11	DLN10	DLN9	DLN8	DLN7	DLN6	DLN5	DLN4	DLN3	DLN2	DLN1	DLN0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–64. Number of Decompressed Lines (DECMPLN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	DLN11–DLN0	Sets number of decompressed lines (number of lines after decompression).

Table 4–65. Burst Decoder Control (BURSTDEC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	ERRST	RSV	RSV	RSV	RSV	RSV	RSV	DECN	DECF
R/W	—	—	—	—	—	—	—	R	—	—	—	—	—	—	R/W	R/W
Default	—	—	—	—	—	—	—	0	—	—	—	—	—	—	0	0

Table 4–66. Burst Decoder Control (BURSTDEC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	ERRST	Error status Becomes 1 when error occurs. An error occurs when Huffman code out-of-table occurs.
7–2	RSV	Reserved bits. Not used.
1	DECN	Decode enable next bit When this register is set to 1, decoding begins and when decoding of the number of lines set in the DECMPLN register is finished, it is automatically cleared. 0: Disabled 1: Enabled
0	DECF	Decode enable first bit When this register is set to 1, decoding begins and when decoding of the number of lines set in the DECMPLN register is finished, it is automatically cleared. 0: Disabled 1: Enabled

The SDRAM start address registers (SDSTAUP and SDSTALO) need to be loaded with the starting address of the compressed data. The SDRAM work address registers (WORKA_HI and WORKA_LO) define where the decompressed image has to be stored within the SDRAM.

4.4 Burst Decompression Flow

4.4.1 Compression Results

This section shows the results of lossless/lossy compression of several CCD raw data files using the burst capture mode compression engine. The compression ratio obtained from the codec is highly dependent on the image itself. The user should use the following figures only as an estimate. On different raw images, the compression ratio could be better or worse.

If a 10-bit CCD sensor is used for a 1288x968 pixels image, 1288x968x2=2,493,568 bytes are necessary to store the raw data. If a bit-packing scheme were used, only 1,558,480 bytes would be used. However, if the compression engine is used in a lossless mode, the size of the compressed data generated can be reduced to something around 1,050,000 bytes. The compression factor is around 2.35.

Table 4–67 gives some average results obtained with a 10-bit CCD sensor. The peak signal-to-reconstructed image (PSNR) gives an estimate of the quality of the reconstructed image compared with the original image.

Table 4–67. Typical Compression Results

NUMBER OF BITS USED	TYPICAL COMPRESSION RATIO RANGE	TYPICAL PSNR RANGE (dB)
10	1.75 to 3.25	∝
9	2.75 to 4.25	62.5 to 63.75
8	3.75 to 5.25	54.5 to 55.25

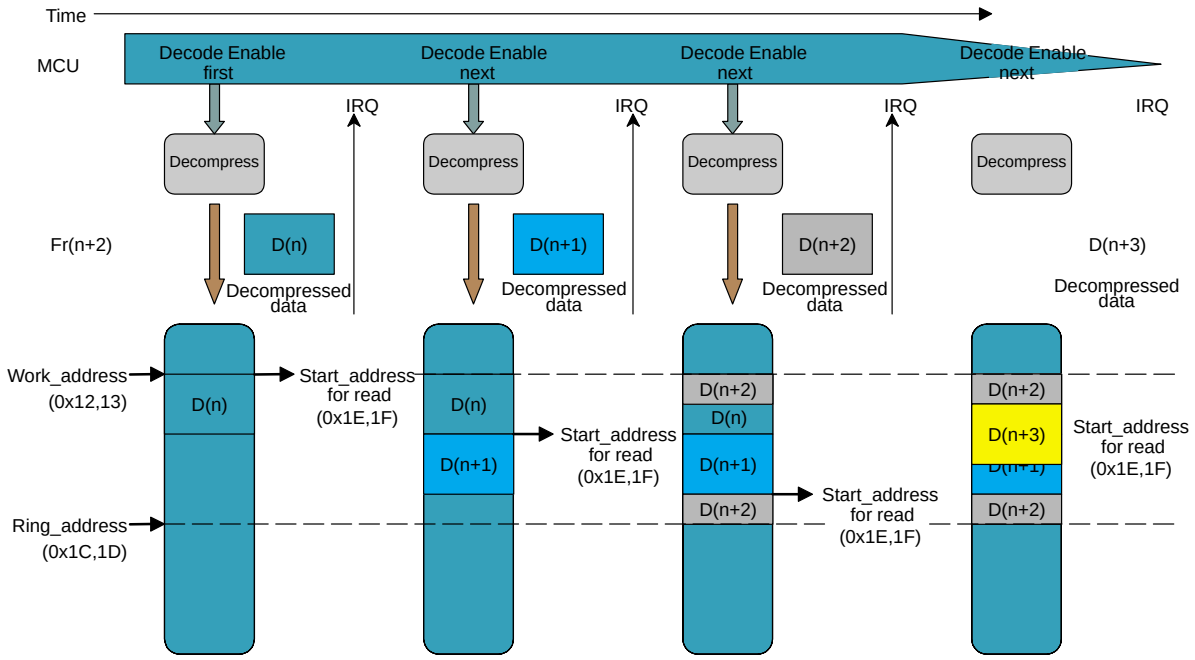


Figure 4–30. Burst Decompression Flow

4.4.2 SDRAM Pointers Used by the Burst Codec

The burst codec needs to be configured to know where to store or retrieve the data within the SDRAM. The following registers are used to specify these addresses.

Table 4–68. Burst Codec Registers

OFFSET	REGISTER ADDRESS	REGISTER NAME	
0x0C	0x0003:0818	ENC_SDSTA_HI	Encode SDRAM start address high
0x0D	0x0003:081A	ENC_SDSTA_LO	Encode SDRAM start address low
0x0E	0x0003:081C	DEC_SDSTA_FLD0_HI	Field 0 decoder SDRAM start address high
0x0F	0x0003:081E	DEC_SDSTA_FLD0_LO	Field 0 decoder SDRAM start address low
0x010	0x0003:0820	DEC_SDSTA_FLD1_HI	Field 1 decoder SDRAM start address high
0x011	0x0003:0822	DEC_SDSTA_FLD1_LO	Field 1 decoder SDRAM start address low
0x012	0x0003:0824	WORKA_HI	SDRAM work address high
0x013	0x0003:0826	WORKA_LO	SDRAM work address low
0x014	0x0003:0828	NEXT_PICTPA_HI	SDRAM next picture address high
0x015	0x0003:082A	NEXT_ PICTPA_LO	SDRAM next picture address low
0x016	0x0003:082C	PICTPA_FLD0_HI	Field 0 picture address high
0x017	0x0003:082E	PICTPA_FLD0_LO	Field 0 picture address low
0x018	0x0003:0830	PICTPA_FLD1_HI	Field 1 picture address high
0x019	0x0003:0832	PICTPA_FLD1_LO	Field 1 picture address low
0x01A	0x0003:0834	ENCA_HI	SDRAM encode buffer end address high
0x01B	0x0003:0836	ENCA_LO	SDRAM encode buffer end address low
0x01C	0x0003:0838	RINGA_HI	Ring buffer address high
0x01D	0x0003:083A	RINGA_LO	Ring buffer address low
0x01E	0x0003:083C	DECOMA_HI	Decompressed image address up
0x01F	0x0003:083E	DECOMA_LO	Decompressed image address low

Table 4–69. Encode SDRAM Start Address High (ENC_SDSTA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	UAD5	UAD4	UAD3	UAD2	UAD1	UAD0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–70. Encode SDRAM Start Address High (ENC_SDSTA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	UAD5–UAD0	Specifies upper lead address of SDRAM (in compression). Specifies lead address at which encoded data is stored. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–71. Encode SDRAM Start Address LOW (ENC_SDSTA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LAD15	LAD14	LAD13	LAD12	LAD11	LAD10	LAD9	LAD8	LAD7	LAD6	LAD5	LAD4	LAD3	LAD2	LAD1	LAD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–72. Encode SDRAM Start Address Low (ENC_SDSTA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	LAD15–LAD0	Specifies lower lead address of SDRAM (in compression). Specifies lead address at which encoded data is stored. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–73. Field 0 Decoder SDRAM Start Address High (DEC_SDSTA_FLD0_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	D0U5	D0U4	D0U3	D0U2	D0U1	D0U0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–74. Field 0 Decoder SDRAM Start Address High (DEC_SDSTA_FLD0_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	D0U5–D0U0	Specifies lead address of SDRAM (field 0, upper) (in decompression). Specifies lead address at which encoded data is stored. When set to non-interlace: Only DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO is valid. When set to interlace: Lead address of field 0 is specified by DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO Lead address of field 1 is specified by: DEC_SDSTA_FLD1_HI, DEC_SDSTA_FLD1_LO Data after decompression is stored alternately in field 0 and field 1 for each line. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–75. Field 0 Decoder SDRAM Start Address Low (DEC_SDSTA_FLD0_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	D0L15	D0L14	D0L13	D0L12	D0L11	D0L10	D0L9	D0L8	D0L7	D0L6	D0L5	D0L4	D0L3	D0L2	D0L1	D0L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–76. Field 0 Decoder SDRAM Start Address Low (DEC_SDSTA_FLD0_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	D0L15–D0L0	Specifies lead address of SDRAM (field 0, lower) (in decompression). Specifies lead address at which encoded data is stored. When set to non-interlace: Only DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO is valid. When set to interlace: Lead address of field 0 is specified by: DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO Lead address of field 1 is specified by: DEC_SDSTA_FLD1_HI, DEC_SDSTA_FLD1_LO Data after decompression is stored alternately in field 0 and field 1 for each line. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–77. Field 1 Decoder SDRAM Start Address High (DEC_SDSTA_FLD1_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	D1U5	D1U4	D1U3	D1U2	D1U1	D1U0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–78. Field 1 Decoder SDRAM Start Address High (DEC_SDSTA_FLD1_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	D1U5–D1U0	Specifies lead address of SDRAM (field 1, upper) (in decompression). Specifies lead address at which encoded data is stored. When set to non-interlace: Only DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO is valid. When set to interlace: Lead address of field 0 is specified by: DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO Lead address of field 1 is specified by: DEC_SDSTA_FLD1_HI, DEC_SDSTA_FLD1_LO Data after decompression is stored alternately in field 0 and field 1 for each line. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–79. Field 1 Decoder SDRAM Start Address Low (DEC_SDSTA_FLD1_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	D1L15	D1L14	D1L13	D1L12	D1L11	D1L10	D1L9	D1L8	D1L7	D1L6	D1L5	D1L4	D1L3	D1L2	D1L1	D1L0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–80. Field 1 Decoder SDRAM Start Address Low (DEC_SDSTA_FLD1_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	D1L15–D1L0	Specifies lead address of SDRAM (field 1, lower) (in decompression). Specifies lead address at which encoded data is stored. When set to non-interlace: Only DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO is valid. When set to interlace: Lead address of field 0 is specified by DEC_SDSTA_FLD0_HI, DEC_SDSTA_FLD0_LO Lead address of field 1 is specified by DEC_SDSTA_FLD1_HI, DEC_SDSTA_FLD1_LO Data after decompression is stored alternately in field 0 and field 1 for each line. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–81. SDRAM Work Address High (WORKA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WU5	WU4	WU3	WU2	WU1	WU0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–82. SDRAM Work Address High (WORKA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	WU5–WU0	Specifies address (upper) of SDRAM (in decompression). Decoded image data is stored at address specified here. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–83. SDRAM Work Address Low (WORKA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	WL15	WL14	WL13	WL12	WL11	WL10	WL9	WL8	WL7	WL6	WL5	WL4	WL3	WL2	WL1	WL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–84. SDRAM Work Address Low (WORKA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	WL15–WL0	Specifies address (lower) of SDRAM (in decompression). Decoded image data is stored at address specified here. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–85. SDRAM Next Picture Address High (NEXT_PICTPA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	PU5	PU4	PU3	PU2	PU1	PU0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–86. SDRAM Next Picture Address High (NEXT_PICTPA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	PU5–PU0	Specifies lead address (upper) of SDRAM (in decompression). In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–87. SDRAM Next Picture Address Low (NEXT_PICTPA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PL15	PL14	PL13	PL12	PL11	PL10	PL9	PL8	PL7	PL6	PL5	PL4	PL3	PL2	PL1	PL0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–88. SDRAM Next Picture Address Low (NEXT_PICTPA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	PL15–PL0	Specifies lead address (lower) of SDRAM (in decompression). In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–89. Field 0 Picture Address High (PICTPA_FLD0_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	P0U5	P0U4	P0U3	P0U2	P0U1	P0U0
R/W	—	—	—	—	—	—	—	—	—	—	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–90. Field 0 Picture Address High (PICTPA_FLD0_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	P0U5–P0U0	The lead address (field 0, upper) of SDRAM is displayed (in decompression). In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–91. Field 0 Picture Address Low (PICTPA_FLD0_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	P0L15	P0L14	P0L13	P0L12	P0L11	P0L10	P0L9	P0L8	P0L7	P0L6	P0L5	P0L4	P0L3	P0L2	P0L1	P0L0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–92. Field 0 Picture Address Low (PICTPA_FLD0_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	P0L15–P0L0	Lead address (field 0, lower) of SDRAM is displayed (in decompression). In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–93. Field 1 Picture Address High (PICTPA_FLD1_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	P1U5	P1U4	P1U3	P1U2	P1U1	P1U0
R/W	—	—	—	—	—	—	—	—	—	—	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–94. Field 1 Picture Address High (PICTPA_FLD1_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	P1U5–P1U0	Lead address (field 1, upper) of SDRAM is displayed (in decompression). In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–95. Field 1 Picture Address Low (PICTPA_FLD0_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	P1L15	P1L14	P1L13	P1L12	P1L11	P1L10	P1L9	P1L8	P1L7	P1L6	P1L5	P1L4	P1L3	P1L2	P1L1	P1L0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–96. Field 1 Picture Address Low (PICTPA_FLD0_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	P1L15–P1L0	Lead address (field 1, lower) of SDRAM is displayed. In encoding, each time one piece of encoding is finished, the lead address at which the next image data (compressed data) is stored is uploaded. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–97. SDRAM Encode Buffer End Address High (ENCA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	ENU5	ENU4	ENU3	ENU2	ENU1	ENU0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	1	1	1	1	1	1

Table 4–98. SDRAM Encode Buffer End Address High (ENCA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	ENU5–ENU0	Specifies final address (upper) of SDRAM. In encoding, specifies final address at which encoded data is stored. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–99. SDRAM Encode Buffer End Address Low (ENCA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ENL15	ENL14	ENL13	ENL12	ENL11	ENL10	ENL9	ENL8	ENL7	ENL6	ENL5	ENL4	ENL3	ENL2	ENL1	ENL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 4–100. SDRAM Encode Buffer End Address Low (ENCA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	ENL15–ENL0	Specifies final address (lower) of SDRAM. In encoding, specifies final address at which encoded data is stored. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–101. Ring Buffer Address High (RINGA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RNU5	RNU4	RNU3	RNU2	RNU1	RNU0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	1	1	1	1	1	1

Table 4–102. Ring Buffer Address High (RINGA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used
5–0	RNU5–RNU0	Specifies upper address of SDRAM (in decompression). Decoded image data may not be stored beyond the address specified here. Ring buffering is performed in a region set by the WORKA_HI (0003:0824), WORKA_LO (0003:0826), RINGA_HI (0003:0838), and RINGA_LO (0003:083A) registers. After data is stored in RINGA_HI (0003:0838), RINGA_LO (0003:083A), the next data is stored in WORKA_HI (0003:0824) and WORKA_LO (0003:0826). If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–103. Ring Buffer Address Low (RINGA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RNL15	RNL14	RNL13	RNL12	RNL11	RNL10	RNL9	RNL8	RNL7	RNL6	RNL5	RNL4	RNL3	RNL2	RNL1	RNL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 4–104. Ring Buffer Address LO (RINGA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	RNL15–RNL0	Specifies lower address of SDRAM (in decompression). Decoded image data may not be stored beyond the address specified here. Ring buffering is performed in region set by the WORKA_HI (0003:0824), WORKA_LO (0003:0826), RINGA_HI (0003:0838), and RINGA_LO (0003:083A) registers. After data is stored in RINGA_HI (0003:0838) and RINGA_LO (0003:083A), the next data is stored in WORKA_HI (0003:0824), WORKA_LO (0003:0826). If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–105. Decompressed Image Address Up (DECOMA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	DMU5	DMU4	DMU3	DMU2	DMU1	DMU0
R/W	—	—	—	—	—	—	—	—	—	—	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–106. Decompressed Image Address Up (DECOMA_HI) Register Bit/Field Descriptions

15–6	RSV	Reserved. Not used
5–0	DMU5–DMU0	Upper lead address of SDRAM is displayed (in decompression). In decompression, the lead address at which decoded data (decompressed data) is stored is displayed. Decompressed data is stored in order from the address specified here. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

Table 4–107. Decompressed Image Address Low (DECOMA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DML15	DML14	DML13	DML12	DML11	DML10	DML9	DML8	DML7	DML6	DML5	DML4	DML3	DML2	DML1	DML0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–108. Decompressed Image Address Low (DECOMA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	DML15–DML0	Lower lead address of SDRAM is displayed (in decompression). In decompression, the lead address at which decoded data (decompressed data) is stored is displayed. Decompressed data is stored in order from the address specified here. If a 32-bit wide SDRAM is selected, a value eight times the value set here is the actual SDRAM address. If a 16-bit wide SDRAM is selected, a value 16 times the value set here is the actual SDRAM address.

4.5 On-Screen Display and Graphics Acceleration

The on-screen display module (OSD) manages OSD data from the different bitmap windows, mixes it with video data, and outputs it. This module reads OSD data from SDRAM, synchronizes it to input HD/VD, and outputs it in YCbCr format.

4.5.1 Specifications

The OSD has the following characteristics:

- Two video display windows and two OSD display windows exist.
- A square cursor can be displayed, its size can be specified, and a look-up table is used to specify its color.
- The four windows and the cursor can be displayed simultaneously.
- The background color can be specified.
- YCbCr data type is used to display video-Window0 and Video-Window1. OSD-Window0 and OSD-Window1 are displayed by bitmap.
- 1-, 2-, 4-, 8-bit width can be selected for bitmap.
- Five blending ratios for the OSD-windows can be selected.
- Default bitmap look-up table in ROM available.
- Programmable bitmap look-up table in dedicated RAM available (24 bits x 256 words).
- Interlaced and noninterlaced modes are supported.
- Zoom horizontal and vertical with factor x2 and x4 available.
- VGA->NTSC/PAL expansion filters built-in.

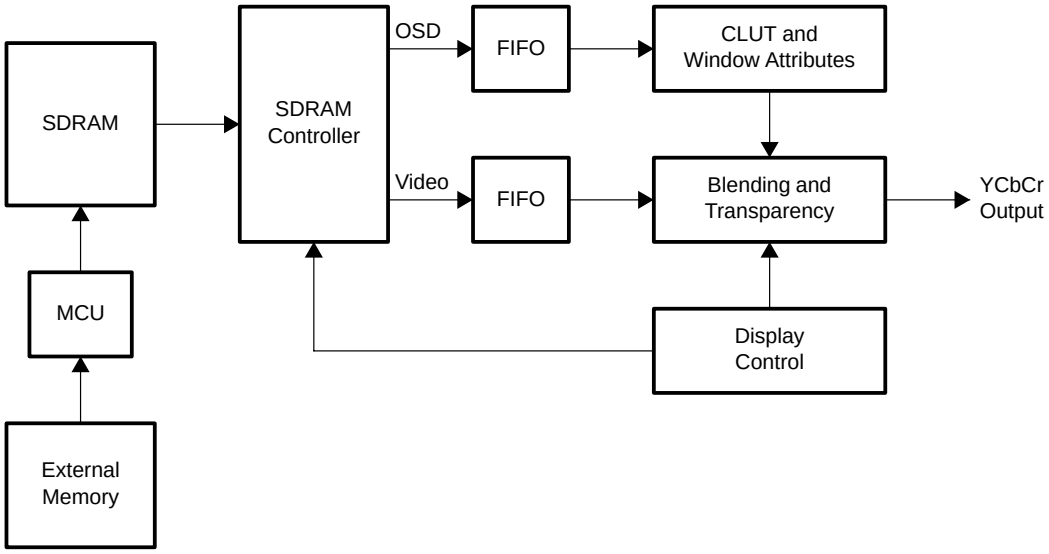


Figure 4–31. DSC24 OSD Block

4.5.2 OSD Data Format

Two different window formats are supported, in addition, the OSD data has a variable size. In the bitmap windows, each pixel can be 1, 2, 4, or 8 bits wide. In the YCbCr 4:2:2 window, it takes 8 bits per component and the components are arranged according to 4:2:2 (Cb/Y/Cr/Y) format, 16 bits per pixel. In the case where RGB graphics data needs to be used as OSD, the application should perform software conversion to YCbCr before storing it. The OSD data is always packed into 32-bit words and left justified. Starting from the upper left corner of the OSD window, all data is packed into adjacent 32-bit words.

Table 4–109. Pixel Location Within Each Window

P0	P1	P2	P3	P4	P5	P6	P7	…	Pn
Pn+1	Pn+2	Pn+3	Pn+4	Pn+5	Pn+5	Pn+6	Pn+7	…	P2n
:	:	:	:	:	:	:	:	:	:

Table 4–110. YCbCr 4:2:2 Data Format of Video Data

Bit	31 24	23 16	15 8	7 0
Pixel	Y1	Cr0	Y0	Cb0

Table 4–111. SDRAM Usage for Video Data

ADDRESS	MSB	LSB
N	P1	P0
N+1	P3	P2
N+2	P5	P4
N+3	:	:

Table 4–112. 8-Bit Bitmap Data Format

Bit	31 24	23 16	15 8	7 0
Pixel	P3	P2	P1	P0

Table 4–113. 4-Bit Bitmap Data Format

Bit	31 28	27 24	23 20	19 16	15 12	11 8	7 4	3 0
Pixel	P6	P7	P4	P5	P2	P3	P0	P1

Table 4–114. 2-Bit Bitmap Data Format

Bit	31 30	…	25 24	23 22	…	17 16	15 14	…	9 8	7 6	…	1 0
Pixel	P12	…	P16	P8	…	P11	P4	…	P7	P0	…	P3

Table 4–115. 1-Bit Bitmap Data Format

Bit	31	…	24	23	…	16	15	…	8	7	…	0
Pixel	P24	…	P31	P16	…	P23	P8	…	P15	P0	…	P7

4.5.3 SDRAM Location

The location were the data is stored within the SDRAM is defined by memory map registers.

Table 4–116. SDRAM Access Control Registers

OFFSET	HEX ADDRESS	REGISTER NAME	
0x06	0x0003:068C	VIDEOWIN0_OFFSET	Video window 0 offset
0x07	0x0003:068E	VIDEOWIN1_OFFSET	Video window 1 offset
0x08	0x0003:0690	OSDWIN0_OFFSET	OSD window 0 offset
0x09	0x0003:0692	OSDWIN1_OFFSET	OSD window 1 offset
0x0A	0x0003:0694	VIDEOWIN0_HI	Video window 0 address high
0x0B	0x0003:0696	VIDEOWIN0_LO	Video window 0 address low
0x0C	0x0003:0698	VIDEOWIN1_HI	Video window 1address high
0x0D	0x0003:069A	VIDEOWIN1_LO	Video window 1address low
0x0E	0x0003:069C	OSDWIN0_HI	OSD window 0 address high
0x0F	0x0003:069E	OSDWIN0_LO	OSD window 0 address low
0x10	0x0003:06A0	OSDWIN1_HI	OSD window 1 address high
0x11	0x0003:06A2	OSDWIN1_LO	OSD window 1 address low
0x3D	0x0003:06FA	VIDEOWIN0_PPC	Video window 0 ping-pong control
0x3E	0x0003:06FC	VIDEOWIN0_PPB_HI	Video window 0 ping-pong buffer address high
0x3F	0x0003:06FE	VIDEOWIN0_PPB_LO	Video window 0 ping-pong buffer address low

Table 4–117. Video Window 0 Offset (VIDEOWIN0_OFFSET) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	V0LO8	V0LO7	V0LO6	V0LO5	V0LO4	V0LO3	V0LO2	V0LO1	V0LO0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–118. Video Window 0 Offset (VIDEOWIN0_OFFSET) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	V0LO8–V0LO0	Specifies 1-line offset in video window 0. Specifies address distance from next line stored in SDRAM by a number divided by 8.

Table 4–119. Video Window 1 Offset (VIDEOWIN1_OFFSET) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	V1LO8	V1LO7	V1LO6	V1LO5	V1LO4	V1LO3	V1LO2	V1LO1	V1LO0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–120. Video Window 1 Offset (VIDEOWIN1_OFFSET) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	V1LO8–V1LO0	Specifies 1-line offset in video window 1. Specifies address distance from next line stored in SDRAM by a number divided by 8.

Table 4–121. OSD Window 0 Offset (OSDWIN0_OFFSET) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	O0LO8	O0LO7	O0LO6	O0LO5	O0LO4	O0LO3	O0LO2	O0LO1	O0LO0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–122. OSD Window 0 Offset (OSDWIN0_OFFSET) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	O0LO8–O0LO0	Specifies 1-line offset in OSD window 0. Specifies address distance from next line stored in SDRAM by a number divided by 8.

Table 4–123. OSD Window 1 Offset (OSDWIN1_OFFSET) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	O1LO8	O1LO7	O1LO6	O1LO5	O1LO4	O1LO3	O1LO2	O1LO1	O1LO0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–124. OSD Window 1 Offset (OSDWIN1_OFFSET) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	O1LO8–O1LO0	Specifies 1-line offset in OSD window 1. Specifies address distance from next line stored in SDRAM by a number divided by 8.

Table 4–125. Video Window Address High (VIDEOWINx_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VxA21	VxA20	VxA19	VxA18	VxA17	VxA16
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–126. Video Window Address High (VIDEOWINx_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	VxA21–VxA16	Specifies lead address of SDRAM of image data displayed by video window. Here, address 1 is offset from SDRAM area lead in 32-byte units.

Table 4–127. Video Window Address Low (VIDEOWINx_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	VxA15	VxA14	VxA13	VxA12	VxA11	VxA10	VxA9	VxA8	VxA7	VxA6	VxA5	VxA4	VxA3	VxA2	VxA1	VxA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–128. Video Window Address Low (VIDEOWINx_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	VxA15–VxA0	Specifies lower 16 bits of the lead address of SDRAM of image data (YCbCr) displayed by the video window. Here, address 1 is offset from the SDRAM area lead in 32-byte units.

Table 4–129. OSD Window 0 Address High (OSDWINx_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WAx21	WAx20	WAx19	WAx18	WAx17	WAx16
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–130. OSD Window 0 Address High (OSDWINx_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	WAx21–WAx16	OSD window 0 address Specifies the upper 6 bits of the lead address of SDRAM of image data (YCbCr) displayed by OSD window 0. Here, address 1 is offset from the SDRAM area lead in 32-byte units.

Table 4–131. OSD Window 0 Address Low (OSDWINx_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	WAx15	WAx14	WAx13	WAx12	WAx11	WAx10	WAx09	WAx08	WAx07	WAx06	WAx05	WAx04	WAx03	WAx02	WAx01	WAx00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–132. OSD Window 0 Address Low (OSDWINx_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	WAx15–WAx0	OSD window 0 address. Specifies the lower 16 bits of the lead address of SDRAM of image data (YCbCr) displayed by OSD window. Here, address 1 is offset from the SDRAM area lead in 32-byte units.

4.5.4 Main Video Window Ping-Pong Buffers

The main video window (video window 0) supports ping-pong buffers. The VIDEOWIN0_HI and VIDEOWIN0_LO registers point to the first one of the ping-pong buffers. The VIDEOWIN0_PPB_HI and VIDEOWIN0_PPB_LO registers described below point to the second one.

In the control register (VIDEOWIN0_PPC) associated with this feature, one of the bits (ABSEL) is used to switch between the two buffers.

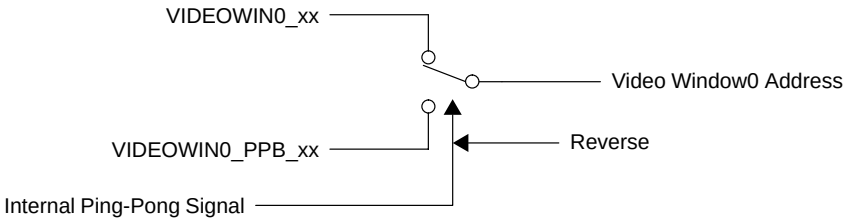


Figure 4–32. Ping-Pong Buffers for the Main Video Window

Table 4–133. Video Window 0 Ping-Pong Control (VIDEOWIN0_PPC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	ABSEL	REV	RSV
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 4–134. Video Window 0 Ping-Pong Control (VIDEOWIN0_PPC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved bits. Not used.
2	ABSEL	Selects or indicates which buffer is used 0: Buffer A (addressed by VIDEOWIN0_xx) 1: Buffer B (addressed by VIDEOWIN0_PPB_xx)
1	REV	Inverts polarity of A/B address select signal 0: Normal 1: Inverted
0	RSV	Reserved bit. Always 0

Table 4–135. Video Window 0 Ping-Pong Buffer Address High (VIDEOWIN0_PPB_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	V0PA21	V0PA20	V0PA19	V0PA18	V0PA17	V0PA16
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–136. Video Window 0 Ping-Pong Buffer Address High (VIDEOWIN0_PPB_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits
5–0	V0PA21–V0PA16	Upper address of video window 0 of the ping-pong buffer.

Table 4–137. Video Window 0 Ping-Pong Buffer Address Low (VIDEOWIN0_PPB_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	V0PA15	V0PA14	V0PA13	V0PA12	V0PA11	V0PA10	V0PA09	V0PA08	V0PA07	V0PA06	V0PA05	V0PA04	V0PA03	V0PA02	V0PA01	V0PA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–138. Video Window 0 Ping-Pong Buffer Address Low (VIDEOWIN0_PPB_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	V0PA15–V0PA00	Lower address of video window 0 of the ping-pong buffer.

4.5.5 Window Priority

If overlapping occurs between different windows, the color of each pixel corresponds to the color of the window with the highest priority. The priority order is fixed.

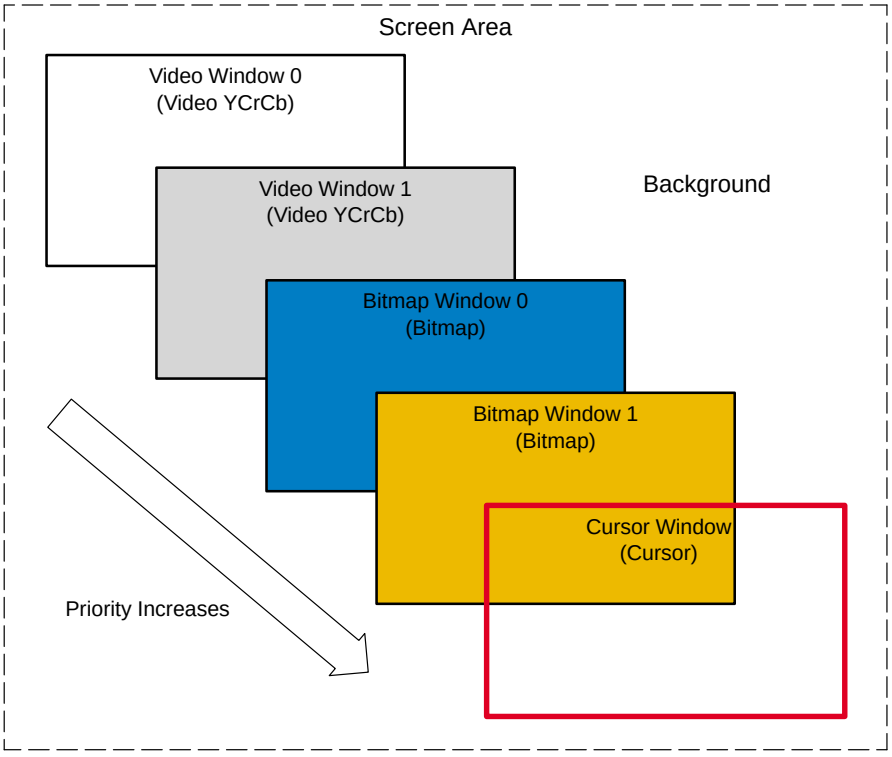


Figure 4–33. Window Priorities

4.5.6 Color Look-Up Table

A color look-up table (CLUT) is used to convert the bitmap information of the OSD window pixels into Y, Cb, and Cr values. There is a default CLUT stored in the ROM of the chip. The user can also create a CLUT and store it within a special RAM block of the imaging peripherals subsystem. If a lower resolution is used (1 to 4 bits per pixel), the 16 registers BITMAPxx(WINy) can be used to specify which of the 256 CLUT entries need to be used.

Table 4–139. Color Loop-Up

OFFSET	HEX ADDRESS	REGISTER NAME	
0x28	0x0003:06D0	W0BMP01	OSD window bitmap colors 0 and 1
0x29	0x0003:06D2	W0BMP23	OSD window bitmap colors 2 and 3
0x2A	0x0003:06D4	W0BMP45	OSD window bitmap colors 4 and 5
0x2B	0x0003:06D6	W0BMP67	OSD window bitmap colors 6 and 7
0x2C	0x0003:06D8	W0BMP89	OSD window bitmap colors 8 and 9
0x2D	0x0003:06DA	W0BMPAB	OSD window bitmap colors A and B
0x2E	0x0003:06DC	W0BMPCD	OSD window bitmap colors C and D
0x2F	0x0003:06DE	W0BMPEF	OSD window bitmap colors E and F
0x30	0x0003:06E0	W1BMP01	OSD window bitmap colors 0 and 1
0x31	0x0003:06E2	W1BMP23	OSD window bitmap colors 2 and 3
0x32	0x0003:06E4	W1BMP45	OSD window bitmap colors 4 and 5
0x33	0x0003:06E6	W1BMP67	OSD window bitmap colors 6 and 7
0x34	0x0003:06E8	W1BMP89	OSD window bitmap colors 8 and 9
0x35	0x0003:06EA	W1BMPAB	OSD window bitmap colors A and B
0x36	0x0003:06EC	W1BMPCD	OSD window bitmap colors C and D
0x37	0x0003:06EE	W1BMPEF	OSD window bitmap colors E and F
0x3A	0x0003:06F4	CLUT_RAM_CTRL	RAM color look-up table control
0x3B	0x0003:06F6	CLUT_RAM_YCB	RAM color look-up table Y and Cb data
0x3C	0x0003:06F8	CLUT_RAM_CR	RAM color look-up table Cr data and address

When less than 8 bits are used for the bitmap windows, the WxBMPxx registers are used as pointer on the CLUT (either RAM or ROM) to specify which color is used.

Table 4–140. OSD Window Bitmap Colors 0 and 1 (WxBMP01) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDx17	OSDx16	OSDx15	OSDx14	OSDx13	OSDx12	OSDx11	OSDx10	OSDx07	OSDx06	OSDx05	OSDx04	OSDx03	OSDx02	OSDx01	OSDx00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–141. OSD Window Bitmap Colors 0 and 1 (WxBMP01) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDx17–OSDx10	Specifies color of bitmap value 1 by address of CLUT (0–255). (width = 4)
7–0	OSDx07–OSDx00	Specifies color of bitmap value 0 by address of CLUT (0–255). (width = 1, 2, 4)

Table 4–142. OSD Window Bitmap Colors 2 and 3 (WxBMP23) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDx37	OSDx36	OSDx35	OSDx34	OSDx33	OSDx32	OSDx31	OSDx30	OSDx27	OSDx26	OSDx25	OSDx24	OSDx23	OSDx22	OSDx21	OSDx20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–143. OSD Window Bitmap Colors 2 and 3 (WxBMP23) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDx37–OSDx30	Specifies color of bitmap value 3 by address of CLUT (0–255). (width = 4)
7–0	OSDx27–OSDx20	Specifies color of bitmap value 2 by address of CLUT (0–255). (width = 4)

Table 4–144. OSD Window Bitmap Colors 4 and 5 (WxBMP45) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDx57	OSDx56	OSDx55	OSDx54	OSDx53	OSDx52	OSDx51	OSDx50	OSDx47	OSDx46	OSDx45	OSDx44	OSDx43	OSDx42	OSDx41	OSDx40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–145. OSD Window Bitmap Colors 4 and 5 (WxBMP45) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDx57–OSDx50	Specifies color of bitmap value 5 by address of CLUT (0–255). (width = 2, 4)
7–0	OSDx47–OSDx40	Specifies color of bitmap value 4 by address of CLUT (0–255). (width = 4)

Table 4–146. OSD Window Bitmap Colors 6 and 7 (WxBMP67) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDx77	OSDx76	OSDx75	OSDx74	OSDx73	OSDx72	OSDx71	OSDx70	OSDx67	OSDx66	OSDx65	OSDx64	OSDx63	OSDx62	OSDx61	OSDx60
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–147. OSD Window Bitmap Colors 6 and 7 (WxBMP67) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDx77–OSDx70	Specifies color of bitmap value 7 by address of CLUT (0–255). (width = 4)
7–0	OSDx67–OSDx60	Specifies color of bitmap value 6 by address of CLUT (0–255). (width = 4)

Table 4–148. OSD Window Bitmap Colors 8 and 9 (WxBMP89) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDx97	OSDx96	OSDx95	OSDx94	OSDx93	OSDx92	OSDx91	OSDx90	OSDx87	OSDx86	OSDx85	OSDx84	OSDx83	OSDx82	OSDx81	OSDx80
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–149. OSD Window Bitmap Colors 8 and 9 (WxBMP89) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDx97–OSDx90	Specifies color of bitmap value 9 by address of CLUT (0–255). (width = 4)
7–0	OSDx87–OSDx80	Specifies color of bitmap value 8 by address of CLUT (0–255). (width = 4)

Table 4–150. OSD Window Bitmap Colors A and B (WxBMPAB) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDxB7	OSDxB6	OSDxB5	OSDxB4	OSDxB3	OSDxB2	OSDxB1	OSDxB0	OSDxA7	OSDxA6	OSDxA5	OSDxA4	OSDxA3	OSDxA2	OSDxA1	OSDxA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–151. OSD Window Bitmap Colors A and B (WxBMPAB) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDxB7–OSDxB0	Specifies color of bitmap value B by address of CLUT (0–255). (width = 4)
7–0	OSDxA7–OSDxA0	Specifies color of bitmap value A by address of CLUT (0–255). (width = 2, 4)

Table 4–152. OSD Window Bitmap Colors C and D (WxBMPCD) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDxD7	OSDxD6	OSDxD5	OSDxD4	OSDxD3	OSDxD2	OSDxD1	OSDxD0	OSDxC7	OSDxC6	OSDxC5	OSDxC4	OSDxC3	OSDxC2	OSDxC1	OSDxC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–153. OSD Window Bitmap Colors C and D (WxBMPCD) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDxD7–OSDxD0	Specifies color of bitmap value D by address of CLUT (0–255). (width = 4)
7–0	OSDxC7–OSDxC0	Specifies color of bitmap value C by address of CLUT (0–255). (width = 4)

Table 4–154. OSD Window Bitmap Colors E and F (WxBMPEF) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OSDxF7	OSDxF6	OSDxF5	OSDxF4	OSDxF3	OSDxF2	OSDxF1	OSDxF0	OSDxCE7	OSDxE6	OSDxE5	OSDxE4	OSDxE3	OSDxE2	OSDxE1	OSDxE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–155. OSD Window Bitmap Colors E and F (WxBMPEF) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	OSDxF7–OSDxF0	Specifies color of bitmap value F by address of CLUT (0–255). (width = 1, 2, 4)
7–0	OSDxE7–OSDxE0	Specifies color of bitmap value E by address of CLUT (0–255). (width = 4)

To modify a color from the RAM CLUT, you need to do the following:

1. Wait for the BUSY bit of CLUT_RAM_CTRL to be zero.
2. Write into the CLUT_RAM_CR register. The first byte corresponds to the Cb information of the color the user wants to add to their palette; the second byte corresponds to the location of the color within the palette.
3. Wait again for the BUSY bit of CLUT_RAM_CTRL to return to zero.
4. Write into the CLUT_RAM_YCB register. The first byte corresponds to the Y information, the second to the Cb information of the color the user adds to their palette.

After a device reset, the color values of the RAM CLUT are undetermined.

Table 4–156. RAM Color Look-Up Table Control (CLUT_RAM_CTRL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BUSY
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 4–157. RAM Color Look-Up Table Control (CLUT_RAM_CTRL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	W20007–W20000	Reserved bits. Not used.
0	BUSY	Look-up table write busy signal When writing into CLUT_RAM_YCB or CLUT_RAM_CR, write is possible if 0. 0: Ready 1: Busy

Table 4–158. RAM Color Look-Up Table Y and Cb Data (CLUT_RAM_YCB) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	Y07	Y06	Y05	Y04	Y03	Y02	Y01	Y00	CB07	CB06	CB05	CB04	CB03	CB02	CB01	CB00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–159. RAM Color Look-Up Table Y and Cb Data (CLUT_RAM_YCB) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	Y07–Y00	Write data (Y) into built-in CLUT_RAM. Write into CLUT_RAM_CR first is required.
7–0	CB07–CB00	Write data (Cb) into built-in CLUT_RAM. Write into CLUT_RAM_CR first is required.

Table 4–160. RAM Color Look-Up Table Cr Data and Address (CLUT_RAM_CR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	CR07	CR06	CR05	CR04	CR03	CR02	CR01	CR00	AD07	AD06	AD05	AD04	AD03	AD02	AD01	AD00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–161. RAM Color Look-Up Table Cr Data and Address (CLUT_RAM_CR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	CR07–CR00	Write data (Cr) into built-in CLUT–RAM. Write into CLUT_RAM_YCB, CLUT_RAM_CR, in that order, is required.
7–0	AD07–AD00	CLUT write RAM address

4.5.7 Blending and Transparency

The hardware supports a transparency mode with bitmap windows. If transparency is enabled, then any pixel on the bitmap display that has a value of 0 allows video to be displayed. Essentially, 0 valued pixels are considered transparent and the background color shows through the bitmap.

Color blending at the pixel level is also supported. This feature is available for the bitmap windows only. If the window color blending is enabled, the amount of blending of each pixel is determined by the blending factor. As shown in Table 4–162, the window blending supports five different levels according to the selected blending factor.

Table 4–162. Blending and Transparency in OSD

TRANSPARENCY	BLENDING FACTOR	OSD WINDOW CONTRIBUTION	VIDEO CONTRIBUTION
OFF	0	0	1
	1	1/4	3/4
	2	1/2	1/2
	3	3/4	1/4
	4	1	0
ON		If pixel value = 0	
	0	0	1
	1	1/4	3/4
	2	1/2	1/2
	3	3/4	1/4
	4	1	0
		If pixel value = 0	
	X	1	0

When blending is on, if a pixel of bitmap window 1 overlaps the bitmap window 0, the corresponding pixel of bitmap window 0 is ignored.

4.5.8 Zoom

The video windows and OSD windows can be zoomed along their horizontal and vertical directions with a x2 or x4 expansion coefficient. When the video data is resized, a horizontal and vertical filter is applied to the Y information. There is no filter applied to the OSD windows.

The video window 0 or video window 1 can also be resized with a x6/5 vertical coefficient and a x9/8 horizontal coefficient. This feature is used when displaying a normal VGA image onto an NTSC/PAL screen.

Note that the x6/5 and x9/8 expansion filter is valid for only one video window at a time. One of the window images is confused if both video window 0 and video window 1 are displayed.

The expand filter can be bypassed by changing the bit EF of the ENCMODE1 register to 0.

4.5.9 OSD Configuration Registers

Five main registers in the OSD module control what windows are displayed if an expansion filter is used and which CLUT is used for each window. These registers are described below.

Table 4–163. Windows Configuration Control Registers

OFFSET	HEX ADDRESS	REGISTER NAME	
0x00	0x0003:0680	ENCMODE1	Encode mode 1 (for video window 0)
0x01	0x0003:0682	ENCMODE2	Encode mode 2 (for video window 0)
0x02	0x0003:0684	VIDEOWIN1_MODE	Video window 1 mode
0x03	0x0003:0686	OSDWIN0_MODE	OSD window 0 mode
0x04	0x0003:0688	OSDWIN1_MODE	OSD window 1 mode

Table 4–164. Encode Mode 1 (ENCMODE1) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PF	CF	CS	VMODE	CP	EF	VVRSZ	VHRSZ	CABG7	CABG6	CABG5	CABG4	CABG3	CABG2	CABG1	CABG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–165. Encode Mode 1 (ENCMODE1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15	PF	Picture format Sets the image output mode. Set the set values of the display base register and horizontal display position so that they are multiples of 2 when 4:2:2 or multiples of 4 when 4:1:1. 0: 4:2:2 1: 4:1:1
14	CF	Cb/Cr or Cr/Cb format Selects whether the Cb/Cr signal output is offset binary or a complement of 2. If set to 1, MSB bit is inverted with respect to input. 0: Offset binary 1: Complement of 2
13	CS	Cb/Cr or Cr/Cb format Selects whether output is in order of Cb/Cr or output is in order of Cr/Cb. 0: Cb/Cr 1: Cr/Cb
12	FSINV	Field signal inversion bit 0: Uninverted 1: Inverted
11	CP	CbCr position Compensates for the CbCr position of the image output to the video encoder. 0: Cb/Cr are output with no delay 1: Cb/Cr are output with 1 cycle delay

Table 4–165. Encode Mode 1 (ENCMODE1) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
10	EF	Expand filter on/off Specifies on/off of expand filter. Valid when either VVRSZ or VHRSZ is on or video window smooth is set. Furthermore, caution is required when using the filter since expanded filter memory only corresponds to 720 horizontal pixels. 0: Off 1: On
9	VVRSZ	Video window vertical expansion Set when the video window image is a multiple of 6/5 in vertical direction. 0: x 1 1: x 6/5
8	VHRSZ	Video window horizontal expansion Set when video window image is a multiple of 9/8 in horizontal direction. 0: x 1 1: x 9/8
7–0	CABG7–CABG0	Color address of background Specifies image display background color by CLUT address. In parts that do not display image, the color specified by this register is displayed.

Table 4–166. Encode Mode 2 (ENCMODE2) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	BCLUT	ULSC	RSV	TEST	VHZ01	VHZ00	VVZ01	VV00	VFF0	ACT
R/W	—	—	—	—	—	—	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–167. Encode Mode 2 (ENCMODE2) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9	BCLUT	Background CLUT selection Selects look-up table for background color display. 0: ROM 1: RAM
8	ULSC	Cb/Cr upper/lower selection Specifies bit position that is loaded in 4:1:1 format output. 0: Upper/lower (first/last) 1: Lower/upper (first/last)
7	VIFDT	Video interface data through bit This bit swaps the Y and C outputs of the video window 0: 0: Normal: Upper byte = Y; lower byte = C 1: Through: Upper byte = C; lower byte = Y
6	TEST	Reserved. 0 must be written to this bit
5–4	VHZ01–VHZ00	Video window horizontal direction zoom VHZ01 VHZ00 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as 00)
3–2	VVZ01–VVZ00	Video window vertical direction zoom VVZ01 VVZ00 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as 00)
1	VFF1	Video window display mode 0: Field mode 1: Frame mode
0	ACT	Sets image display on/off 0: Off 1: On

Table 4–168. Video Window 1 Mode (VIDEOWIN1MODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VHZ11	VHZ10	VVZ11	VVZ10	VFF1	ACT1
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 4–169. Video Window 1 Mode (VIDEOWIN1MODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved. Not used.
5–4	VHZ11–VHZ10	Video window horizontal direction zoom VHZ11 VHZ10 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
3–2	VVZ11–VVZ10	Video window vertical direction zoom VVZ11 VVZ10 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
1	VFF1	Video window display mode 0: Field mode 1: Frame mode
0	ACT1	Specifies image display on/off 0: Off 1: On

Table 4–170. OSD Window 0 Mode (OSDWIN0_MODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OVHZ01	OVHZ00	OVVZ01	OVVZ00	CLUTS	T0E	BW01	BW00	BF02	BF01	BF00	OACT0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–171. OSD Window 0 Mode (OSDWIN0_MODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–10	OVHZ01–OVHZ00	Window vertical direction zoom OVHZ01 OVHZ00 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
9–8	OVVZ01–OVVZ00	Window horizontal direction zoom OVVZ01 OVVZ00 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
7	CLUTS	CLUT select Selects look-up table that is used. 0: ROM look-up table 1: RAM look-up table
6	T0E	Transparency of OSD 0 enable bit 0: Off 1: On

Table 4–171. OSD Window 0 Mode (OSDWIN0_MODE) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
5–4	BW01–BW00	Bitmap width of OSD window 0 BW11 BW10 Bit width 0 0 1 0 1 2 1 0 4 1 1 8
3–1	BF02–BF00	Blending factor of OSD window 0 Sets blending ratio of OSD window 0 and video window 0, 1. BF12 BF11 BF10 OSD window 0 Video window 0 0 0 0 1 0 0 1 1/4 3/4 0 1 0 1/2 1/2 0 1 1 3/4 1/4 1 0 0 1 0 Other settings Reserved Reserved
0	OACT0	OSD window 0 activation Sets image display on/off 0: Off 1: On

Table 4–172. OSD Window 1 Mode (OSDWIN1_MODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	OVHZ11	OVHZ10	OVVZ11	OVVZ10	CLUTS	T1E	BW11	BW10	BF12	BF11	BF10	OACT1
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–173. OSD Window 1 Mode (OSDWIN1_MODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–10	OVHZ11–OVHZ10	Window horizontal direction zoom OVHZ11 OVHZ10 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
9–8	OVVZ11–OVVZ10	Window vertical direction zoom OVVZ11 OVVZ10 0 0 x1 0 1 x2 1 0 x4 1 1 Reserved (same as ‘00’)
7	CLUTS	CLUT select. Specifies look-up table that is used. 0: ROM look-up table 1: RAM look-up table
6	T1E	Transparency of OSD 1 enable bit 0: Off 1: On
5–4	BW11–BW10	Bitmap width of OSD window 1 BW11 BW10 Bit width 0 0 1 0 1 2 1 0 4 1 1 8

Table 4–173. OSD Window 1 Mode (OSDWIN1_MODE) Register Bit/Field Descriptions (Continued)

BIT	REGISTER NAME	DESCRIPTION
3–1	BF12–BF10	Blending factor of OSD window 0 Specifies blending ratio of OSD window 1 and video window 0, 1. BF12 BF11 BF10 OSD window 1 Video window 0 0 0 0 1 0 0 1 1/4 3/4 0 1 0 1/2 1/2 0 1 1 3/4 1/4 1 0 0 1 0 Other settings Reserved Reserved
0	OACT1	OSD window 1 activation Sets image display on/off 0: Off 1: On

4.5.10 Windows Positioning

All the OSD windows use a common reference pixel (base pixel). The position of this pixel is determined from the beginning of the HD and the beginning of the VD signal.

For each of the window (video or bitmap) and for the cursor, the location of the upper left corner can be specified. The size of the window needs to be specified as well.

The relationship of the display positions of each window is shown in Figure 4–34.

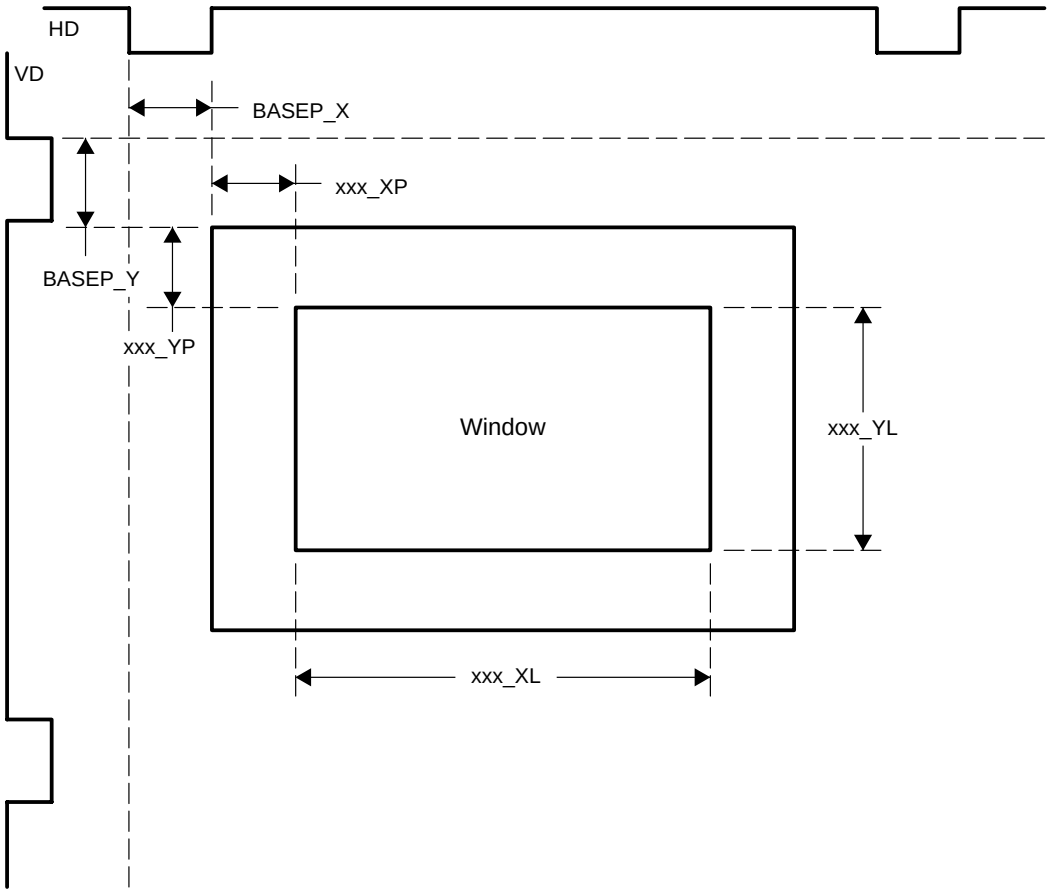


Figure 4–34. Window Positioning

Table 4–174. Window Position and Size Control Registers

OFFSET	Hex ADDRESS	REGISTER NAME	
0x12	0x0003:06A4	BASEP_X	Base pixel X coordinate
0x13	0x0003:06A6	BASEP_Y	Base pixel Y coordinate
0x14	0x0003:06A8	VIDEOWIN0_XP	Video window 0 horizontal position
0x15	0x0003:06AA	VIDEOWIN0_YP	Video window 0 vertical position
0x16	0x0003:06AC	VIDEOWIN0_XL	Video window 0 horizontal size
0x17	0x0003:06AE	VIDEOWIN0_YL	Video window 0 vertical size
0x18	0x0003:06B0	VIDEOWIN1_XP	Video window 1 horizontal position
0x19	0x0003:06B2	VIDEOWIN1_YP	Video window 1 vertical position
0x1A	0x0003:06B4	VIDEOWIN1_XL	Video window 1 horizontal size
0x1B	0x0003:06B6	VIDEOWIN1_YL	Video window 1 vertical size
0x1C	0x0003:06B8	OSDWIN0_XP	OSD window 0 horizontal position
0x1D	0x0003:06BA	OSDWIN0_YP	OSD window 0 vertical position
0x1E	0x0003:06BC	OSDWIN0_XL	OSD window 0 horizontal size
0x1F	0x0003:06BE	OSDWIN0_YL	OSD window 0 vertical size
0x20	0x0003:06C0	OSDWIN1_XP	OSD window 1 horizontal position
0x21	0x0003:06C2	OSDWIN1_YP	OSD window 1 vertical position
0x22	0x0003:06C4	OSDWIN1_XL	OSD window 1 horizontal size
0x23	0x0003:06C6	OSDWIN1_YL	OSD window 1 vertical size

Table 4–175. Base Pixel X Coordinate (BASEP_X) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	BPX09	BPX08	BPX07	BPX06	BPX05	BPX04	BPX03	BPX02	BPX01	BPX00
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–176. Base Pixel X Coordinate (BASEP_X) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	BPX09–BPX00	Sets horizontal direction display reference position of video windows 0, 1, OSD windows 0, 1, and cursor. Minimum set value is 24 pixels. If less than that, it becomes 24 pixels.

Table 4–177. Base Pixel Y Coordinate (BASEP_Y) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BPY08	BPY07	BPY06	BPY05	BPY04	BPY03	BPY02	BPY01	BPY00
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 4–178. Base Pixel Y Coordinate (BASEP_Y) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	BPY09–BPY00	Sets vertical direction display reference position of video windows 0, 1, OSD windows 0, 1, and cursor. Specified by number of lines from the falling edge of VD. Minimum value is 1 line. If 0, it becomes 1 line.

Table 4–179. Video Window Horizontal Position (VIDEOWINx_XP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VxX08	VxX07	VxX06	VxX05	VxX04	VxX03	VxX02	VxX01	VxX00
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 4–180. Video Window Horizontal Position (VIDEOWINx_XP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	VxX09–VxX00	Sets display start position in horizontal position of video window. Specified by number of pixels from the display reference position.

Table 4–181. Video Window Vertical Position (VIDEOWINx_YP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VxY08	VxY07	VxY06	VxY05	VxY04	VxY03	VxY02	VxY01	VxY00
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 4–182. Video Window Vertical Position (VIDEOWINx_YP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	VxY08–VxY00	Sets display start position in vertical position of video window. Sets value equivalent to number of lines of field. Specified by number of lines from the display reference position.

Table 4–183. Video Window Horizontal Size (VIDEOWINx_XL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	VxW11	VxW10	VxW9	VxW8	VxW7	VxW6	VxW5	VxW4	VxW3	VxW2	VxW1	VxW0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–184. Video Window Horizontal Size (VIDEOWINx_XL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	VxW11–VxW0	Length in horizontal direction of the video window is specified in number of pixels.

Table 4–185. Video Window Vertical Size (VIDEOWIN0_YL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	VxH10	VxH9	VxH8	VxH7	VxH6	VxH5	VxH4	VxH3	VxH2	VxH1	VxH0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 4–186. Video Window Vertical Size (VIDEOWIN0_YL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	VxH10–VxH0	Length in vertical direction of the video window is specified in number of lines. Sets the value equivalent to the number of lines of the field. For example, when 240 lines are set in NTSC, it becomes the valid vertical display range.

Table 4–187. OSD Window Horizontal Position (OSDWINx_XP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	WxX09	WxX08	WxX07	WxX06	WxX05	WxX04	WxX03	WxX02	WxX01	WxX00
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–188. OSD Window Horizontal Position (OSDWINx_XP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	WxX09–WxX00	Sets the horizontal direction display start position of the OSD window. Specified by the number of pixels from the display reference position.

Table 4–189. OSD Window Vertical Position (OSDWINx_YP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	WxY08	WxY07	WxY06	WxY05	WxY04	WxY03	WxY02	WxY01	WxY00
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 4–190. OSD Window Vertical Position (OSDWINx_YP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	WxY08–WxY00	Sets the vertical direction display start position of the OSD window. Sets the value equivalent to the number of lines of the field. Specified by the number of lines from the display reference position.

Table 4–191. OSD Window Horizontal Size (OSDWINx_XL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	WxW11	WxW10	WxW9	WxW8	WxW7	WxW6	WxW5	WxW4	WxW3	WxW2	WxW1	WxW0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–192. OSD Window Horizontal Size (OSDWINx_XL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	WxW11–WxW0	Length in horizontal direction of the OSD window is specified in number of pixels.

Table 4–193. OSD Window Vertical Size (OSDWINx_YL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	WxH10	WxH9	WxH8	WxH7	WxH6	WxH5	WxH4	WxH3	WxH2	WxH1	WxH0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 4–194. OSD Window Vertical Size (OSDWINx_YL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	WxH10–WxH0	Length in vertical direction of the OSD window is specified in number of lines. Sets the value equivalent to the number of lines of the field. For example, when 240 lines are set in NTSC, it becomes the valid vertical display range.

4.5.11 Hardware Cursor

The cursor always appears on top of other OSD windows. The user specifies the size, color, horizontal, and vertical thickness of the cursor block.

Table 4–195. Cursor Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x05	0x0003:068A	CURSOR_MODE	Cursor mode
0x24	0x0003:06C8	CUR_XP	Cursor window horizontal position
0x25	0x0003:06CA	CUR_YP	Cursor window vertical position
0x26	0x0003:06CC	CUR_XL	Cursor window horizontal size
0x27	0x0003:06CE	CUR_YL	Cursor window vertical size

Table 4–196. Cursor Mode (CURSOR_MODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	CAC7	CAC6	CAC5	CAC4	CAC3	CAC2	CAC1	CAC0	CLUTS	CHW2	CHW1	CHW0	CVW2	CVW1	CVW0	CACT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–197. Cursor Mode (CURSOR_MODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	CAC7–CAC0	Specifies cursor color by look-up table address
7	CLUTS	CLUT select. Selects the look-up table that is used. 0: ROM look-up table 1: RAM look-up table
6–4	CHW2–CHW0	Specifies horizontal direction line width of the cursor 1, 4, 8 – 28 pixels
3–1	CVW2–CVW0	Specifies vertical direction line width of the cursor 1, 2, 4 – 14 lines
0	CACT	Specifies cursor mode on/off 0: Off 1: On

Table 4–198. Cursor Window Horizontal Position (CUR_XP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	CSX09	CSX08	CSX07	CSX06	CSX05	CSX04	CSX03	CSX02	CSX01	CSX00
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 4–199. Cursor Window Horizontal Position (CUR_XP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	CSX09–CSX00	Sets the display start position in horizontal direction of the cursor. Specified by the number of pixels from the display reference position.

Table 4–200. Cursor Window Vertical Position (CUR_YP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	CSY08	CSY07	CSY06	CSY05	CSY04	CSY03	CSY02	CSY01	CSY00
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 4–201. Cursor Window Vertical Position (CUR_YP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8–0	CSY08–CSY00	Sets the display start position in vertical direction of the cursor. Specified by the number of lines from the display reference position.

Table 4–202. Cursor Window Horizontal Size (CUR_XL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	CSW11	CSW10	CSW9	CSW8	CSW7	CSW6	CSW5	CSW4	CSW3	CSW2	CSW1	CSW0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 4–203. Cursor Window Horizontal Size (CUR_XL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	CSW11–CSW0	Length in horizontal direction of the cursor is specified by the number of pixels.

Table 4–204. Cursor Window Vertical Size (CUR_YL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	CSH10	CSH9	CSH8	CSH7	CSH6	CSH5	CSH4	CSH3	CSH2	CSH1	CSH0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 4–205. Cursor Window Vertical Size (CUR_YL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	CSH10–CSH0	Length in vertical direction of the cursor is specified by the number of lines. Sets the value equivalent to the number of lines of the field. For example, in NTSC, when 240 lines are set, that becomes the valid vertical display range.

5 External Memory and External CPU Interface

This chapter describes the different blocks related to the external bus of the DSC24 chip. The external memory interface (EMIF) is made up of the SDRAM controller, the asynchronous memory interface (AMIF), and a DMA controller. This part of the chip, along with the external CPU interface, is discussed in this chapter.

5.1 Introduction

When processing data, it is necessary to have access to large memory spaces. The internal memory of the DSC24 chip is often too small to hold all the data and programs. The memory controller offers the possibility to access a large amount of external memory. This memory space can use different types of memories.

The external memory controller (EMIF) can be separated into three different subblocks the asynchronous external memory interface (AMIF), the SDRAM controller (SDRAMC), and the DMA controller (DMAC). All are described in this chapter.

Sometimes it is also necessary to replace the integrated ARM7 with an external CPU. This functionality of the DSC24 is also described in this chapter.

See the *DMA Controller* section for the different DMA channels available with the EMIF and ARM bus controllers. These DMA channels allow fast and efficient memory transfers between the different sections of the external memory and the ARM.

5.1.1 Memory Map

The whole memory map of the DSC24 MCU subsystem is represented in Table 5–1. This section describes the 128M bytes of dynamically allocated memory that starts at address 0010:0000h. The different blocks accessible by a host processor are:

- The internal peripheral area. This block is described in the *MCU Subsystem* section.
- The DSP memory area seen from the MCU side. The characteristics of this block of memory are detailed in the *ARM-DSP Communication* chapter.
- The SDRAM block. This block is described within the SDRAM interface section that is discussed later on in this chapter.

Table 5–1. ARM Memory Address Map

REGION NAME	START ADDRESS	END ADDRESS	SIZE (BYTES)
Reset vector ROM	0x0000:0000	0x0000:0003	4
ARM internal memory (AIM)	0x0000:0004	0x0000:7FFF	32K
Internal peripheral	0x0003:0000	0x0003:0FFF	4K
DSP memory (DARAM)	0x0004:0000	0x0004:FFFF	64K
Dynamically allocated external memory	0x0010:0000	0x080F:FFFF	128M

For details about the general memory map, see the *Bus Controller* section.

After reset is canceled, the MCU jumps to the beginning of the CS0 region (0x0010:0000h; start of the external memory).

5.1.2 Block Diagram

The EMIF controller is the main point of communication between the different elements of the DSC24. The image buffer from the DSP, the on-screen display module, the burst codec from the imaging peripheral, and the DMA are connected directly to the SDRAM controller. The video interface is connected either to the SDRAM controller or to the AMIF. The MODESET register from the video interface module controls the choice. Depending on the memory location accessed, the MCU accesses the SDRAM controller or the AMIF. Finally, the DMA busses coming from the serial ports interfaces and the AMIF DMA bus are multiplexed. The SDRAM can only access one at a time. The SDMODE register makes the selection.

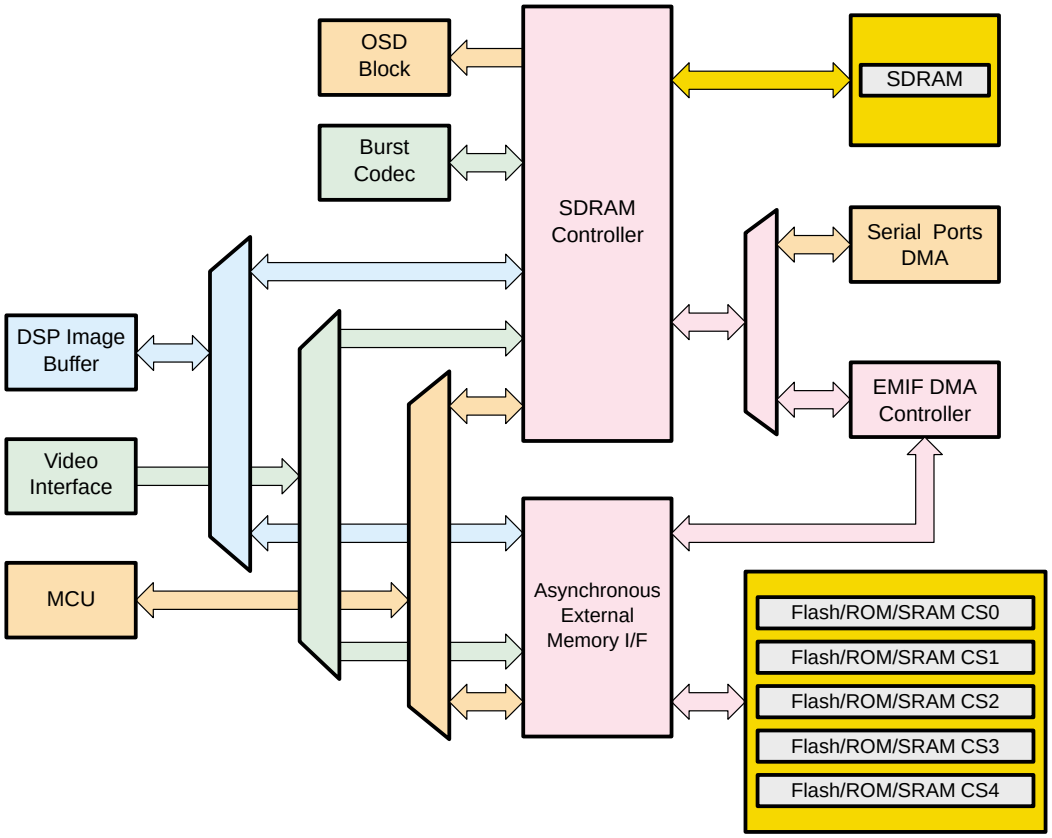


Figure 5–1. EMIF Block Diagram

5.1.3 Register Settings

As for any other peripheral, the EMIF module is controlled by the memory mapped registers setting. The address of each one of these registers is given in the tables below. There is a detailed description of each of these registers in this chapter.

The registers to configure each of the regions of the dynamically allocated external memory are located in the bus controller peripheral register area.

Table 5–2. EMIF Region Configuration Registers

OFFSET	ADDRESS	REGISTER NAME	
0x010	0x0003:0920	SDRST	SDRAM region start
0x011	0x0003:0922	EMST0	EM0 region start
0x012	0x0003:0924	EMST1	EM1 region start
0x013	0x0003:0926	EMST2	EM2 region start
0x014	0x0003:0928	EMST3	EM3 region start
0x015	0x0003:092A	EMST4	EM4 region start
0x020	0x0003:0940	SDRSZ	SDRAM region size
0x021	0x0003:0942	EMSZ0	EM0 region size
0x022	0x0003:0944	EMSZ1	EM1 region size
0x023	0x0003:0946	EMSZ2	EM2 region size
0x024	0x0003:0948	EMSZ3	EM3 region size
0x025	0x0003:094A	EMSZ4	EM4 region size

Table 5–3. AMIF Timing Control Registers Addresses

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0A00	CS0CTL1	CS0 control register 1
0x01	0x0003:0A02	CS0CTL2	CS0 control register 2
0x02	0x0003:0A04	CS1CTL1	CS1 control register 1
0x03	0x0003:0A06	CS1CTL2	CS1 control register 2
0x04	0x0003:0A08	CS2CTL1	CS2 control register 1
0x05	0x0003:0A0A	CS2CTL2	CS2 control register 2
0x06	0x0003:0A0C	CS3CTL1	CS3 control register 1
0x07	0x0003:0A0E	CS3CTL2	CS3 control register 2
0x08	0x0003:0A10	CS4CTL1	CS4 control register 1
0x09	0x0003:0A12	CS4CTL2	CS4 control register 2

Table 5–4. AMIF Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x0A	0x0003:0A14	CCDCDSPDEST	CCD and DSP transfer destination
0x0B	0x0003:0A16	PRIORCTL	Priority control

Table 5–5. EMIF DMA Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x0C	0x0003:0A18	SOURCEA_HI	Source address high
0x0D	0x0003:0A1A	SOURCEA_LO	Source address low
0x0E	0x0003:0A1C	DESTA_HI	Destination address high
0x0F	0x0003:0A1E	DESTA_LO	Destination address low
0x010	0x0003:0A20	DMASIZE	DMA transfer size
0x011	0x0003:0A22	DMADEVSEL	DMA transfer device selection
0x012	0x0003:0A24	DMACTL	DMA control

Table 5–6. External Host, Bus Open Control Register

OFFSET	ADDRESS	REGISTER NAME	
0x013	0x0003:0A26	BUSRLS	BUS release control

Table 5–7. SDRAM Buffer Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	0x0003:0980	SDBUF_D0L	SDRAM buffer data register low
0x01	0x0003:0982	SDBUF_D0H	SDRAM buffer data register high
0x02	0x0003:0984	SDBUF_D1L	SDRAM buffer data register low
0x03	0x0003:0986	SDBUF_D1H	SDRAM buffer data register high
0x04	0x0003:0988	SDBUF_D2L	SDRAM buffer data register low
0x05	0x0003:098A	SDBUF_D2H	SDRAM buffer data register high
0x06	0x0003:098C	SDBUF_D3L	SDRAM buffer data register low
0x07	0x0003:098E	SDBUF_D3H	SDRAM buffer data register high
0x08	0x0003:0990	SDBUF_D4L	SDRAM buffer data register low
0x09	0x0003:0992	SDBUF_D4H	SDRAM buffer data register high
0x0A	0x0003:0994	SDBUF_D5L	SDRAM buffer data register low
0x0B	0x0003:0996	SDBUF_D5H	SDRAM buffer data register high
0x0C	0x0003:0998	SDBUF_D6L	SDRAM buffer data register low
0x0D	0x0003:099A	SDBUF_D6H	SDRAM buffer data register high
0x0E	0x0003:099C	SDBUF_D7L	SDRAM buffer data register low
0x0F	0x0003:099E	SDBUF_D7H	SDRAM buffer data register high
0x010	0x0003:09A0	SDBUFA_HI	SDRAM buffer address high
0x011	0x0003:09A2	SDBUFA_LO	SDRAM buffer address low
0x012	0x0003:09A4	SDBUF_CTL	SDRAM buffer transfer control

Table 5–8. SDRAM Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x013	0x0003:09A6	SDMODE	SDRAM mode
0x014	0x0003:09A8	REFCTL	SDRAM refresh control
0x015	0x0003:09AA	SDPRTY1	SDRAM priority 1
0x016	0x0003:09AC	SDPRTY2	SDRAM priority 2
0x017	0x0003:09AE	SDPRTY3	SDRAM priority 3
0x018	0x0003:09B0	SDPRTY4	SDRAM priority 4
0x019	0x0003:09B2	SDPRTY5	SDRAM priority 5
0x01A	0x0003:09B4	SDPRTY6	SDRAM priority 6
0x01B	0x0003:09B6	SDPRTY7	SDRAM priority 7
0x01C	0x0003:09B8	SDPRTY8	SDRAM priority 8
0x01D	0x0003:09BA	PRTYON	SDRAM priority ON

5.2 Access to External Memory Regions

5.2.1 EMIF Regions

The EMIF can access 128M bytes of memory separated into the following six regions.

- CS0 or external flash memory area
- SDRAM area
- CS1 to CS4 external memory areas

The start address, size, and width of this memory are configurable. The exception is for the start of the external flash memory area (CS0).

The start address of the external memory region 0 is fixed at the lead address of the external memory region (0x10:0000). Since it is coded so that the DSC24 reset vector jumps to this address, the MCU accesses address 0 of external memory region 0 after reset is canceled. Normally, program memory such as FLASH ROM and so forth is connected to external memory region 0 (EM_CS0).

A SDRAM memory area is set up within the external memory range in the same way as the other external memory blocks 0 to 4. Specifying the offset values from the lead of the external memory area and their sizes can set up the areas.

5.2.2 Size and Start of the External Memory Regions

There is 128M bytes of external memory that can be accessed. This range can be divided into a maximum of six regions (SDRAM, EM_CS0 to EM_CS4). External memories of different capacities can be used efficiently by changing the memory map dynamically.

Each memory region is set by specifying the offset value from the lead of the external memory region in six start address registers (SDRST, EMST0–4). The minimum unit of increase/decrease that can be set is 1M byte. The values of each start address register after reset is shown below.

Table 5–9. Initial Values of Offset

VARIABLE REGION NAME	INITIAL VALUE	CORRESPONDING ADDRESS
SDRAM region	08h	0090:0000h
External memory region 0	00h	0010:0000h
External memory region 1	20h	0210:0000h
External memory region 2	38h	0390:0000h
External memory region 3	50h	0510:0000h
External memory region 4	68h	0690:0000h

Also, the size of each region can be specified in 256K byte units. The offset value is set in the dynamic partitioning size registers (SDRSZ, EMSZ[4:0]), with 1 being 256K bytes. The capacity of each region after reset is shown below.

Table 5–10. Initial Values of Capacity of Each Region

VARIABLE REGION NAME	INITIAL VALUE	CORRESPONDING SIZES
SDRAM region	60h	24M bytes
External memory region 0	20h	8M bytes
External memory region 1	60h	24M bytes
External memory region 2	60h	24M bytes
External memory region 3	60h	24M bytes
External memory region 4	60h	24M bytes

The start address and size configuration registers are described in the tables below.

Table 5–11. SDRAM Region Start (SDRST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	SDST6	SDST5	SDST4	SDST3	SDST2	SDST1	SDST0
R/W	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	0	0	0	1	0	0	0

Table 5–12. SDRAM Region Start (SDRST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–7	RSV	Reserved bits. Not used.
6–0	SDST6–SDST0	External SDRAM region set: The SDRAM region start address is set by specifying the offset value from the lead address of external memory region (0010:0000h). The setting is in 1M-byte units.

Table 5–13. EMx Memory Region Start (EMSTx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	EST06	EST05	EST04	EST03	EST02	EST01	EST00
R/W	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	See Table 5–9						

Table 5–14. EMx Memory Region Start (EMSTx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–7	RSV	Reserved bits. Not used.
6–0	ESTx6–ESTx0	External memory region start The EMx region start address is set by specifying the offset value from the lead address of external memory region (0010:0000h). The setting is in 0.1M-byte units.

Table 5–15. SDRAM Memory Size (SDRSZ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	SDSZ9	SDSZ8	SDSZ7	SDSZ6	SDSZ5	SDSZ4	SDSZ3	SDSZ2	SDSZ1	SDSZ0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	1	1	0	0	0	0	0

Table 5–16. SDRAM Memory Size (SDRSZ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	SDSZ9–SDSZ0	External SDRAM region size set Sets size of external SDRAM region Setting is in 256K-byte units Maximum size of SDRAM region is 64M bytes

Table 5–17. Memory Region Size (EMSZx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	ESZ09	ESZ08	ESZ07	ESZ06	ESZ05	ESZ04	ESZ03	ESZ02	ESZ01	ESZ00
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	See Table 5–10									

Table 5–18. Memory Region Size (EMSZx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	ESZx9–ESZx0	External memory region size set Sets size of external EMx region Setting is in 256K-byte units

5.2.3 Hardware Interface

As external memory interfaces, the DSC24 has an SDRAM interface and a general-purpose memory interface. The MCU accesses SDRAM via the SDRAM controller module (SDRAMC). It can also access general-purpose SRAM or flash ROM via the asynchronous external memory interface module (AMIF). In access from MCU to external memory, a wait is automatically inserted and its cycle varies depending on SDRAM frequency, external memory access time, and so forth. As access modes to external memory, byte (8 bits), half word (16 bits), and word (32 bits) access are possible.

The functions of the terminals of the EMIF busses are shown in Table 5–19.

Table 5–19. External Memory Busses Pin Table

	PIN NAME	FUNCTION	EXPLANATION
	BUS_REQ/ EM_WAIT	I	External bus request
	BUS_ACK/ EM_BS	O	External bus acknowledge
Asynchronous External Memory Interface	EM_CS0	I/O	EMIF chip select #0
	EM_CS1	O	EMIF chip select #1
	EM_CS2	O	EMIF chip select #2
	EM_CS3	O	EMIF chip select #3
	EM_CS4	O	EMIF chip select #4
	EM_WE	I/O	Write enable signal
	EM_RNW	O	READ/WRITE strobe signal
	EM_BEL	O	Lower byte enable signal Lower word enable when 32-bit width
	EM_BEH	O	Upper byte enable signal Upper word enable when 32-bit width
	EM_OE	I/O	Read enable signal
	EM_WIDTH	I	External bus width selection for CS0
	ARM_A[22–0]	I/O	AMIF ADDRESS bus Outputs byte address to 8-bit devices Half word address to 16-bit devices Word address to 32-bit devices
	ARM_D[15–0]	I/O	AMIF data bus (lower 16 bits)
	ARM_D[31–16] SDR_DQ[31–16]	I/O	AMIF DATA bus (upper 16 bits) SDRAM DATA bus (upper 16 bits) If the width of one of the CS regions is set to 32 bits, upper 16 bits of SDRAM data bus are used. In this case, SDRAM has 16-bit width.

Table 5–19. External Memory Busses Pin Table (Continued)

	PIN NAME	FUNCTION	EXPLANATION
SDRAM Controller	SDR_DQ[15–0]	I/O	SDRAM data bus
	SDR_A[14–0]	O	SDRAM address bus
	SDR_RAS	O	SDRAM raw address strobe
	SDR_CAS	O	SDRAM column address strobe
	SDR_WE	O	SDRAM write enable
	SDR_CS0	O	SDRAM support chip select
	SDR_CKE	O	SDRAM clock enable
	SDR_DQMHH	O	SDRAM mask for DQ[31:24]
	SDR_DQMHL	O	SDRAM mask for DQ[23:16]
	SDR_DQMLH	O	SDRAM mask for DQ[15:8]
	SDR_DQMLL	O	SDRAM mask for DQ[7:0]
	SDR_CLK	O	SDRAM clock

If 32-bit width is selected for any of the CS regions, the upper 16 bits (SDR_DQ[31:16]) of the SDRAM IF data bus are used. In this case, SDRAM has a 16-bit width (SDR_DQ[15:0]).

A chip select signal (EM_CS0–EM_CS4, SDR_CS) is generated for each partitioned region. Correspondence between each memory region and CS is shown below.

Table 5–20. Initial Values of Chip Select Signals

VARIABLE AREA NAME	CHIP SELECT
SDRAM area	SDR_CS
External memory area 0	EM_CS0
External memory area 1	EM_CS1
External memory area 2	EM_CS2
External memory area 3	EM_CS3
External memory area 4	EM_CS4

If the regions are set so that they overlap, the chip select is valid for the region with the higher priority ranking. If possible, try not to overlap any memory areas.

Table 5–21. Chip Select Priority Order

VARIABLE REGION NAME	CHIP SELECT	PRIORITY
SDRAM region	SDRAM	Low
External memory region 0	EM_CS0	
External memory region 1	EM_CS1	
External memory region 2	EM_CS2	
External memory region 3	EM_CS3	
External memory region 4	EM_CS4	High

5.3 Asynchronous General-Purpose Memory Interface

In the external memory region 0x0010:0000–0x080F:FFFF, general-purpose memory such as SRAM, EPROM, and FLASH memory can be used. For the asynchronous general-purpose memory interface, SDRAM, smart media (SSFDC), compact flash (CF) memory, and so forth cannot be connected without adding external logic circuitry.

There are five control output terminals from CS0 through CS4 that can be directly connected with these general-purpose memories without an external decoding circuit. Their timing can be controlled by software. For example, if a low-speed device in which output data is not immediately set to high impedance is used, a bus IDLE state cycle can be inserted between that write cycle and the next memory access cycle. Thus, data collision on the data bus can be avoided.

5.3.1 Width of External Memory Regions

In modes 0, 1, and 3, the width of the CS0 region can be 8 or 16 bits. The EM_WIDTH pin selects this. In mode 2, the bus width of this region is either 16 or 32 bits.

Table 5–22. CS0 Bus Width

EM_WIDTH STATE	CS0 BUS WIDTH	
	MODES 0, 1, 3	MODE 2
Low	8 bits	16 bits
High	16 bits	32 bits

For the memory areas defined as CS1 to CS4, the data width can either be 8, 16, or 32 bits. The selection is made by register settings (BUSWx1:0). After reset, a width of 8 bits is selected.

If 32-bit width is selected for any of the CS regions, the upper 16 bits (SDR_DQ[31:16]) of the SDRAM IF data bus are used. In this case, SDRAM has a 16-bit width (SDR_DQ[15:0]).

Write is possible in byte units for any of sections CS0 to CS4.

The registers that control the bus width of the external memory regions are described below.

5.3.2 Timing Control

5.3.2.1 Software Control

The timing for each CS can be controlled by software in response to the access timing of the device connected to each CS. Also, an idle state can be inserted in order to prevent bus collision when continuous memory access occurs. Table 5–23 shows the parameters that can be modified.

Table 5–23. Timing Setting Parameters

PARAMETER	EXPLANATION	CLOCK VARIATION	VARIABLE UNIT	VALUE AT RESET FOR CS0 IN MODE 2	VALUE AT RESET IN OTHER CASES
CYCLE	Number of cycles	2 to 16 clock cycles	1 CLK unit	3 CLK	10 CLK
CS_SU	Number of CS setups	0 to 1.5 clock cycles	1/2 CLK unit	0 CLK	0 CLK
WE_SU	Number of WE setups	0 to 1.5 clock cycles	1/2 CLK unit	0.5 CLK	0.5 CLK
OE_SU	Number of OE setups	0 to 1.5 clock cycles	1/2 CLK unit	0.5 CLK	0.5 CLK
CS_WAIT	Number of CS waits	1 to16 clock cycles	1 CLK unit	3 CLK	10 CLK
WE_WAIT	Number of WE waits	1 to16 clock cycles	1 CLK unit	2 CLK	9 CLK
OE_WAIT	Number of OE waits	1 to16 clock cycles	1 CLK unit	2 CLK	9 CLK
IDLE	Number of idles	0 to 3 clock cycles	1 CLK unit	1 CLK	1 CLK

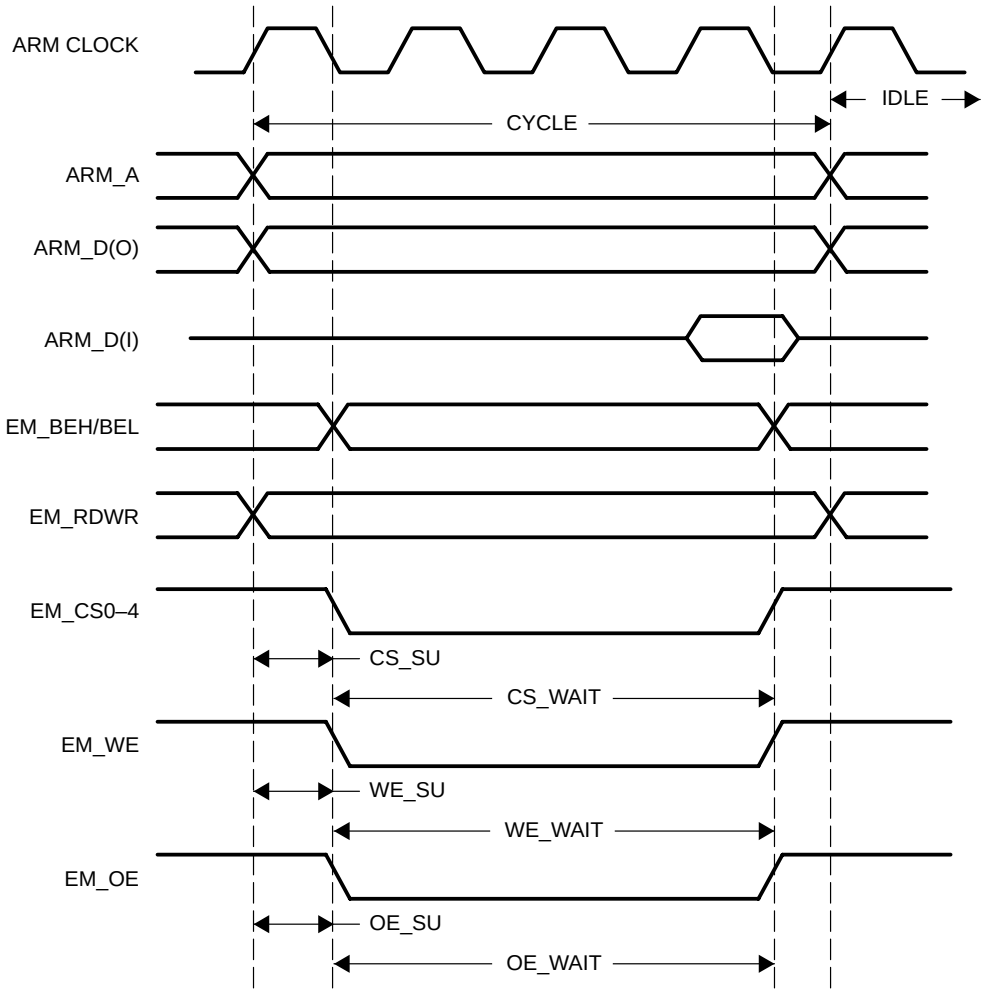


Figure 5–2. External Memory Bus Timing

The following registers control the variations of the timings.

Table 5–24. CSx Control Register 1 (CSxCTL1) Register†

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	OEC x3	OEC x2	OEC x1	OEC x0	WEC x3	WEC x2	WEC x1	WEC x0	CEC x3	CEC x2	CEC x1	CEC x0	CYCLE x3	CYCLE x2	CYCLE x1	CYCLE x0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	1	0	0	0	1	0	0	0	1	0	0	1	1	0	0	1

† In mode 2, the reset value of this register is 0x1122h.

Table 5–25. CSx Control Register 1 (CSxCTL1) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	OECx3–OECx0	Output enable width. The width of the OE_WAIT is the set value + 1.
11–8	WECx3–WECx0	Write enable width. The width of the WE_WAIT is the set value + 1.
7–4	CECx3–WECx0	Chip enable width. The width of the CS_WAIT signal is the set value + 1.
3–0	CYCLEx3–CYCLEx0	Cycle width. The width of the cycle is the set value + 1. However, the value 0000 corresponds to two clocks.

Table 5–26. CSx Control Register 2 (CSxCTL2) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	BUSWx1	BUSWx0	IDLEx1	IDLEx0	OESUx1	OESUx0	WESUx1	WESUx0	CESUx1	CESUx0
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	1	0	1	0	1	0	0

Table 5–27. CSx Control Register 2 (CSxCTL2) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–8	BUSWx1–BUSWx0	External device bus width (only for CS1 to CS4) 0 0 8 bits 0 1 16 bits 1 0 32 bits 1 1 Reserved For CS0, the bus width is selected by hardware; these two bits are reserved.
7–6	IDLEx1–IDLEx0	Number of clocks of idle state The number of clocks used is equal to the set value.
5–4	OESUx1–OESUx0	Number of clocks of output enable setup The number of clocks used is equal to half the set value.
3–2	WESUx1–WESUx0	Number of clocks of write enable setup The number of clocks used is equal to half the set value.
1–0	CESUx1–CESUx0	Number of clocks of chip enable setup The number of clocks used is equal to half the set value.

5.3.2.2 Hardware Control

In device operating mode 2, the EM_WAIT signal is available for the CS0 space only to enable a slow device to be interfaced with the DSC24. This feature is not available in any other operating modes, neither for CS1 to CS4 in mode 2.

In mode 2, each time CS0 is accessed, EM_BS (bus cycle start) is driven low during one cycle. If EM_WAIT is driven low during the N–1 bus cycle by an external device, an extra wait state (hardware wait state) is added to the programmed wait states (software wait states).

For instance, if CS_WAIT = 5 and EM_WAIT is held low for five clock cycles, the total number of wait states is 10.

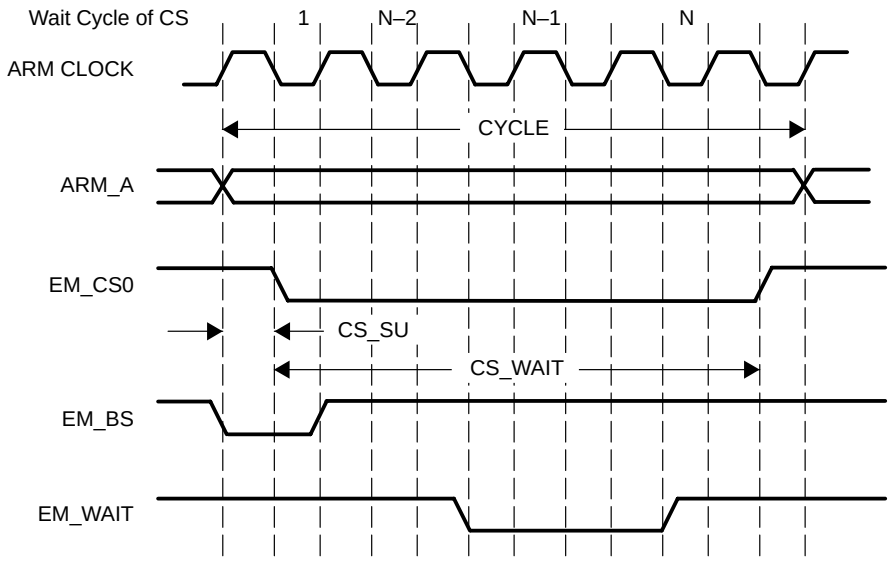


Figure 5–3. External Memory Bus Timing With Hardware Wait States

When the WAIT signal is used, the setup and wait time for the CS0, WE, and OE signals must be configured in order for the rising edge of these signals to appear less than 1.5 ARM clock cycles before the end of the address valid area.

CS_SU+CS_WAIT ≥ CYCLE – 1.5 × clocks
WE_SU+WE_WAIT ≥ CYCLE – 1.5 × clocks
OE_SU+OE_WAIT ≥ CYCLE – 1.5 × clocks

The EM_WAIT signal needs to be externally synchronized with the rising edge of CLK_ARM for the interface to work correctly. See the *TMS320DSC24 GHK* data manual, literature number SPRS195, for the setup and hold time of this signal.

5.3.2.3 AMIF Access Priority

The asynchronous external memory interface is connected to the MCU, the image buffer memory of the DSP, the DMA controller of the EMIF, and to the video interface. It is possible to change the access priority by register settings. The register described below concern only the transfer for the memory region CS0 to CS4; it does not concern the SDRAM area, even for the DSP image buffer.

Table 5–28. Priority Control (PRIORCTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	1PRY0	1PRY0	2PRY1	2PRY0	3PRY1	3PRY0	4PRY1	4PRY0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	1	1	0	1	1

Table 5–29. Priority Control (PRIORCTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
7–6	1PRY1–1PRY0	Specifies first priority 1PRY1 1PRY0 First priority 0 0 Video interface 0 1 DMA 1 0 MCU 1 1 DSP
5–4	2PRY1–2PRY0	Specifies second priority 2PRY1 2PRY0 Second priority 0 0 Video interface 0 1 DMA 1 0 MCU 1 1 DSP
3–2	3PRY1–3PRY0	Specifies third priority 3PRY1 3PRY0 Third priority 0 0 Video interface 0 1 DMA 1 0 MCU 1 1 DSP
1–0	4PRY1–4PRY0	Specifies fourth priority 4PRY1 4PRY0 Fourth priority 0 0 Video interface 0 1 DMA 1 0 MCU 1 1 DSP

5.3.2.4 Video Interface and DSP Access to AMIF

When bit 12 of the MODESET register in the video interface module is switched to 1, the data generated by the video interface is redirected to the AMIF instead of the SDRAM controller. In that case, the CCDCDSPDEST register lets the user specify in which CS region the data is to be written to.

As well, bit 8 of the image buffer DMA control register (DMA_CTRL) lets the user specify if they want the shared memory DMA module to be connected to the SDRAM controller or to the AMIF interface. When the AMIF interface is selected, bits 2:0 of the CCDCDSPDEST selects which one of the five regions is used.

Table 5–30. CCD and DSP Transfer Destination (CCDCDSPDEXT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	CDST2	CDST1	CDST0	RSV	DDST2	DDST1	DDST0
R/W	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	—	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	0	0	0	—	0	0	0

Table 5–31. CCD and DSP Transfer Destination (CCDCDSPDEXT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–7	RSV	Reserved bits. Not used.
6–4	CDST2–CDST0	CCD / Video input data destination CDST[2:0] Transfer destination 0 0 0 CS0 0 0 1 CS1 0 1 0 CS2 0 1 1 CS3 1 0 0 CS4
3	RSV	Reserved bit. Not used.
2–0	DDST2–DDST0	DSP data destination DDST[2:0] Transfer destination 0 0 0 CS0 0 0 1 CS1 0 1 0 CS2 0 1 1 CS3 1 0 0 CS4

5.4 SDRAM Controller

The SDRAM controller block acts as the primary interface between SDRAM and all functional blocks such as the processors (ARM, DSP), video interface block, OSD, and DMA controller. This block supports SDRAM timing of a maximum of 75 MHz (see the *TMS320DSC24 GHK* data manual, literature number SPRS195, for the exact values) and provides continuous data access with low overhead. The transfer of data between the peripheral modules and the SDRAM controller is done through some DMA channels. The SDRAM access priority can be modified.

5.4.1 Features of the SDRAM Controller

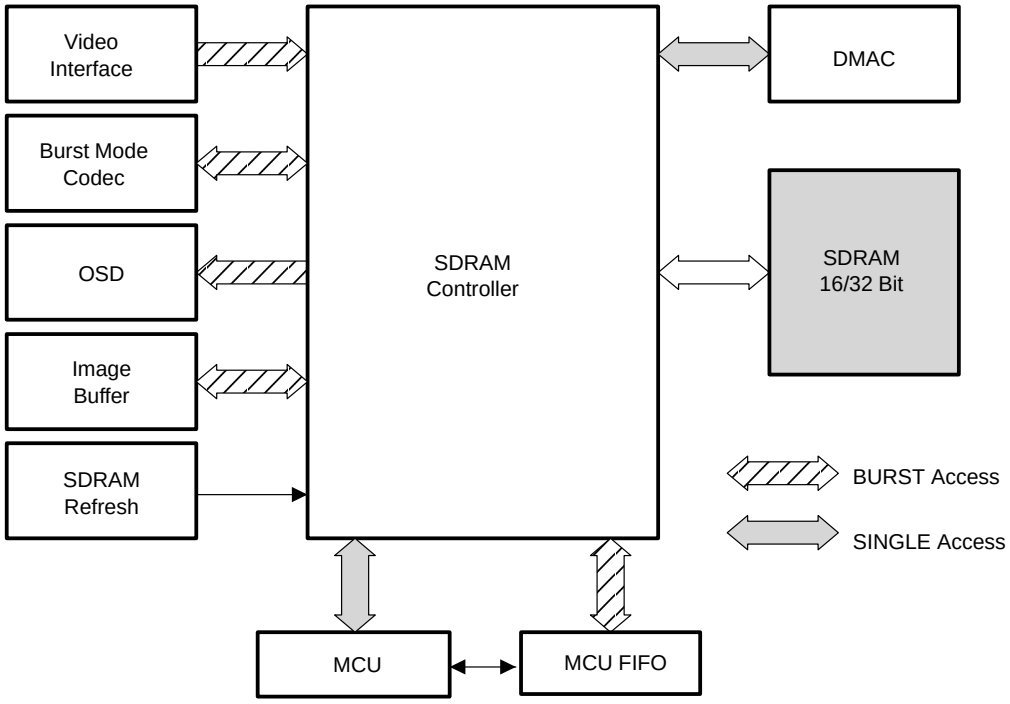


Figure 5–4. SDRAM Controller Access

The SDRAM region can be set within the external memory area similar to the other external memory regions 0 to 4. The region is set by specifying the offset value from the lead of the external memory region in the start address register (SDRST). After reset, it becomes 08h. The size of the SDRAM area can be specified in 256K-byte units similar to the external memory regions 0–4. The offset value is set in the dynamic partitioning size register (SDRSZ). The initial value is 24M bytes. See the *Size and Start of the External Memory Regions* chapter for more details on the register configuration.

The SDRAM controller can access 64M bytes of SDRAM and supports both 16- and 32-bit type of devices. Each access can be done in word, half-word, or byte units from the MCU side.

The controller also supports the refresh commands and can place the device in a self-refresh mode prior to going into a power-down mode. The refresh interval is programmable.

The latency for the column access strobe (CAS) is programmable from 1 to 3.

5.4.2 Memory Structure Setting

The DSC24 can connect to SDRAM with a maximum of 512M bits. The address that is connected differs depending on the memory structure of the SDRAM. See Table 5–32.

Table 5–32. Correspondence Between Memory Structure and Address

MEMORY STRUCTURE	ADDRESS	NUMBER OF BANKS	BANK
2K × 256 WORD	SDR_A10–SDR_A0	2	SDR_A13
		4	SDR_A14–SDR_A13
4K × 256 WORD	SDR_A11–SDR_A0	2	SDR_A13
		4	SDR_A14–SDR_A13
4K × 512 WORD	SDR_A11–SDR_A0	2	SDR_A13
		4	SDR_A14–SDR_A13
8K × 512 WORD	SDR_A12–SDR_A0	2	SDR_A13
		4	SDR_A14–SDR_A13

If one of the external memory regions CS0 to CS4 is configured as 32 bits, or if the device is in operating mode 2, only a 16-bit SDRAM can be used. In that case, only SDR_DQ15 to SDR_DQ0 are used for the data line.

5.4.3 SDRAM Configuration Registers

Before using the SDRAM, the SDRAM controller clock must be turned on by setting bit 4 of the MOD1 register from the clock controller module. When the module is on, the user configures it to match the SDRAM memory specifications. The SDMODE register lets the user change the number of cycles to access the memory as well as the latency for the CAS. The memory can be configured for using two or four banks. The SDMODE register is also used to write the commands that need to be sent to the SDRAM chip.

In addition to this register, the REFCTL allows the user to specify the auto-refresh frequency used by the controller.

Bit 14 of SDMODE selects the bus width for SDRAM access. If the device is used in operating mode 2, or if one of the CS regions is selected to have a bus width of 32 bits, this bit is ignored by the system and always considered as 1.

Table 5–33. SDRAM Mode (SDMODE) Register

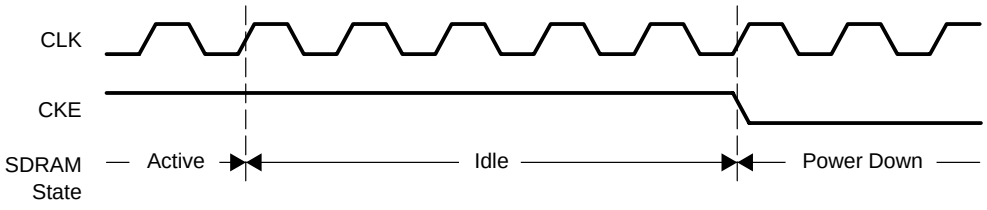
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	TRDL	SDBW	SDMAS	BNS	CASL1	CASL0	MEMT1	MEMT0	DQMC	APO	SDCTL5	SDCTL4	SDCTL3	SDCTL2	SDCTL1	SDCTL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0

Table 5–34. SDRAM Mode (SDMODE) Register Bit/Field Descriptions

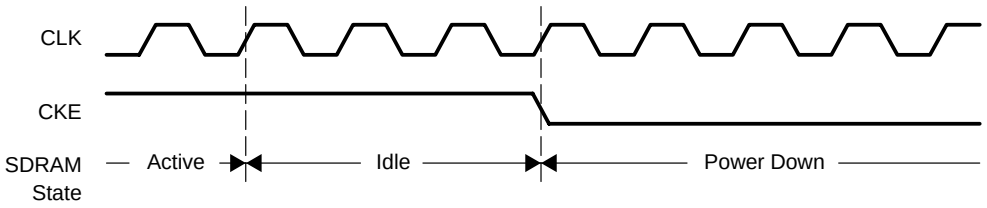
BIT	REGISTER NAME	DESCRIPTION
15	TRDL	t_{RDL} select One or two cycles can be selected in SDRAM clock units. 0: 1 cycle 1: 2 cycles
14	SDBW	SDRAM bus width select 0: 32 bits 1: 16 bits
13	SDMAS	DMA select 0: Between asynchronous external memory interface and SDRAM 1: Between serial IF and SDRAM
12	BNS	Bank select 0: 2 banks 1: 4 banks
11–10	CASL1–CASL0	CAS latency CAS latency can be selected in SDRAM clock units. CASL1 CASL0 0 0: Reserved 0 1: 1 cycle 1 0: 2 cycles 1 1: 3 cycles
9–8	MEMT1–MEMT0	Memory type MEMT1 MEMT0 0 0 2K x 256 words 0 1 4K x 256 words 1 0 4K x 512 words 1 1 8K x 512 words
7	DQMC	DQM control 0: Normal 1: Forcibly sets DQM signal to 1
6	APO	Automatic power-down mode 0 : Disabled 1 : Enabled
5–0	SDCTL5–SDCTL0	SDRAM control select bit SDCTL[5:0] 000000 NOP No operation 000001: MSR Sets mode 000010: PREA Precharges all banks 000100: REF Issues auto refresh command 001000: SELF Enters self refresh mode 010000: SELFC Cancels self refresh mode, power-down mode 100000: PDN Enters power-down mode

The SDRAM automatic power-down mode controlled by bit 6 of the SDMODE register can be used to reduce the SDRAM power consumption. When this bit is set to 0, the SDR_CKE signal of the SDRAM interface is fixed in a high level. When APO is set to 1, the status of the SDR_CKE signals is controlled by hardware. In this case, when the SDRAM state is idle and no more SDRAM access is required, the CKE signal is negated (low level). Also, when the next SDRAM access arrives, the signal is asserted (high level) one cycle before the active state. See Figure 5–5.

a. SDRAM going into power-down mode after a write:



b. SDRAM going into power-down mode after a read:



c. SDRAM waking-up from power-down mode:

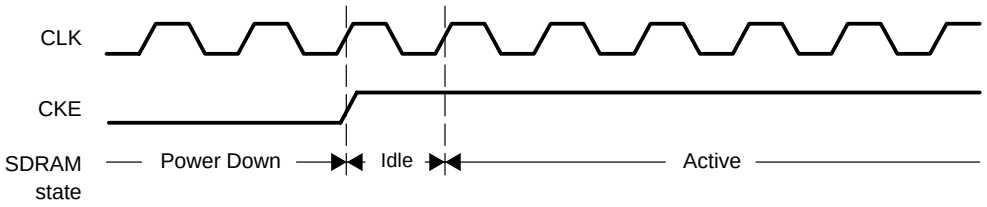


Figure 5–5. SDR_CKE Signal During Automatic Power-Down Mode

Table 5–35. SDRAM Refresh Control (REFCTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	REFEN	REF C7	REF C6	REF C5	REF C4	REF C3	REF C2	REF C1	REF C0
R/W	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0

Table 5–36. SDRAM Refresh Control (REFCTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	REFEN	Auto refresh enable 0: Disabled 1: Enabled
7–0	REFC7–REFC0	Refresh cycle Sets interval of auto refresh. Interval is (set value + 1) x 8 SDRAM clocks.

5.4.3.1 SDRAM Command

The MCU can issue the following commands to SDRAM by settings in the registers of the SDRAM controller (SDMODE bits 5 to 0).

- MRS (mode register set)
- PREA (precharge all)
- REF (auto refresh)
- SELF (self refresh entry)
- SELFC (self refresh end)
- PDN (power-down mode)

When a MRS command is issued, the following data settings are performed in the SDRAM mode register.

Table 5–37. MRS Value

SDRAM ADDRESS PIN	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Meaning	Reserved				0	0	Read Latency			S/I	Burst Length		
DSC24 Output Value	0	0	0	0	0	0	0	CL1	CL0	0	0	BST1	BST0

Table 5–38. MRS Value BIt Information

BIT	REGISTER NAME	DESCRIPTION
5–4	CL1, CL0	Selects CAS latency 1(01b), 2(10b), 3(11b)
1–0	BST1, BST0	Burst length The length of a burst is four when the SDRAM bus is configured as 32 bits; the length of a burst is eight if the bus is configured as 16 bits.

5.4.3.2 SDRAM Priority Access

The access priority to the SDRAM can be modified. The SDPRTY1 to SDPRTY8 registers described below control the access priority. By default, the video interface has the highest priority and the SDRAM refresh request has the lowest priority.

Table 5–39. Default SDRAM Priority Access

SDRAM CONTROLLER INPUT	PRIORITY
Video interface DMA request	Low
Burst mode codec DMA request	
OSD DMA request	
External memory I/F, serial I/F DMA request	
MCU core request	
MCU FIFO request	
Image buffer DMA request	
SDRAM auto refresh request	High

Table 5–40. SDRAM Priority (SDPRTYx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	PRTYx7	PRTYx6	PRTYx5	PRTYx4	PRTYx3	PRTYx2	PRTYx1	PRTYx0
R/W	—	—	—	—	—	—	—	—		R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	See Table 5–42							

Table 5–41. SDRAM Priority (SDPRTYx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7–0	PRTYx7–PRTYx0	SDRAM access priority The access priority is determined by writing a 1 in any bit of bit 0 through bit 7.

The binary value 1 should not be written twice or more into any one register. If this happens, bit 7 has the highest priority and bit 0 has the lowest priority (e.g. ignored). Also, if a 1 is written in the same bit position in different registers, the request of the low register address takes priority and the request of the high register address is ignored. The contents of the requests selected by each register are as shown in Table 5–42.

Table 5–42. Content of SDRAM Priority Configuration Registers

REGISTER NAME	SDRAM CONTROLLER INPUT	DEFAULT VALUE OF REGISTER
SDPRTY1:	Video interface DMA request	0x80
SDPRTY2:	Burst mode codec DMA request	0x40
SDPRTY3:	OSD DMA request	0x20
SDPRTY4:	EMIF DMA request, serial I/F DMA request	0x10
SDPRTY5:	MCU core request	0x08
SDPRTY6:	MCU FIFO request	0x04
SDPRTY7:	Image buffer DMA request	0x02
SDPRTY8:	SDRAM auto refresh request	0x01

When the priority configuration is ready, the user must write a 1 into bit 0 of the PRTYON to enable it. Since the SDRAM controller internal priority is changed automatically when access to SDRAM becomes IDLE (e.g. at the end of the current burst transfer); the user does not need to pay attention to the change period.

Table 5–43. SDRAM Priority ON (PRTYON) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	PTYON
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 5–44. SDRAM Priority ON (PRTYON) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	PTYON	Priority on By writing a 1 into this register, the priority written into registers SDPRTY1–8 becomes valid.

If a request with a higher priority than the current transfer one occurs, the burst transfer in progress is terminated prior to serving the received request. The size of a burst is four in a 32-bit mode and eight in a 16-bit mode (eight 16-bit words worth of data in both cases).

5.4.4 SDRAM Initialization

SDRAM has a power-on procedure. Since the procedure differs depending on the SDRAM that is used, the SDRAM must be initialized by software using the SDRAM command. Also, since the DQM (input/output mask) signal is high output as default, after the mode register set command (MRS) is finished, the DQM must be set to low level. Setting values in the SDRAM controller registers performs these SDRAM settings.

A typical sequence is given below:

1. Apply power and start the clock. Attempt to maintain the NOP command at the input.
2. Maintain stable power, stable clock, and the NOP condition for a minimum of 100 μs or 200 μs. (Minimum time depends on devices.)
3. Precharge all banks by the *Precharge All* (PREA) command.
4. Assert a minimum of two or eight auto-refresh commands. Assert times depend on the devices. See the *TMS320DSC24 GHK* data manual, literature number SPRS195.
5. Program the mode register by the mode register set command (MRS).

5.4.5 MCU Buffered Access

In the DSC24 there are two types of access to SDRAM: burst access via a buffer and ordinary memory access. Transfer to and from SDRAM is performed in 16-word units. There is a control register in the SDRAM controller for burst access as well as sixteen 16-bit data registers to store the data from a 16-word burst transfer.

To write into the SDRAM, data needs to be set in these data registers. After the write destination SDRAM address is set in the buffer address registers, a write command can be written in the buffer transfer control register. When the proper bit (WM or WA) in this register is written, writing to the SDRAM is performed.

To read from SDRAM, the read origin from the SDRAM address must be set first. This information is stored into the buffer address registers. When a read command is issued by writing the bit RSD of the control register, the data is read from SDRAM and stored in the data registers.

The user must check the status of the WM, WA, and RSD bit to know if a transfer is complete or not.

The buffer registers as well as the registers to control the buffer are described in the following tables.

Table 5–45. SDRAM Buffer Data Register Low (SDBUF_DxL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SDx15	SDx14	SDx13	SDx12	SDx11	SDx10	SDx09	SDx08	SDx07	SDx06	SDx05	SDx04	SDx03	SDx02	SDx01	SDx00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–46. SDRAM Buffer Data Register Low (SDBUF_DxL) Register Bit/FIeld Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	SDx15–SDx00	SDRAM data register low

Table 5–47. SDRAM Data Register High (SDBUF_DxH) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SDx31	SDx30	SDx29	SDx28	SDx27	SDx26	SDx25	SDx24	SDx23	SDx22	SDx21	SDx20	SDx19	SDx18	SDx17	SDx16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–48. SDRAM Data Register High (SDBUF_DxH) Register Bit/FIeld Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	SDx31–SDx16	SDRAM data register high

Table 5–49. SDRAM Buffer Address High (SDBUFA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	AAI	RSV	RSV	SDA21	SDA20	SDA19	SDA18	SDA17	SDA16
R/W	—	—	—	—	—	—	—	R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	—	—	0	0	0	0	0	0

Table 5–50. SDRAM Buffer Address High (SDBUFA_HI) Register Bit/FIeld Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	AAI	Automatic address increment When set to 1, the address set in SDBUFA_HI, SDBUFA_LO is automatically incremented.
7–6	RSV	Reserved bits. Not used.
5–0	SDA21–SDA16	SDRAM address 1 Upper 6 bits of the SDRAM address in which data is read and written is set in this register. The address here is offset from the SDRAM area lead in 32-byte units.

Table 5–51. SDRAM Buffer Address Low (SDBUFA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SDA15	SDA14	SDA13	SDA12	SDA11	SDA10	SDA9	SDA8	SDA7	SDA6	SDA5	SDA4	SDA3	SDA2	SDA1	SDA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–52. SDRAM Buffer Address Low (SDBUFA_LO Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	SDA15–SDA0	SDRAM address 2 Lower 16 bits of the SDRAM address in which data is read and written is set in this register. The address here is offset from the SDRAM area lead in 32-byte units.

Table 5–53. SDRAM Buffer Transfer Control (SDBUF_CTL)

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BUFC	WM	WA	RSD
R/W	—	—	—	—	—	—	—	—	—	—	—	—	W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0	0

Table 5–54. SDRAM Buffer Transfer Control (SDBUF_CTL) Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–4	RSV	Reserved bits. Not used.
3	BUFC	Buffer clear When set to 1, all data in data registers is cleared. This bit is automatically cleared.
2	WM	Modified data write When set to 1, only the data in registers where write was performed are written into SDRAM. This register is automatically cleared when the transfer to SDRAM is finished.
1	WA	All data write When set to 1, all data in data registers is written into SDRAM. This register is automatically cleared when the transfer to SDRAM is finished.
0	RSD	All data read When set to 1, data is read from SDRAM into data registers. This register is automatically cleared when the transfer from SDRAM is finished.

5.5 DMA Controller

The external memory interface has a built-in DMA controller. This DMA transfers data from the SDRAM or a CS region to the SDRAM or a CS region (see Table 5–55).

5.5.1 Sources and Destinations Devices

To transfer SDRAM or an external general-purpose memory region to SDRAM or an external general-purpose memory region is possible without going through the MCU by the external bus DMA controller.

The DMA transfer is executed by setting the transfer origin device and transfer destination device in the proper registers in the external memory interface module and setting the total number of bytes to be transferred. After transmission ends, an interrupt can be generated to the MCU. Transfer from a given region to the same one is allowed.

Table 5–55. DMA Transfer Source and Transfer Destination Devices

TRANSFER SOURCE DEVICE	TRANSFER DESTINATION DEVICE
External general-purpose memory CS0	External general-purpose memory CS0
External general-purpose memory CS1	External general-purpose memory CS1
External general-purpose memory CS2	External general-purpose memory CS2
External general-purpose memory CS3	External general-purpose memory CS3
External general-purpose memory CS4	External general-purpose memory CS4
SDRAM	SDRAM

The following register lets the user specify the source and destination device.

Table 5–56. DMA Transfer Device Selection (DMADEVSEL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	SDV2	SDV1	SDV0	RSV	DDV2	DDV1	DDV0
R/W	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	—	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	0	0	0	—	0	0	0

Table 5–57. DMA Transfer Device Selection (DMADEVSEL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–7	RSV	Reserved bits. Not used.
6–4	SDV2–SDV0	Specifies DMA transfer source device SDV[6:4] Transfer destination device 0 0 0 CS0 0 0 1 CS1 0 1 0 CS2 0 1 1 CS3 1 0 0 CS4 1 0 1 SDRAM
2–0	DDV2–DDV0	Specifies DMA transfer destination device DDV[2:0] Transfer destination device 0 0 0 CS0 0 0 1 CS1 0 1 0 CS2 0 1 1 CS3 1 0 0 CS4 1 0 1 SDRAM

5.5.2 Sources and Destinations Address and Transfer Size

The registers to configure the DMA transfer are shown below. The address used to configure the DMA is the address relative to the start of the memory region used. Because the transfer address needs to be a multiple of 4, the lower 2 bits of the SOURCEA_LO and DESTA_LO are ignored. The transfer size must always be specified in a multiple of 4. Therefore, bits TXB1 and TXB0 are ignored.

Table 5–58. Source Address High (SOURCEA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	SA25	SA24	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 5–59. Source Address High (SOURCEA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	SA25–SA16	Describes upper 10-bit address of the DMA transfer source byte address.

Table 5–60. Source Address Low (SOURCEA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–61. Source Address Low (SOURCEA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	SA15–SA0	Describes lower 16-bit address of the DMA transfer source byte address. Bits SA1 and SA0 are ignored and considered as 0.

Table 5–62. Destination Address High (DESTA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	DA25	DA24	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 5–63. Destination Address High (DESTA_HI Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	DA25–DA16	Describes upper 10-bit address of the DMA transfer destination byte address.

Table 5–64. Destination Address Low (DESTA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–65. Destination Address Low (DESTA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	DA15–DA0	Describes lower 16-bit address of the DMA transfer destination byte address. Bits DA1 and DA0 are ignored and considered as 0.

Table 5–66. DMA Transfer Size (DMASIZE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	TXB15	TXB14	TXB13	TXB12	TXB11	TXB10	TXB9	TXB8	TXB7	TXB6	TXB5	TXB4	TXB3	TXB2	TXB1	TXB0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5–67. DMA Transfer Size (DMASIZE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	TXB15–TSB0	Specifies total transfer size in bytes. Bits TXB1 and TXB0 are ignored and considered as 0. If 0x0 is used, no transfer will occur.

5.5.3 DMA Transfer Control

When the previously mentioned registers are set up, the user can start the DMA transfer by setting bit 0 of the DMA control register (DMACTL). This bit is automatically cleared when the DMA transfer is finished. Writing to this bit while a transfer is in progress has no effect. The request is then ignored.

The EMIF DMA controller can change the endianess property of the data transferred. This is controlled by the ENDI[1:0] bits of the DMACTL register.

Table 5–68. DMA Control (DMACTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	ENDI1	ENDI0	RSV	RSV	RSV	RSV	RSV	RSV	RSV	GO
R/W	—	—	—	—	—	—	R/W	R/W	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	0	0	—	—	—	—	—	—	—	0

Table 5–69. DMA Control (DMACTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–8	ENDI1–ENDI0	Controls endian switching ENDI1 ENDI0 Endian switching 0 0 D3 D2 D1 D0 –> D3 D2 D1 D0 0 1 D3 D2 D1 D0 –> D2 D3 D0 D1 1 0 D3 D2 D1 D0 –> D0 D1 D2 D3 1 1 same as combination ‘00’
7–1	RSV	Reserved bits. Not used.
0	GO	DMA transfer start bit When 1 is set, the DMA transfer is started. After transfer is finished, it is automatically cleared to 0.

5.6 External CPU Interface

5.6.1 Replacing the Internal ARM7 by Another Device

Sometimes, it may be necessary to use a different device to control the DSC24 other than the internal ARM7. By setting the EXTCPU pin to high, access by an external device to the DSC24 internal registers and memory areas is possible. When in external CPU mode, access from the ARM inside the DSC24 is not possible because the internal ARM bus is switched to an external CPU I/F bus. The structure in this mode is shown in Figure 5–6.

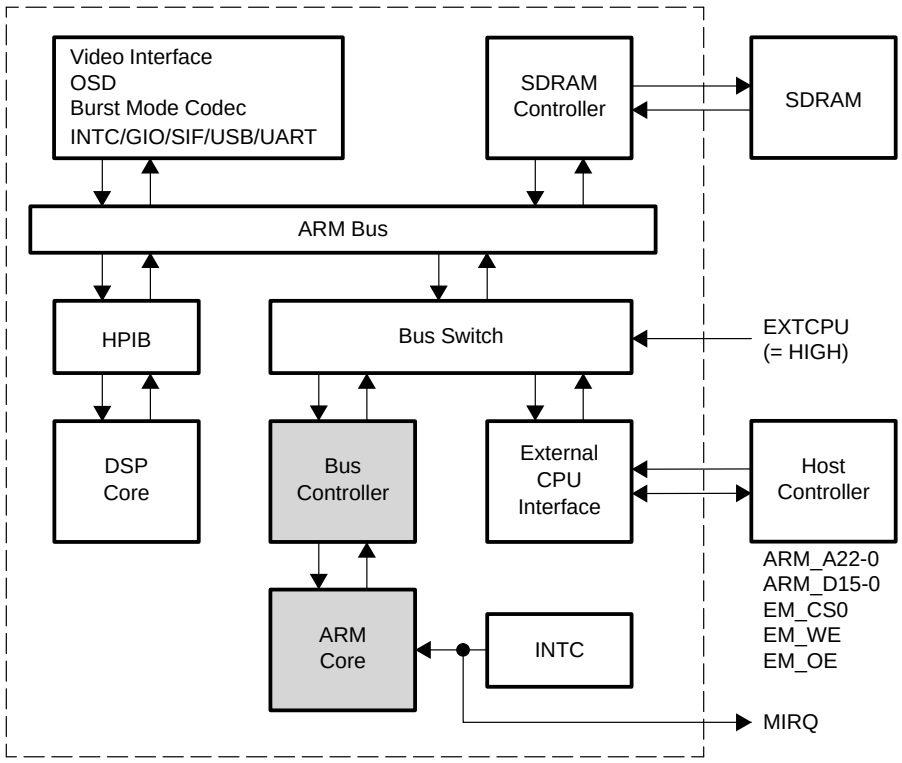


Figure 5–6. External CPU Interface Mode

In case the internal ARM processor needs to be used, the device must be set to internal CPU mode (EXTCPU pin = 0) during power-on and reset. The DSC24 does not allow dynamic control of the EXTCPU pin.

When the general input/output pin GIO20 is set to the MIRQ output, the internal interrupt request signal (IRQ) signal from the interrupt controller can be supplied outside the DSC24. This MIRQ pin is used to interrupt the external CPU.

When in external CPU mode, the clock controller does not drive a clock to the ARM core and the bus controller anymore. Both modules are stopped to save power.

5.6.2 Memory Access

The memory areas that can be accessed from an external host controller are all the MCU peripheral modules, SDRAM (16M bytes), and DSP internal memory (64K bytes). However, the addresses differ from the internal MCU memory map. Table 5–70 shows the memory map of the DSC24 as seen from the external CPU after a reset. Note that the portion of SDRAM in the range 0x018E:0000 to 0x020F:FFFF cannot be accessed by any external CPU.

Table 5–70. Memory Map (External CPU Mode)

AREA NAME	DSC24 ADDRESSES	EXTERNAL CPU ADDRESS	SIZE (BYTE)
Internal peripheral register	0x0003:0000 – 0x0003:0FFF	0x00:0000 – 0x00:0FFF	4K
DSP memory	0x0004:0000 – 0x0004:FFFF	0x01:00C0 – 0x01:FFFF	~64K
SDRAM	0x0090:0000 – 0x018D:FFFF	0x02:0000 – 0xFF:FFFF	16,256K

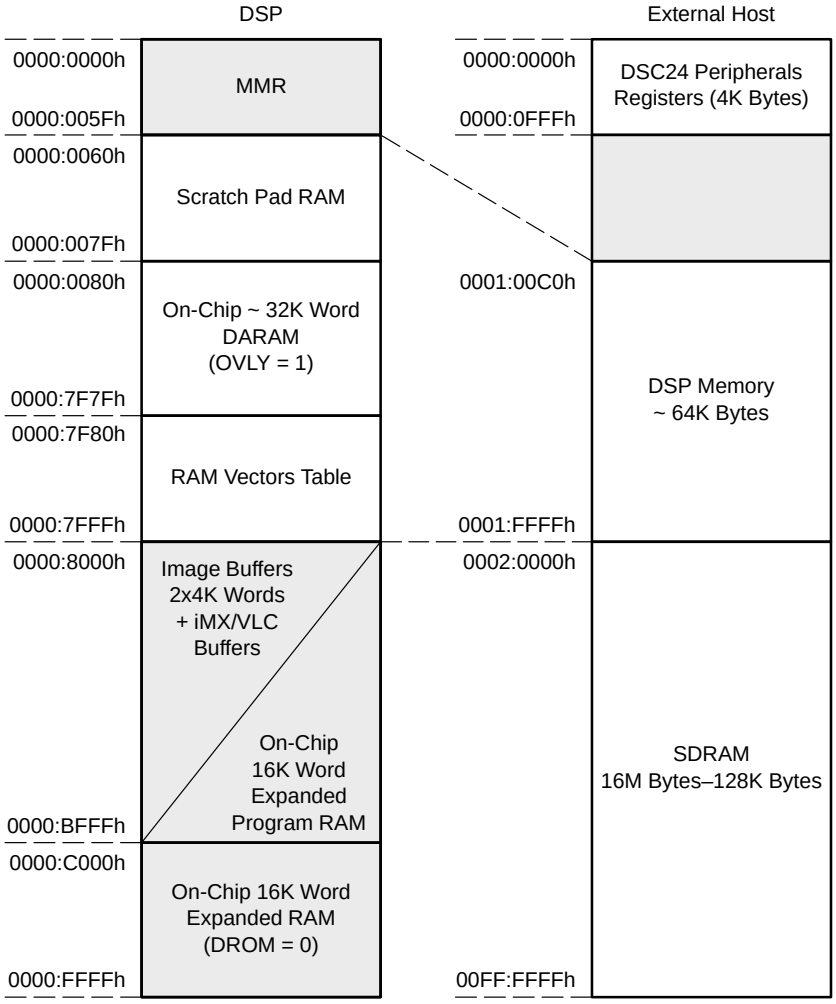


Figure 5–7. Memory Map in External CPU Mode

The external CPU cannot access the AMIF; however, its control and configuration registers can be accessed. The DMA function to external memory cannot be used.

When the EMIF controller is reconfigured to change the address where the SDRAM block of memory starts, the start address of this block as seen by the external CPU is not modified.

5.6.3 Bus Access

When in external CPU I/F mode, ARM_A22–ARM_A0, EM_CS0, EM_WE, and EM_OE become input pins. Also, the data bus is fixed at 16 bits and uses ARM_D15–ARM_D0. Both read and write are half word (16 bits) access.

If a read is performed after another read, EM_OE or EM_CS0 must return to a high level in between the accesses.

5.6.4 Access Time

The ARM clock synchronizes the external CPU I/F module. The interface has been designed to be able to operate asynchronously with the external device. It can easily interface with various devices, however dynamic wait control is not performed by the interface with the external HOST controller. The number of WAITs of the external HOST controller must be set according to the maximum access time.

The required number of wait cycles varies depending on the settings of the DSC24 ARM clock (CLK_ARM) frequency, SDRAM, and DSP clock frequency.

The number of cycles of the ARM clock is shown below.

- Internal peripheral other than USB: 4 cycles
- USB/DSP memory/SDRAM: 4 + n cycles (n: Peripheral WAIT cycles)

The SDRAM access time differs depending on the state of the bus. Since the SDRAM controller has a 16K-word FIFO, when SDRAM is accessed via FIFO, it results in the same access time as a peripheral register.

Figure 5–8 and Figure 5–9 show examples of access in the external CPU mode.

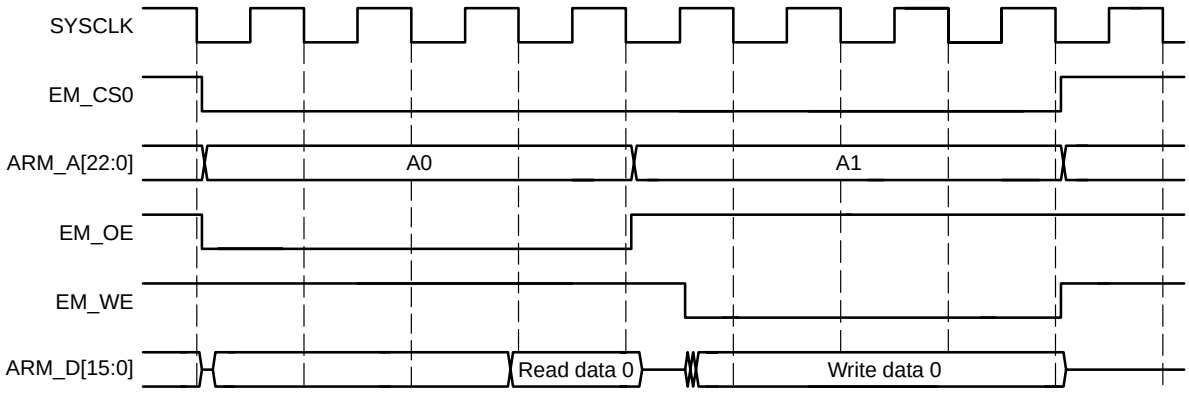


Figure 5–8. External Host Interface for DSC24 No-Wait Peripherals

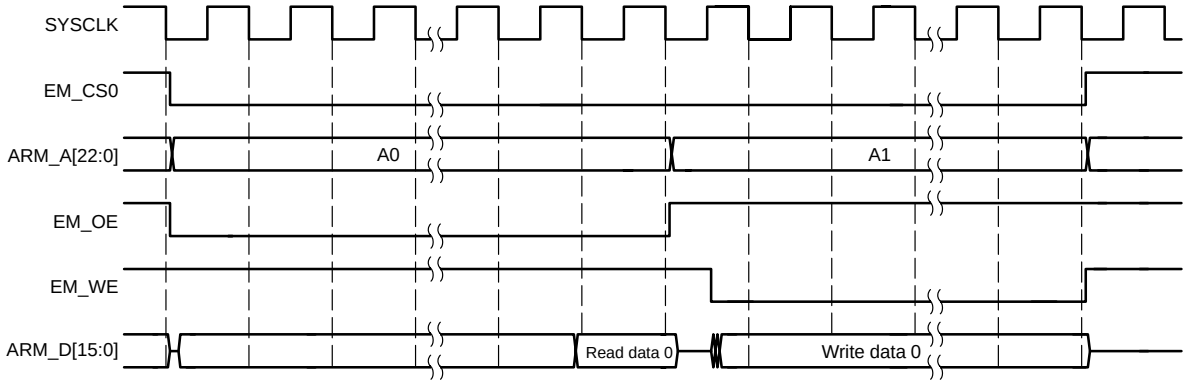


Figure 5–9. External Host Interface for DSC24 Handshake Peripherals (USB, DSP)

Table 5–71 gives an example of the required software wait cycles to interface with an SH-3 device running at 27 MHz. In this example, the SDRAM frequency is 54 MHz, the DSP frequency is 94 MHz, and the SYSCLK is either 27 MHz or 40 MHz.

Table 5–71. Required Software WAIT Cycles

SYSCLK FREQUENCY	27 MHz		40 MHz	
Read/write	R	W	W	R
Internal peripheral	2	3	1	2
DSP memory	3	7	2	6
SDRAM	6	7	6	7

5.6.5 BUSC Registers in External CPU Mode

When in external CPU mode, the bus controller registers (ECR, EBYTER, EBITR, REVR) are not accessible by the external device. Instead, the host CPU sees the host controller interface (HRCIF) registers. Table 5–72 gives their address as seen from the host.

Table 5–72. HRCIF Register List

OFFSET	ADDRESS	REGISTER NAME	DESCRIPTION
0x00	0x0000:0900	REVR	Device revision ID
0x01	0x0000:0902	RTRGSEL	Data read trigger select

The device revision ID register is identical to the one found in the bus controller.

Table 5–73. Device Revision (REVR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	REV7	REV6	REV5	REV4	REV3	REV2	REV1	REV0
R/W	—	—	—	—	—	—	—	—	R	R	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Table 5–74. Device Revision (REVR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits
7–0	REV7–REV0	Represents device version

The revision number is represented with one decimal digit. For example, REV=0x10 corresponds to silicon version 1.0.

Table 5–75. Data Read Trigger Select (RTRGSEL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RTRG
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 5–76. Data Read Trigger Select (RTRGSEL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits
0	RTRG	Data read trigger select bit 0: CS falling edge 1: OE falling edge

When the host reads the data from the DSC24, the event used to start the read access can either be the falling edge of CS or the falling edge of OE.

5.7 Bus Open

The EMIF bus can be opened for an external device by the BUS_REQ signal.

In the device operating modes 0, 1, and 3, at the falling edge of the BUS_REQ pin, an interrupt to the DSC24 MCU is generated (EMIF1). If the MCU is ready, by setting the external memory bus open register in the external memory interface module, all IF signals (ADDRESS/DATA/$\overline{CS}$/WE/OE) are opened (output Hi-Z)

All IF signals are driven when BUS_ACK rises in the next clock after BUS_REQ rises. By using this function, it is possible to share memory with external processors.

Timing is shown in Figure 5–10. The external memory bus outputs Hi-Z during the reset period and while BUS_ACK is low.

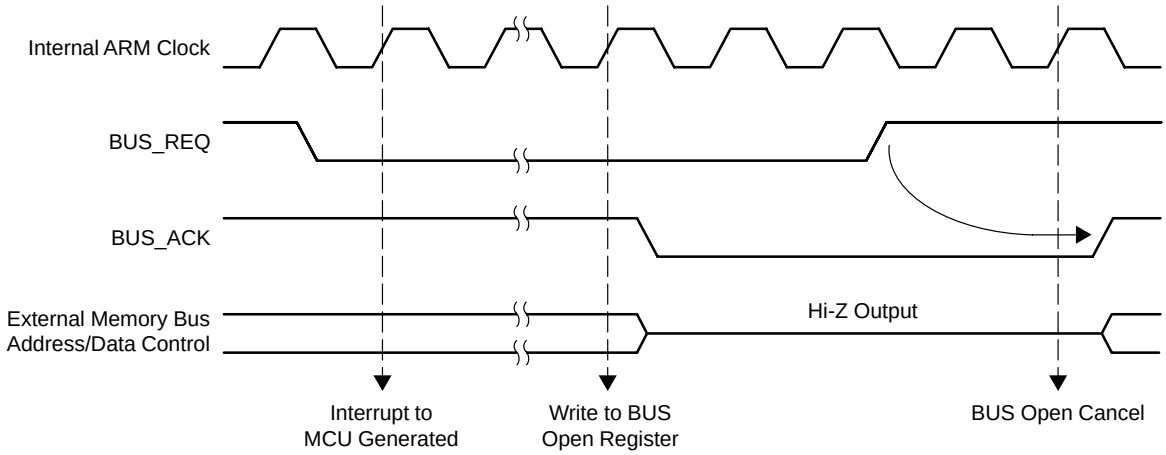


Figure 5–10. Bus Open, Acknowledgment Sequence

In addition, in the DSC24 the SDRAM bus can be opened to an external device. All SDRAM buses (ADDRESS, DATA, RAS, CAS, CS, WE, DQM, CKE, CLK) can be opened (output Hi–Z) by setting the SDRAM bus open registers in the external memory interface module. For SDRAM buses, bus open is not automatically canceled even when BUS_REQ becomes high. Setting cancel in the SDRAM bus open register turns on an SDRAM bus. By using this function, SDRAM can be shared between external devices. During the reset period, the SDRAM address and data bus are in a high-impedance state. While the SDRAM bus open register is on, the SDRAM control bus is in that state as well.

The BUS_REQ and BUS_ACK pins are not available in device operating mode 2. The work around is to use GIO external interrupts pins, since the BUSRLS register bits can be controlled in the software.

The bus open control register is shown below:

Table 5–77. Bus Release Control (BUSRLS) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	SDREL	EMREL
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0

Table 5–78. Bus Release Control (BUSRLS) Register Bit/Field Descriptions

Bits	Register Name	Description
15–2	RSV	Reserved bits. Not used.
1	SDREL	SDRAM bus open control bit 0: SDRAM bus is not opened Cleared by writing 0 1: SDRAM bus is opened
0	EMREL	External memory bus open control bit. 0: External memory bus is not opened Cleared when 0 is written or BUS_ACK becomes 1 1: External memory bus is opened

6 DSP Subsystem and Coprocessors Subsystem

This chapter deals with the DSP that is found in the DSC24 as well as with the two DSP coprocessors (the imaging extension module and the variable length coder block).

6.1 Introduction

The DSP subsystem is based on the TMS320C5409 DSP from the TI catalog. This DSP core and its peripherals are fully code-compatible with other C54x products. The DSP software development tools for the C54x are mature and can be used to develop, compile, and debug the DSP code.

The coprocessor subsystem includes several blocks of volatile memories, an imaging extension coprocessor (iMX), and a variable length-coding accelerator (VLC).

This chapter gives an overview of the DSP core and focuses on its peripherals. To get more details on the architecture of this 16-bit DSP core or on the tools associated with it, see the C54x user's guides and tools documentation.

To realize image processing, image compression processing, and voice processing, the DSP controls the image buffers DMA (IB-DMA) transfer data to and from internal or expanded memory, and the iMX/VLC accelerators block of memory. Since programming of the DSP is flexible and easy, the user can improve the image processing flow and add new functions.

6.2 DSP Core and Peripherals

6.2.1 Features of the DSP Core

The C54x devices are fixed-point digital signal processors (DSPs) from the Texas Instruments' TMS320 family. The C54x CPU, with its advanced modified Harvard architecture, features minimized power consumption and a high degree of parallelism.

This processor has one program memory bus and three data memory buses. The processor also provides an arithmetic logic unit (ALU) that has a high degree of parallelism, application-specific hardware logic, on-chip memory, and additional on-chip peripherals. The basis of the operational flexibility and speed of this DSP is a highly specialized instruction set.

Separate program and data spaces allow simultaneous access to program instructions and data providing the high degree of parallelism. Two read operations and one write operation can be performed in a single cycle. Instructions with parallel store and application-specific instructions can fully utilize this architecture. In addition, data can be transferred between data and program spaces. Such parallelism supports a powerful set of arithmetic, logic, and bit manipulation operations that can all be performed in a single machine cycle. In addition, this processor includes the control mechanisms to manage interrupts, repeated operations, and function calls.

The DSP core includes the following features:

- Low-power C54x DSP CPU
- Software-programmable wait-state generator with bank-switching wait state logic
- External memory interface to access the additional on-chip DSP memory
- Program space
- Data space
- I/O space
- Scan-based emulation logic

In the DSC24, the wait-state generator and the external memory interface are used to connect the DSP core to the expansion memory of the DSP subsystem and to the image buffers of the coprocessors subsystem. The DSP core has no direct access to memory located outside the DSC24 chip.

The maximum DSP cycle frequency is programmable up to 94.5 MHz (see the *TMS320DSC24 GHK* data manual, literature number SPRS195, for the exact value). The MCU handles the configuration of the clock.

The DSP core features 32K words of internal RAM (DARAM) that are mapped in both program and data spaces.

6.2.2 DSP Peripherals

The DSP CPU core is associated with an HPI interface, an interrupt handler, a parallel interface XIO, two multichannel buffered serial ports (McBSPs), and a JTAG interface.

The JTAG interface is connected to the JTAG interface of the ARM core through logic. The logic and the interface are described with more details in the data sheet.

The parallel interface is used to connect the core to the expanded memory and some of the peripherals control registers. This interface is also used to control the VLC engine and iMX coprocessor. See the appropriate chapters to get more details on each one of these points.

The host port interface (HPI) of the DSP is connected to the DSP controller of the MCU subsystem. This connection cannot be modified. The HPI cannot serve any other purpose other than to communicate back and forth with the MCU. The HPI bridge is described in the *ARM-DSP Communication* chapter.

The DSP core also features a DMA controller that is used to transfer data between the image buffers, the expanded memory block, the internal memory, and the McBSPs.

The PLL that can normally be found on a standard C54x device has been removed. The DSP PLL is now part of the clock controller module and fully controlled by the MCU.

The only peripherals that can be used by the DSP to communicate directly with external devices to the DSC24 are the two multichannel buffered serial ports and the XF pin. The DSP controls the XF signal by writing into the proper memory map register. This signal is multiplexed with the MCU GIO14 pin. This signal can also be read by the MCU through the DSP controller MMRs.

6.2.2.1 Multichannel Buffered Serial Ports

The DSC24 has two high-speed, full-duplex multichannel buffered serial ports (McBSPs), one of which (McBSP1) has its pins multiplexed with the MCU GIO8 to GIO13. The McBSP ports allow direct interface to other C54x devices, codecs, and other devices in a system. The McBSPs used in the DSP are based on the standard serial port interface found on other TMS320C54x devices. The two McBSPs that are accessible to the DSP are named McBSP0 and McBSP1.

The McBSPs provide:

- Full-duplex communication
- Double-buffered data registers that allow a continuous data stream
- Independent framing and clocking for receive and transmit

Capabilities

In addition, the McBSPs have direct interface to:

- T1/E1 framers
- MVIP switching-compatible and ST-BUS-compliant device
- IOM-2 compliant devices
- AC97 compliant devices
- Some IIS-compliant devices
- Multichannel transmit and receive of up to 32 channels
- Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected A/D and D/A devices
- A wide selection of data sizes, including 8, 12, 16, 20, 24, and 32 bits

- μ-law and A-law companding
- External shift clock or an internal, programmable-frequency shift clock for data transfer
- Autobuffering capability through the six-channel DSP-DMA controller
- Programmable polarity for both frame synchronization and data clocks

External Interface

The McBSPs consist of separate transmit and receive channels that operate independently. The external interface of each McBSP consists of the following pins:

- BCLKX: Transmit reference clock
- BDX: Transmit data
- BFSX: Transmit frame synchronization
- BCLKR: Receive reference clock
- BDR: Receive data
- BFSR: Receive frame synchronization

The McBSP0 port has an extra pin:

- BCLKS: External clock reference for the programmable clock generator

The BCLKS pin is an additional signal to provide a clock reference to the McBSP programmable clock generator. BCLKS is not implemented on McBSP1.

Transmitter/Receiver Pins

The six pins listed are functionally equivalent to the pins of the previous serial port interface pins in the C54x generation of DSPs. On the transmitter, transmit frame synchronization and clocking is indicated by the BFSX and BCLKX pins, respectively. The CPU or DSP-DMA initiates transmission of data by writing to the data transmit register (DXR). Data written to DXR is shifted out on the BDX pin through a transmit shift register (XSR). This structure allows DXR to be loaded with the next word to be sent, while the transmission of the current word is in progress.

On the receiver, receive frame synchronization and clocking is indicated by the BFSR and BCLKR pins, respectively. The CPU or DSP- DMA can read received data from the data receive register (DRR). Data received on the BDR pin is shifted into a receive shift register (RSR) and then buffered in the receive buffer register (RBR). If the DRR is empty, the RBR contents are copied into the DRR. If not, the RBR holds the data until the DRR is available. This structure allows storage of the two previous words while the reception of the current word is in progress.

Data Movement

The CPU and DSP-DMA move data to and from the McBSPs and synchronize transfers based on McBSP interrupts, event signals, and status flags. The DSP-DMA is capable of handling data movement between the McBSPs and memory with no intervention from the CPU.

Programmable Functions

In addition to the standard serial port functions, the McBSP provides programmable clock and frame sync generation. Among the programmable functions are:

- Frame synchronization pulse width
- Frame period
- Frame synchronization delay
- Clock reference (internal vs external)
- Clock division
- Clock and frame sync polarity

The on-chip companding hardware allows compression and expansion of data in either μ-law or A-law format. When companding is used, transmit data is encoded according to specified companding law and received data is decoded to 2s-complement format.

Multiple Channel Selection

The McBSPs allow multiple channels to be independently selected for the transmitter and receiver. When multiple channels are selected, each frame represents a time-division multiplexed (TDM) data stream. In using TDM data streams, the CPU may only need to process a few of them. Thus, to save memory and bus bandwidth, multichannel selection allows independent enabling of particular channels for transmission and reception. Up to 32 channels can be enabled.

Data Clock Generation

The user selects from a variety of data bit-clocks independently for the receiver and transmitter. These options include:

- The input clock to the sample rate generator is either the CPU clock or a dedicated external clock input (CLKS) for McBSP0.
- The input clock (CPU clock or external clock CLKS for McBSP0) source to the sample rate generator can be divided down by a programmable value.

Because the CLKS pin is not available on McBSP1, the user must always write a 1 into the CLKSM bit of the sample rate generator 2 register (SRGR2) of the DSP.

6.2.2.2 Direct Memory Access Controller (DSP-DMA)

Features

The DSP subsystem includes a six-channel DMA controller for performing data transfers independent of the CPU. The DMA controller has access to off-chip memory and moves data from expanded DSP memory to internal memory (or internal to expanded). The primary function of the DMA controller is to manage data transfers between on-chip memory and the serial ports.

The DMA controller includes the following features:

- Operates independently of the CPU
- Six independent channels (DMA controller keeps track of the context of six independent block transfers)
- Higher priority for memory accesses than CPU
- Each channel has an independently programmable priority level
- Each channel's source and destination address registers include configurable indexing modes. The address can remain constant, be post incremented/decremented, or be adjusted by a programmable value.
- Each read or write transfer may be initialized by selected events (internally only)
- Each DMA channel can interrupt the CPU upon completion of a half or entire block transfer
- Supports double word transfers; i.e., a 32-bit transfer of two 16-bit words (internally only)

The DSP-DMA cannot access the shared memory if its access is switched to another module (iMX, VLC, or IB–DMA).

The DMA memory map, as shown in Figure 6–1, allows DMA transfers to be unaffected by the status of the MP/MC, DROM, and OVLY bits.

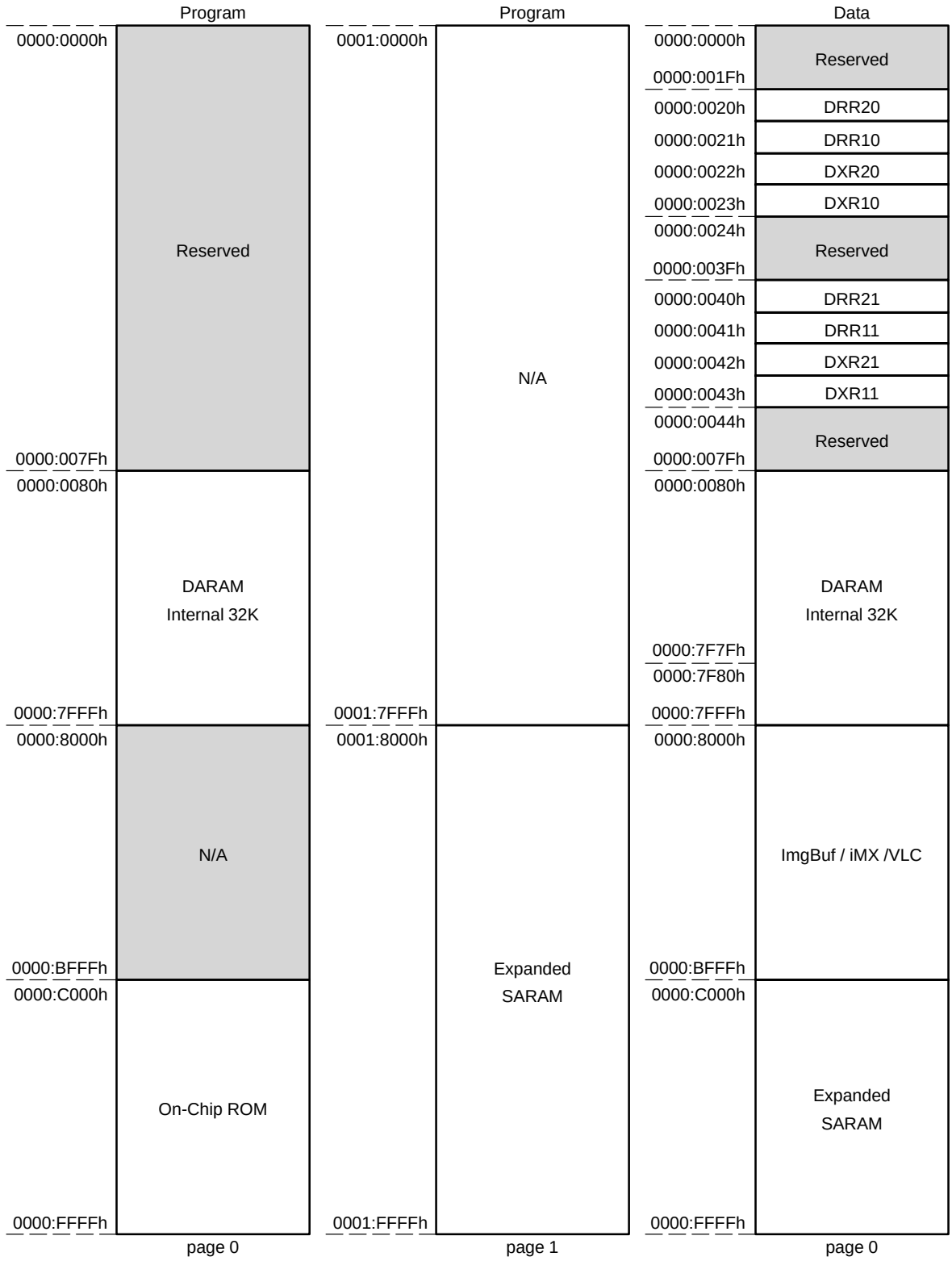


Figure 6–1. DMA Memory Map

DMA Expanded Access

The DSP DMA supports expanded memory accesses (external to the DSP core, on-chip access). A maximum of two DMA channels can be used for expanded memory accesses. These types of accesses require nine cycle's minimum for writes and 13 cycle's minimum for reads.

The control of the bus is arbitrated between the CPU and the DMA. While the DMA or CPU is in control of the XIO bus, the other is held off via wait states until the current transfer is complete. The DMA takes precedence over XIO requests. CPU and DMA arbitration testing is performed for each XIO bus cycle regardless of the bus activity.

Only two channels are available for expanded memory accesses. One for reads and one for writes. Single-word (16-bit) transfers are supported for expanded memory accesses. The DMA does not support transfers from peripherals to expanded memory. transfers from expanded memory to the peripherals, or transfers between two expanded memory data locations.

The DSC24 DSP-DMA mode control register (DMMCRx) has two additional bits; DLAXS (DMMCRn[5]) and SLAXS (DMMCRn[11]). These bits specify the on-/off-core memory for the source and destination of the program, data, and I/O spaces.

When DLAXS is set to 0 (default), the DMA does not perform an expanded memory access for the destination. When DLAXS is set to 1, the DMA performs an expanded memory access to the destination location.

When SLAXS is set to 0 (default), the DMA does not perform an expanded memory access for the source. When DLAXS is set to 1, the DMA performs an expanded memory access from the source location.

Two registers are available to the DSP-DMA to support DMA accesses to/from DMA expanded data memory. The DMA extended source data page register (XSRCDP[6:0]) is located at subbank address 028h. The DMA extended destination data page register (XDSTDP[6:0]) is located at subbank address 029h. The DMA memory map allows DMA transfers to be unaffected by the status of the MP/MC, DROM, and OVLY bits.

6.2.2.3 Hardware Timer

The DSP subsystem features one independent software-programmable timer. Three memory-mapped registers (MMRs) control the operation of the timer. These registers are the timer control register (TCR), the timer period register (PRD), and the timer counter register (TIM). The timer resolution is the CPU clock rate of the DSP. The timer has a 20-bit dynamic range. The timer consists of a programmable 16-bit main counter (TIM) and a programmable 4-bit prescaler.

The 4-bit prescaler drives the main counter, which decrements by one at every CPU clock. Once the prescaler reaches zero, the 16-bit counter decrements by one. When the 16-bit counter decrements to zero, a maskable interrupt (TINT) is generated and the counter is reloaded with the period value defined in the PRD. The timer can be stopped, restarted, reset, or disabled via the bits of the timer control register (TCR).

For more information, see the *TMS320*C54x *DSP Reference Set Volume 1: CPU and Peripherals*, literature number SPRU131.

6.3 DSP Subsystem Memory

6.3.1 Overview of the DSP Subsystem Memory

Figure 6–2 shows the DSP subsystems. An expansion memory (RAM outside the DSP core) made of 32K words of SARAM and a 16K words of shared memory for image processing (image buffer) is connected to the DSP expansion memory bus (XIO 16-bit data bus). Since the DSP (C5409) has 32K words of internal memory, a total of 80K words is used for DSP program and data regions. In addition to DSP, VLC, iMX image processing hardware accelerator, the IB-DMA controller (in the coprocessor subsystem) can access some shared memory blocks. Since the shared buffer is mapped to the DSP memory addresses, it is accessed directly. The DSP memory is constructed from three independent spaces: program, data, and I/O. SARAM is mapped to program and data spaces. Shared memory is mapped to the data space.

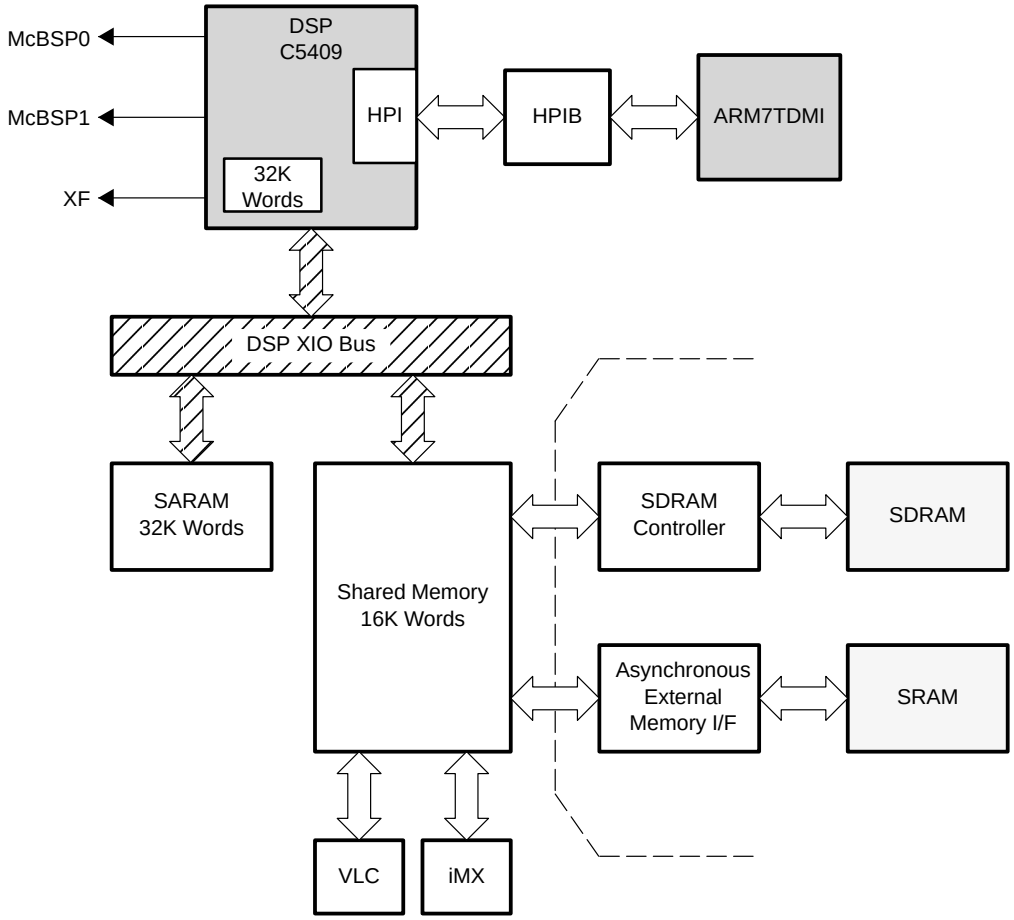


Figure 6–2. DSP and Coprocessor Subsystems Block Diagram

6.3.1.1 Program Data and I/O Spaces

The DSC DSP memory is organized into three individually selectable spaces: program, data, and I/O. Within any of these spaces, DARAM, ROM, SARAM, or memory-mapped peripherals can reside either within the DSP core or off the core.

The program memory space contains the instructions to execute, as well as the tables used in the execution. The data-memory space stores data used by instructions. The I/O memory space interfaces to the iMX, VLC engines, and the image buffer controller.

There are three CPU status register bits that affect memory configuration. The MP/MC, OVLY, and DROM bits are located in the processor mode status register (PMST). The effects of these bits are explained later in this chapter.

6.3.1.2 Memory Map

The DSP subsystem offers 32K words (16 bits) of DSP internal DARAM memory, 32K words of single access expanded memory and 16K words of ROM memory. The coprocessors subsystem includes 16K words of shared memory.

Figure 6–3 shows the DSP memory map after a reset. The data memory space goes from address 0x0h to 0xFFFFh, the program space page 0 is within the addresses boundary 0x0h and 0xFFFFh, and the program space page 1 goes from 0x10000h to 0x1FFFFh. Each block of memory has a width of 16 bits when seen by the DSP. Some of the blocks are available for both program and data spaces.

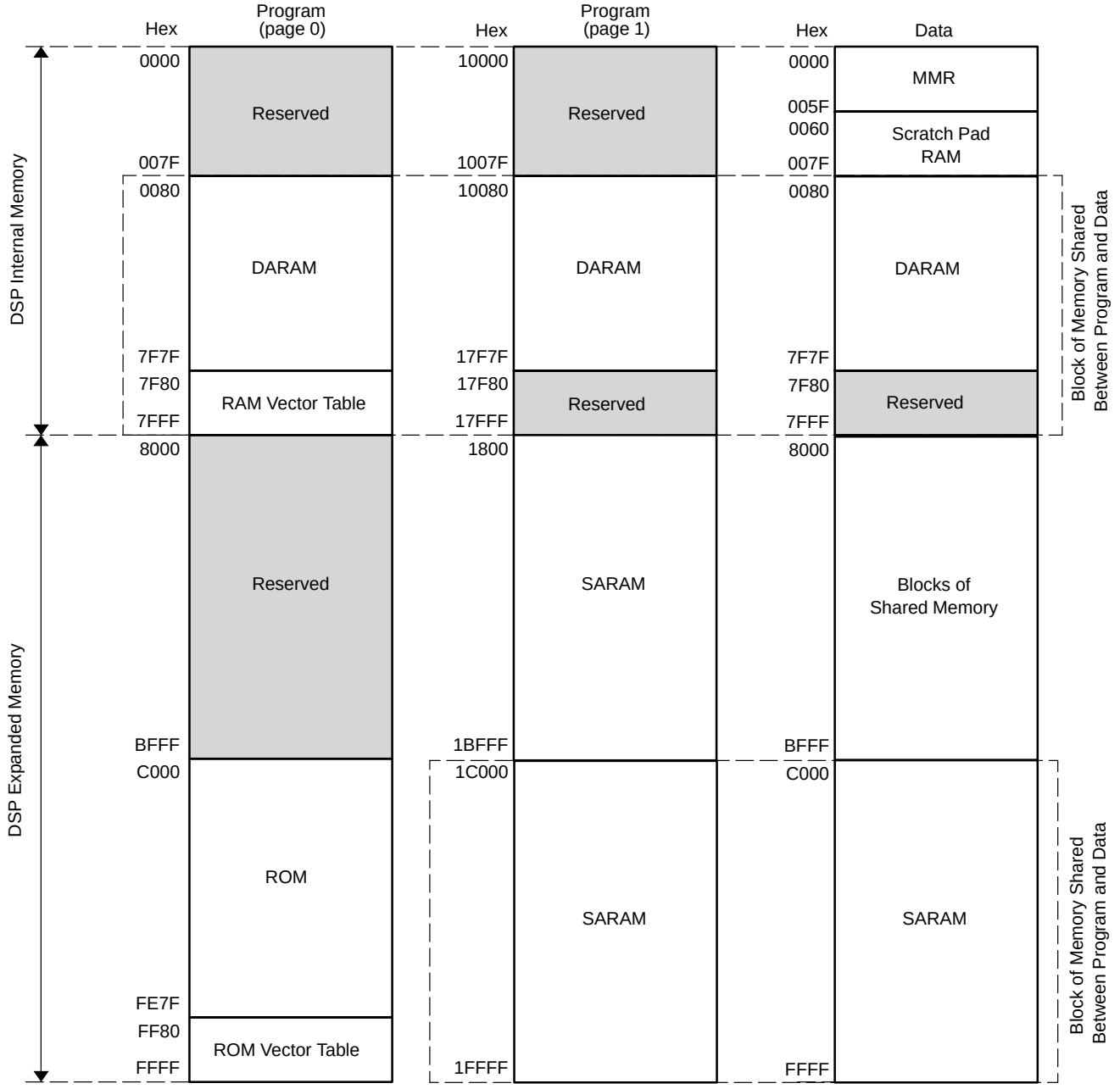


Figure 6–3. DSP Subsystem Memory Map at Reset

The portion of memory that is marked as *DSP Expanded Memory* is seen as external memory for the DSP core, however, it is still on-chip. The DSC24 does not allow the DSP to access directly off-chip memory like the SDRAM.

6.3.1.3 I/O Memory Space

The C54x devices offer an I/O memory space in addition to the program-memory and data-memory spaces. The I/O memory space is a 64K-word address space (0000h–FFFFh) and is external to the DSP core. Two instructions, PORTR and PORTW, are used to access this space. Read timings vary from those of the program-memory and data-memory spaces to facilitate access to individual I/O mapped devices rather than to memories.

On the DSC24, only some addresses from this space are used. Registers in the image buffer, iMX, and VLC are mapped to I/O spaces. Table 6–1 shows the address allocation of each module.

Table 6–1. I/O Space

I/O ADDRESS	MODULE NAME
0x0000h–0x0009h	Image buffer
0x0010h–0x0014h	iMX
0x0020h–0x0037h	VLC
0xFFFFh	CLOCK controller / ARM interrupt

6.3.2 DSP Internal Memory

6.3.2.1 Organization of the Memory

The DSP subsystem features 32K x 16 bits of on-chip DARAM. The OVLY bit controls the internal memory configuration. After a reset, this bit is set to 1. In this mode, the DARAM memory block is shared between data and program space.

It is possible to change this bit to 0 by software. However, it is not recommended as the internal memory is not available anymore to the store program and the vector table has to be relocated in the expanded memory area.

The MCU through the HPI bridge can access the DSP internal memory. This capability is explained in greater detail within the chapter on the ARM–DSP communication.

6.3.2.2 DARAM Properties

The internal memory of the DSP is built from four blocks of 8K x16-bit DARAM. The DSP CPU performs two accesses to a DARAM block in one machine cycle (two reads in one cycle or a read and a write in one cycle). It also performs multiple accesses to the separate memory blocks in one machine cycle.

Table 6–2 shows the RAM block organization for the DSC24 DSP DARAM memory. The horizontal lines in the table indicate block boundaries. The organization of the first 1K of DARAM includes the memory-mapped CPU and peripheral registers, 32 words of scratch pad DARAM, and 896 words of DARAM.

Table 6–2. Internal DSP RAM Block Organization

MEMORY BLOCK	MEMORY RANGE
MMR	0000–005F
DARAM0	0060–1FFF†
DARAM1	2000–3FFF
DARAM2	4000–5FFF
DARAM3	6000–7FFF

† Addresses 0x0060 to 0x007F are available only in the data memory area (scratch pad)

6.3.2.3 Access of DSP Internal Memory From MCU Side

The MCU can access DSP internal memory (32K words) between addresses 0x0080 and 0x7FFF through the DSP HPI port. The DSP internal memory can be accessed by a JTAG emulator connected to the MCU. However, it is highly recommended to use the debugger on the DSP side to access this bock of memory.

6.3.2.4 Reset and Relocatable Interrupt Vector Table

Communication between the DSP and DSP peripheral modules (image buffer, iMX, VLC) is performed using the respective interrupts. When the operation of each module ends, an interrupt is generated to the DSP. The interrupts of the DSP interrupt ports are as shown in Table 6–3 and Table 6–4:

Table 6–3. DSC24 Specific DSP Interrupt Ports

PORT NAME	MODULE NAME
INT0	ARM
INT1	iMX
INT2	VLC
INT3	Image buffer-DMA controller

Table 6–4. Interrupt Locations and Priorities

NAME	LOCATION	PRIORITY	FUNCTION
RS, SINTR	00	1	Reset (hardware and software reset)
NMI, SINT16	04	2	Non maskable interrupt
SINT17	08	—	Software interrupt #17
SINT18	0C	—	Software interrupt #18
SINT19	10	—	Software interrupt #19
SINT20	14	—	Software interrupt #20
SINT21	18	—	Software interrupt #21
SINT22	1C	—	Software interrupt #22
SINT23	20	—	Software interrupt #23
SINT24	24	—	Software interrupt #24
SINT25	28	—	Software interrupt #25
SINT26	2C	—	Software interrupt #26
SINT27	30	—	Software interrupt #27
SINT28	34	—	Software interrupt #28
SINT29	38	—	Software interrupt #29
SINT30	3C	—	Software interrupt #30
INT0 (MCU), SINT0	40	3	MCU interrupt
INT1 (iMX), SINT1	44	4	Imaging extension interrupt
INT2 (VLC), SINT2	48	5	Variable length coder interrupt
TINT, SINT3	4C	6	Timer interrupt
BRINT0, SINT4	50	7	McBSP #0 receive interrupt (default)
BXINT0, SINT5	54	8	McBSP #0 transmit interrupt (default)
Reserved	58–5F	—	Reserved
INT3 (IB–DMA), SINT8	60	11	Image buffer DMA module
Reserved	64	—	Reserved
BRINT1, SINT10, DMAC2	68	13	McBSP #1 receive interrupt (default)
BXINT1, SINT11, DMAC3	6C	14	McBSP #1 transmit interrupt (default)
DMAC4,SINT12	70	15	DSP-DMA channel 4 interrupt (default)
DMAC5,SINT13	74	16	DSP-DMA channel 5 interrupt (default)
Reserved	78–7F	—	Reserved

The reset, interrupt, and trap vectors are addressed in program space. These vectors are soft—meaning that the processor, when taking the trap, loads the program counter (PC) with the trap address and executes the code at the vector location. Four words are reserved at each vector location to accommodate a delayed branch instruction—either two 1-word instructions or one 2-word instruction. This allows branching to the appropriate interrupt service routine with minimal overhead.

At device reset, the reset, interrupt, and trap vectors are mapped to address FF80h in ROM space. However, these vectors can be remapped to the beginning of any 128-word page in program space after device reset. Loading the interrupt vector pointer (IPTR) bits in the PMST register with the appropriate 128-word page boundary address does this. The ROM based DSP code changes the IPTR value for the interrupt vector table to be located at address 0x7F80h.

6.3.2.5 Memory Mapped Registers and Scratch Pad RAM

Most of the DSP control registers are located within the address range 0x00 and 0x5F in the data space of the DSP subsystem memory map. The first 32 MMRs control the CPU itself, the following ones are used to control the DSP peripherals. The MMRs that control the iMX, VLC, and shared memory blocks are mapped into the I/O space and do not appear in Table 6–5 and Table 6–6.

The McBSPs need more than two registers each to be controlled. To work around the limited space in the MMR area, a sub-bank scheme is used to control the McBSP. This is described with further details in the *TMS320C54x DSP Reference Set Volume 5 – Enhanced Peripherals,* literature number SPRU302*.*

Table 6–5 and Table 6–6 give the name and a short description of the different memory mapped registers.

Table 6–5. CPU Memory-Mapped Registers

ADDRESS (Hex)	NAME	DESCRIPTION
0	IMR	Interrupt mask register
1	IFR	Interrupt flag register
2–5	—	Reserved for testing
6	ST0	Status register 0
7	ST1	Status register 1
8	AL	Accumulator A low word (bits 15–0)
9	AH	Accumulator A high word (bits 31–16)
A	AG	Accumulator A guard bits (bits 39–32)
B	BL	Accumulator B low word (bits 15–0)
C	BH	Accumulator B high word (bits 31–16)
D	BG	Accumulator B guard bits (bits 39–32)
E	T	Temporary register
F	TRN	Transition register
10	AR0	Auxiliary register 0
11	AR1	Auxiliary register 1
12	AR2	Auxiliary register 2
13	AR3	Auxiliary register 3
14	AR4	Auxiliary register 4
15	AR5	Auxiliary register 5
16	AR6	Auxiliary register 6
17	AR7	Auxiliary register 7
18	SP	Stack pointer
19	BK	Circular-buffer size register
1A	BRC	Block-repeat counter
1B	RSA	Block-repeat start address
1C	REA	Block-repeat end address
1D	PMST	Processor mode status register
1E	XPC	Program counter extension register
1E–1F	——	Reserved

Table 6–6. DSP Peripheral Memory-Mapped Registers

REGISTER	ADDRESS	DESCRIPTION	TYPE
DRR20	20h	Data receive register 2	McBSP #0
DRR10	21h	Data receive register 1	McBSP #0
DXR20	22h	Data transmit register 2	McBSP #0
DXR10	23h	Data transmit register 1	McBSP #0
TIM	24h	Timer register	Timer
PRD	25h	Timer period counter	Timer
TCR	26h	Timer control register	Timer
–	27h	Reserved	
SWWSR	28h	Software wait-state register	XIO Bus
BSCR	29h	Bank-switching control register	XIO Bus
–	2Ah	Reserved	
SWCR	2Bh	Software wait-state control register	XIO Bus
HPIC	2Ch	HPI control register	HPI
–	2Dh–37h	Reserved	
SPSA0	38h	McBSP0 sub-bank address register	McBSP #0
SPSD0	39h	McBSP0 sub-bank data register	McBSP #0
–	3Ah–3Fh	Reserved	
DRR21	40h	Data receive register 2	McBSP #1
DRR11	41h	Data receive register 1	McBSP #1
DXR21	42h	Data transmit register 2	McBSP #1
DXR11	43h	Data transmit register 1	McBSP #1
–	44h–47h	Reserved	
SPSA1	48h	McBSP1 sub-bank address register	McBSP #1
SPSD1	49h	McBSP1 sub-bank data register	McBSP #1
–	4Ah–53h	Reserved	
DMPREC	54h	DSP-DMA channel priority and enable control register	DMA
DMSA	55h	DSP-DMA sub-bank address register	DMA
DMSDI	56h	DSP-DMA sub-bank data register with auto increment	DMA
DMSDN	57h	DSP-DMA sub-bank data register	DMA
–	58h–5Fh	Reserved	

The advantage of using the scratch pad RAM that is located between address 0x60 and 0x7F is that it can be accessed anytime through an immediate memory access regardless of the data page pointer value. Some special assembly instructions like STM let the user do that. Additionally, since writing to control registers can incur latencies, instructions like STM operate early and avoid most pipeline problems.

6.3.3 DSP Expanded Memory

The DSP subsystem includes a memory-paging scheme to extend the number of addressable program and data space locations from 32K words to 64K words.

The extended program addresses are supported by the following eight instructions:

- FB[D] – Far branch
- FBACC[D] – Far branch to the location specified by the value in accumulator A or accumulator B
- FCALA[D] – Far call to the location specified by the value in accumulator A or accumulator B
- FCALL[D] – Far call
- FRET[D] – Far return
- FRETE[D] – Far return with interrupts enabled

- READA – Read program memory addressed by accumulator A and store in data memory
- WRITA – Write data to program memory addressed by accumulator A

For more information on these instructions, see the *TMS320C54x DSP Reference Set, Volume 2: Mnemonic Instruction Set*, literature number SPRU172.

When the OVLY bit is set, each page of program memory is made up of two parts: a common block of 32K words and a unique block of 32K words. All pages share the common block and each unique block is accessible only through its assigned page. The value of the program counter extension register (XPC) defines the page selection. At hardware reset, the XPC is initialized to 0.

6.3.3.1 Configurations of the Expanded Memory

The 32K words of expanded memory are available in both program and data space. The first 16K words of the expanded memory (address range 0x18000h to 0x1BFFFh) are always mapped into the program space (page 1). However, the second half of the expanded memory is mapped to both program space (address range 0x1C000h to 0x1FFFFh) and data space (address range 0xC000h to 0xFFFFh) after a reset. By changing the DROM configuration bit to 1, it is possible to replace the expanded memory mapped into the data space by the internal ROM memory. The MP/MC bit can also be set to 1, in such a case, the ROM is not available anymore within the program space.

Access to the first half of the program memory page 1 (address range 0x10000h to 0x17FFFh) results in accessing the internal DARAM area.

Table 6–7. DSP Memory Map in TMS320DSC24 (except DARAM)

ADDRESS		DROM = 0 MP/MC = 0	DROM = 1 MP/MC = 0	DROM = 0 MP/MC = 1	DROM = 1 MP/MC = 1
0x8000 0xBFFF	Program	NA	NA	NA	NA
0x8000 0xBFFF	Data	IMG/VLC/iMX	IMG/VLC/iMX	IMG/VLC/iMX	IMG/VLC/iMX
0xC000 0xFFFF	Program	Internal ROM	Internal ROM	NA	NA
0xC000 0xFFFF	Data	Expanded RAM	Internal ROM	Expanded RAM	Internal ROM
0x18000 0x1BFFF	Program	Expanded RAM	Epanded RAM	Expanded RAM	Expanded RAM
0x1C000 0x1FFFF	Program	Expanded RAM	Expanded RAM	Expanded RAM	Expanded RAM

NOTE: Same color column means same physical memory contents.

When added with the internal memory of the DSP, the total amount of memory available for the different MP/MC and DROM configurations is shown in Table 6–8. This table does not include the shared memory blocks or the read only memory.

Table 6–8. Amount of Memory Available for the DSP

	DROM	0	1	0	1
	MP/MC	0	0	1	1
Prog/Data	DARAM	32K	32K	32K	32K
Prog/Data	SARAM	16K	0K	16K	0K
Prog only	SARAM	16K	32K	16K	32K

Access to the first half of the program memory page 1 (address range 0x10000h to 0x17FFFh) results in accessing the internal DARAM area.

6.3.3.2 SARAM Properties

When expansion memory, shared memory, or I/O space is accessed, a 1-cycle wait must be inserted. Since the DSP has a programmable software wait generator, the software wait is generated by setting the number of waits in the WAIT control register (SWWSR register) in the DSP.

```
SWWSW = 0x1208h
```

It is important that the user configures the DSP XIO interface to add this wait state before trying to access any of the memory locations other than the ROM and DARAM block.

The exact number of cycle for expanded memory access depends on the instruction uses. See the TMS320C54x user's manual for more information.

For example, when transferring data from DSP internal memory to IMG or expanded memory, the following instruction needs 5 cycles + 1 wait = 6 cycles.

```
MVDD <DARAM>, <External>
```

However, for transferring data from IMG to expanded memory requires 4 cycles + 2 wait = 6 cycles.

```
MVDD <External>, <External> instruction
```

6.3.3.3 ROM Usage

The DSC24 DSP subsystem also includes 16K words of read-only memory (ROM).

The ROM block of memory is built within the DSP core itself. There is no need for extra wait states to be added when the user wants to access this location. The user can also access data from both expanded and ROM memories at the same time. The XIO bus of the DSP is not used for ROM access.

For program area, internal ROM can be used when MP/MC=0. For data area, internal ROM can be used when DROM=1. In this case, the physical program area and data areas are the same.

The ROM is made of the DSP initialization code.

6.3.4 Coprocessor Subsystem Shared Memory Blocks

In addition to the internal block of DARAM memory, the 32K words of expanded SARAM memory and the 16K words of ROM memory, the DSC24 DSP has access to several small blocks of memory that are used to pass data and commands to the iMX coprocessor and VLC accelerator. These blocks of memory are located within the expanded memory address range. The user must configure the device to add one wait state when accessing these blocks of memory.

The image buffer module has two major functions. One is wrapping the shared memory for the different DSP subsystem units and coprocessors to access the memory blocks correctly. The second one is to allow data DMA transfer between shared memory and SDRAM controller (IB-DMA).

6.3.4.1 Shared Memory Wrapper

There are a total six-shared memory blocks shown in Figure 6–4.

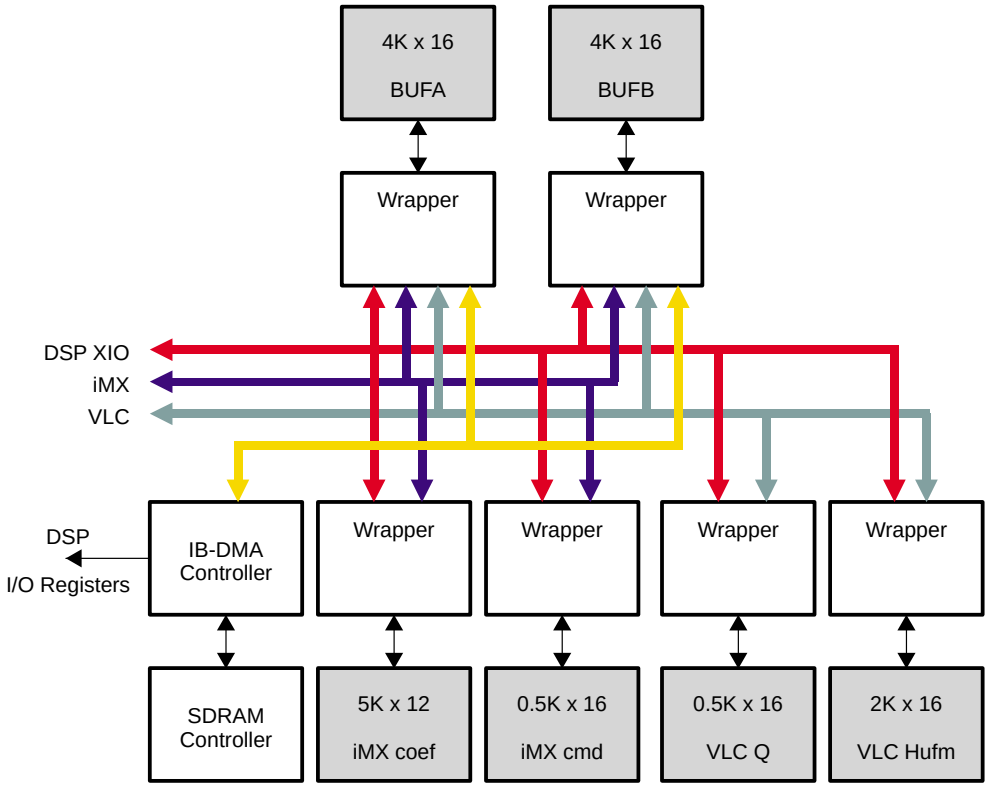


Figure 6–4. Image Buffer Block Diagram

BUF_A and BUF_B (2 × 4K × 16-Bit Memory)

The BUFA and BUFB memory blocks are used as the working memory for the image processing and are shared either by the DSP, iMX, or VLC. The wrapper is doing the switch logic for the shared memory. Note the buffer can not be shared by several of the access units at the same time and the access switch is controlled by the DSP.

iMX Coefficient Memory (5K × 12-Bit RAM)

The major function of this memory is to be used as the coefficient and temporary working space for iMX. The DSP needs to initialize the coefficient of the filter into this memory. The buffer switch is controlled by the DSP. For the same reason as BUF_A and BUF_B, a wrapper is used as the interface between the shared and access units. Note the upper 4 bits are MSB extended.

iMX Command Memory (512K × 16-Bit RAM)

The major function of this memory is to be used as the program command for the iMX. Before iMX is active, the DSP needs to download the program into this memory. The buffer switch is controlled by the DSP.

VLC Quantization Table Memory (512 × 16-Bit RAM)

The major function of this memory is to be used as the quantization table for the VLC engine. The table can be updated by the DSP. The buffer switch is controlled by the DSP.

VLC Huffman Table Memory (2K × 16-Bit RAM)

This memory should be used to store the Huffman tables for the VLC engine. The VLC tables can be updated by the DSP. The buffer switch is controlled by the DSP.

Access Switching

The DSP can access all memory blocks, but other modules have limited access. Selection of the modules that access each memory is controlled by the DSP setting registers in the I/O space. Therefore, in access to each memory, static switching is performed by the DSP and dynamic arbitration is not performed. The modules that can access each memory and the DSP memory address are shown in Table 6–9.

Table 6–9. Image Buffer Memory Structure

MEMORY	ACCESS MODULE				DSP ADDRESS	SIZE (WORDS)
	DSP	iMX	VLC	IB-DMA		
BUFA	✓	✓	✓	✓	8000H–8FFFH	4K
BUFB	✓	✓	✓	✓	9000H–9FFFH	4K
iMX coef	✓	✓	x	x	A000H–B3FFH	5K†
iMX cmd	✓	✓	x	x	B400H–B5FFH	512
VLC Q	✓	x	✓	x	B600H–B7FFH	512
VLC hufm	✓	x	✓	x	B800H–BFFFH	2K

† 1 WORD = 12 bits

The image buffer switch control register controls the switching. This register is located within the IB-DMA controller area at address 0x08 in the DSP I/O space (see Table 6–12) and described below.

Table 6–10. Image Buffer Switch Control (BUF_CTRL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	VLCHB	VLCQT	IMXCMD	IMXCOEF	IMBUFB1	IMBUFB0	IMBUFA1	IMBUFA0
R/W	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	0	0	0	0	0	0	0	0

Table 6–11. Image Buffer Switch Control (BUF_CTRL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–8	RSV	Reserved bits. Not used.
7	VLCHB	VLC Huffman shared memory switch 0: DSP 1: VLC
6	VLCQT	VLC quantization shared memory switch 0: DSP 1: VLC
5	IMXCMD	iMX command shared memory switch 0: DSP 1: iMX
4	IMXCOEF	iMX coefficient shared memory 0: DSP 1: iMX
3–2	IMBUFB	Image buffer B switch 00: DSP 01: iMX 10: VLC 11: Image buffer DMA
1–0	IMBUFA	Image buffer A switch 00: DSP 01: iMX 10: VLC 11: Image buffer DMA

The DSP can be simultaneously connected to both image buffers—bits [3:2] and [1;0] = 00. However, iMX, VLC, and IB-DMA should only be connected to a single image buffer at a time. Operation is undefined if bits [3:2] and [1:0] are loaded with the same nonzero pattern.

The access switch of to a block of shared memory from one module to another takes only one DSP cycle. The block switch is not synchronized with the DSP, iMX, or VLC memory accesses. This means, for example, the user must make sure the iMX does not need access to the image buffer A before switching this memory block to the VLC or DSP.

The iMX coefficient memory is only 12 bits wide. When writing a value into it, bits 15 to 12 are ignored. When a value is read, these bits represent the sign extension of the value (bits 15–12 = bit 11).

6.3.5 Clock Controller

As seen in the *I/O Memory Space* section, the DSP has access to a clock controller register located in its I/O memory space. This register lets the DSP control the clock of its coprocessors and memory modules.

Table 6–12. Clock Controller/ARM Interrupt (CCAI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IMBUF	IMXVLC	HINT
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 6–13. Clock Controller/ARM Interrupt (CCAI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved bits. Not used.
2	IMBUF	Image buffer clock 0 : Off 1 : On
1	IMXVLC	IMX / VLC clock 0 : Off 1 : On
0	HINT	HPI interrupt Interrupt to MCU 0 : Off 1 : On

Bit 0 concerns the HPI interface between the MCU and the DSP. This bit is described with more details in the *ARM / DSP Communication* chapter.

The IMXVLC and IMBUF bits of the CCAI register work in parallel with bits CIMG, CIMX, and CVLC of the MCU MOD1 register. To stop the clock of the iMX, VLC, or image buffer module, the corresponding clock controller bits on the MCU side as well as on the DSP side must bit set to 0.

- If MOD1.CIMG = 1 or CCAI.IMBUF = 1 then clock IMG is on else it is off
- If MOD1.CIMX = 1 or CCAI.IMXVLC = 1 then clock iMX is on else it is off
- If MOD1.CVLC = 1 or CCAI.IMXVLC = 1 then clock iMX is on else it is off

6.4 Image Buffer-DMA Controller

The image buffer DMA (IB-DMA) allows the DSP to transfer data between shared memory (BUFA or BUFB) and EMIF (SDRAM controller or AMIF block) without costing any DSP processing time. Before data transfer, the DSP needs to initialize the DMA request with following parameters.

1. Destination/source address in external memory region selected.(0x0000 and 0x0001)
2. Offset address in EMIF area.(0x0002)
3. Source/Destination address in BUFA or BUFB(0x0003)
4. Offset address in BUFA or BUFB(0x0004)
5. Access block size(X size Y size) (0x0005 and 0x0006)
6. Issuing DMA request (0x0007)

The IB-DMA handles the command request from DSP such as,

- Receives the DMA request from the DSP via XIO interface
- Acknowledges the DMA job done by issuing an interrupt to the DSP

The hardware interrupt signal is connected to the DSP's INT3. The DSP enables this interrupt (through interrupt mask register IMR) so that an interrupt service routine (ISR) is called upon transfer completion. Alternatively, the DSP disables this interrupt and polls the interrupt flag register (IFR) to sense completion status.

The IB-DMA handles the handshaking sequence to/from SDRAM or the external memory controller automatically.

6.4.1 Control Registers

Ten memory mapped registers control the shared memory blocks and the special DMA. These registers are located within the I/O space of the data memory. Their addresses go from port00 to port09.

Table 6–14. Shared Memory Control Registers

OFFSET	ADDRESS	REGISTER NAME	
0x00	port00	IBDDA_HI	IB-DMA destination address high
0x01	port01	IBDDA_LO	IB-DMA destination address low
0x02	port02	IBDDO	IB-DMA destination offset
0x03	port03	BUF_ADDR	Image buffer address
0x04	port04	BUF_LOFST	Image buffer offset
0x05	port05	DMA_XNUM	Number of DMA X bursts
0x06	port06	DMA_YNUM	Number of DMA Y bursts
0x07	port07	DMA_CTRL	Image buffer DMA control
0x08	port08	BUF_CTRL	Image buffer switch control
0x09	port09	IMG_MODE	Image buffer mode control

6.4.2 SDRAM 2D Access

The connection of image buffer A or B to the SDRAM controller or AMIF allows the DSP to access image data in the large central SDRAM storage (or external SRAM).

The DMA unit has the following capabilities and restrictions.

- It can access only image buffer A or image buffer B.
- DMA access can handle 2D addressing in external memory and image buffers (read a block of NxM out of a larger data unit).

Figure 6–5 specifies the number of pixels in the horizontal direction (DMA_XNUM) and the number of lines in the vertical direction (DMA_YNUM). By setting each memory lead address (BUF_ADDR, IBDDA) and line offset value (BUF_LOFST, IBDDO), block transfer of second-order image data is possible. This function is effective in cases where processing in macro block units is required, such as in JPEG compression.

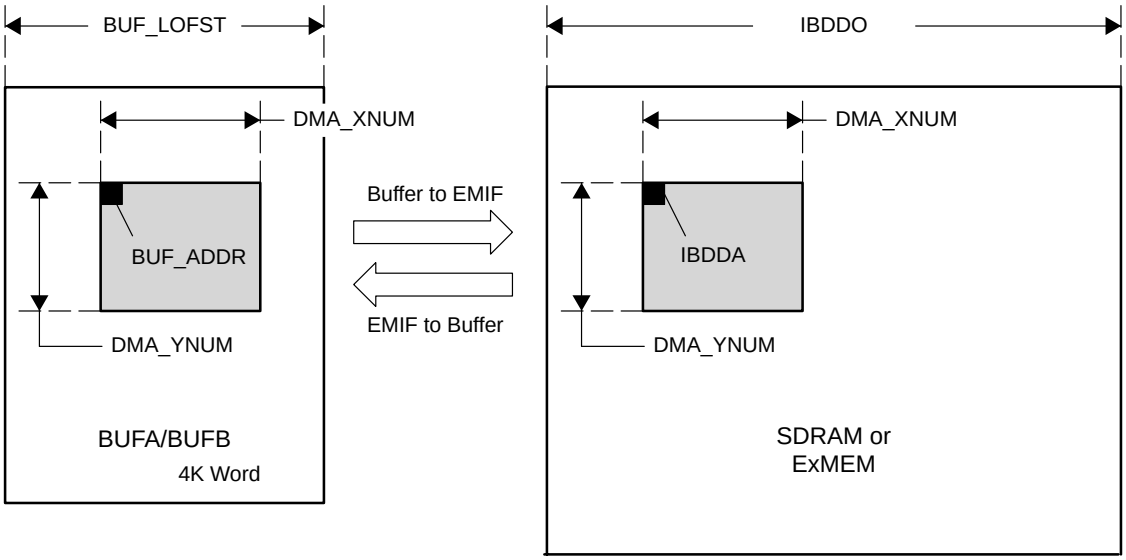


Figure 6–5. Second-Order Data Transfer

The following registers are used to set up and control DMA transfers between the DSP image buffers and EMIF. These registers are located in the DSP I/O space (see Table 6–14).

Table 6–15. IB–DMA Destination Address High (IBDDA_HI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	ADDR25	ADDR24	ADDR23	ADDR22	ADDR21	ADDR20	ADDR19	ADDR18	ADDR17	ADDR16
R/W	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

Table 6–16. IB–DMA Destination Address High (IBDDA_HI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9–0	ADDR24–ADDR16	Bits [25:16] of EMIF relative address

Table 6–17. IB–DMA Destination Address Low (IBDDA_LO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ADDR15	ADDR14	ADDR13	ADDR12	ADDR11	ADDR10	ADDR09	ADDR08	ADDR07	ADDR06	ADDR05	ADDR04	ADDR03	ADDR02	ADDR01	ADDR00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–18. IB–DMA Destination Address Low (IBDDA_LO) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	ADDR15–ADDR0	Bits [15:0] of EMIF relative address (unit = word)

Table 6–19. IB–DMA Destination Offset (IBDDO) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	LOFST11	LOFST10	LOFST09	LOFST08	LOFST07	LOFST06	LOFST05	LOFST04	LOFST03	LOFST02	LOFST01	LOFST00
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–20. IB–DMA Destination Offset (IBDDO) Register

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	LOFST11–LOFST0	EMIF offset address (unit = word) This value corresponds to the number of words used to represent on line of the image. This register must be set to an even value.

Table 6–21. Image Buffer Address (BUF_ADDR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	ADDR11	ADDR10	ADDR09	ADDR08	ADDR07	ADDR06	ADDR05	ADDR04	ADDR03	ADDR02	ADDR01	ADDR00
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–22. Image Buffer Address (BUF_ADDR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	ADDR11–ADDR0	Starting address of image buffer (unit = word) This defines an offset (0 to 4095) into an image buffer. The desired image buffer (A or B) should be switched to IB–DMA controller before initiating an external memory transfer, (See the IMG_IMGCTL register).

Table 6–23. Image Buffer Offset (BUF_LOFST) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	LOFST11	LOFST10	LOFST09	LOFST08	LOFST07	LOFST06	LOFST05	LOFST04	LOFST03	LOFST02	LOFST01	LOFST00
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–24. Image Buffer Offset (BUF_LOFST) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0	LOFST11–LOFST0	EMIF offset address (unit = word) This value (0–4095) corresponds to the number of pixels per line.

Table 6–25. Number of DMA X Bursts (DMA_XNUM) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	X12	X11	X10	X09	X08	X07	X06	X05	X04	X03	X02	X01	X00
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–26. Number of DMA X Bursts (DMA_XNUM) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–13	RSV	Reserved bits. Not used.
12–0	X12–X0	Number of X-direction bursts (unit = word)

Table 6–27. Number of DMA Y Bursts (DMA_YNUM) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	Y12	Y11	Y10	Y09	Y08	Y07	Y06	Y05	Y04	Y03	Y02	Y01	Y00
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–28. Number of DMA Y Bursts (DMA_YNUM) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–13	RSV	Reserved bits. Not used.
12–0	Y12–Y0	Number of X-direction bursts (unit = word)

Table 6–29. Image Buffer DMA Control (DMA_CTRL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	DDS	RSV	RSV	RSV	RSV	RSV	IRSV	REQ	HRW
R/W	—	—	—	—	—	—	—	R/W	—	—	—	—	—	—	R/W	R/W
Default	—	—	—	—	—	—	—	0	—	—	—	—	—	—	0	0

Table 6–30. Image Buffer DMA Control (DMA_CTRL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	DDS	DMA destination select 0 : SDRAM 1 : Asynchronous external memory interface When AMIF is chosen, bits DDST2:0 of CCDCDSPDEST select the CS region used.
7–2	RSV	Reserved bits. Not used.
1	REQ	DMA request 1:start request This bit is cleared automatically when DMA transfer is done.
0	RW	Read or write SDAM/AMIF 0: Buffer to SDRAM/AMIF 1: SDRAM/AMIF to buffer

Table 6–31. Image Buffer Mode Control (IMG_MODE) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BSWR	BSWW	RSV	IRSV	SHIFT1	SHIFT0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	—	—	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	—	—	0	0

Table 6–32. Image Buffer Mode Control (IMG_MODE) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5	BSWR	Byte swap on AMIF/SDRAM read 0: No swap 1: Byte swap
4	BSWW	Byte swap on AMIF/SDRAM write 0: No swap 1: Byte swap
3–2	RSV	Reserved bits. Not used.
1–0	SHIFT	Data bus shift down 00: 0 bit 01: 4 bits 10: 6 bits 11: 8 bits

6.5 Imaging Extension Coprocessor

The imaging extension coprocessor (iMX) is a parallel multiply and accumulate (MAC) engine for expanding the DSP image processing performance. It has a flexible control, a memory interface, and four MACs. The iMX is effective for fast computation of image processing in regular block units.

As a general-purpose image processor, the types of image processing given below are possible by the iMX.

- First-order/second-order filter
- CFA interpolation filter
- Color space conversion
- Color signal down sampling
- Edge intensification
- Color signal component suppression
- Discrete cosine transfer (DCT) and inverse discrete cosine transfer (IDCT)
- Table look-up
- …And many more…

6.5.1 Structure

The iMX has four MACs. Each MAC is constructed from the processing functions of multiplication, addition, subtraction and absolute value of subtraction, and an accumulator (ACC). Input data is read from image buffer BUFA or BUFB and coefficient memory (iMX COEF) and processed. To speed up calculations, it is possible to read four pieces of data and four coefficients continuously, and calculate and output simultaneously. The processes of memory read –> calculation –> output form a pipeline and processing is performed in one cycle. The calculator is constructed of 16 bits and the accumulator is constructed of 32 bits.

Output data is written to the image buffer BUFA or BUFB and coefficient memory (iMX COEF), and at this time, processes such as shift, rounding, overflow, and underflow are possible.

6.5.2 IMX Programming and Control

Variables for each process are put in the iMX command memory (iMX cmd) in the image buffer. One iMX command set is constructed of 7 to 19 words variable data, and by writing multiple command sets into command memory, multiple different processes can be executed in succession. It is possible to stop certain processes along the way and to stop between processes by inserting a 1-word SLEEP command. Since decoding of command codes is performed during calculation, the decoding time does not affect processing time. The iMX is hardware that enables speeding up of image processing in cases where a large quantity of data is processed at once.

The details of the internal architecture of the iMX coprocessor are not available to the user. To use this processing engine, you need to use the C callable API's provided by Texas Instruments. The description of the iMX given in this chapter is very limited. See the documentation provided with the library for more information on these APIs.

The following registers control the operation of the iMX coprocessor—start/stop execution, etc. These registers are located in DSP I/O space.

Table 6–33. iMX Control Registers Locations

OFFSET	ADDRESS	REGISTER NAME	
0x00	port10	IMX_START	iMX start
0x01	port11	IMX_INTR_EN	iMX interrupt enable
0x02	port12	IMX_BUSY	iMX status/busy
0x03	port13	IMX_CMDPTR	iMX command pointer
0x04	port14	IMX_ABORT	iMX abort

Table 6–34. iMX Start (IMX_START) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	A15	A14	A13	A12	A11	A10	A09	A08	A07	A06	A05	A04	A03	A02	A01	A00
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–35. iMX Start (IMX_START) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	A15–A0	Address in IMX command buffer Writing to this register starts the iMX execution at the specified address in the iMX command buffer. The command sequence must be terminated by an IMXCMD_SLEEP command, otherwise the iMX never returns.

Table 6–36. iMX Interrupt Enable (IMX_INTR_EN) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRSV	RSV	EN
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 6–37. iMX Interrupt Enable (IMX_INTR_EN) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	EN	Enables/disables the iMX interrupt (INT1) when command execution is complete. 0: Disable interrupt 1: Enable interrupt

Table 6–38. iMX Status/Busy (IMX_BUSY) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRSV	RSV	BUSY
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 6–39. iMX Status/Busy (IMX_BUSY) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	BUSY	Indicate whether iMX is executing an instruction sequence or is idle. 0: SLEEP/IDLE 1: BUSY

Table 6–40. iMX Abort (IMX_ABORT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRSV	RSV	abort
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 6–41. iMX Abort (IMX_ABORT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	ABORT	Abort iMX operation. When 1 is written into this bit, the iMX stops its execution. There is no need to clear this bit. After an abort, contents of image buffers are not assured.

Once the iMX has started command execution via the IMX_START register, the iMX runs until it encounters the sleep command in the command sequence. When iMX execution is aborted via the iMX_ABORT register, it is not possible to determine the execution status. Thus, for normal iMX operations, the IMX_ABORT should not be used. Instead, use the SLEEP command to properly finish iMX command execution.

When a sequence of command is encoded within the iMX command buffer, it is necessary to know where the next command can start. The IMX_CMDPTR register gives this information.

Table 6–42. iMX Command Pointer (IMX_CMDPTR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	PTR 15	PTR 14	PTR 13	PTR 12	PTR 11	PTR 10	PTR 09	PTR 08	PTR 07	PTR 06	PTR 05	PTR 04	PTR 03	PTR 02	PTR 01	PTR 00
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–43. iMX Command Pointer (IMX_CMDPTR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	PTR15–PTR0	Indicates first available location in the iMX command buffer. When iMX is executing, the value read is the start address of the current command.

6.6 Variable Length Coding Accelerator

The variable length coding (VLC) accelerator is a coprocessor that is optimized for quantization and Huffman encoding within the framework of JPEG compression and MPEG compression. This engine operates both the Huffman tables and the quantization matrices loaded in advance by the DSP. By pipelining in the design, high processing capabilities of 30,000,000 DCT coefficients per second for compression can be obtained.

6.6.1 Characteristics

The VLC has the following functions.

- Quantization, zigzag scanning, and Huffman encoding for JPEG encoding (baseline DCT, 8-bit sample), using a maximum of four quantization matrices (stored as invq[i,j]=2^15/q[i,j]), and two encoded Huffman tables, all of which are loadable. It can process one minimum coding unit (MCU) that contains a maximum of 10 blocks. Each block is constructed from 64 (= 8 x 8) samples.
- Quantization, zigzag scanning, and Huffman encoding for MPEG–1 video encoding. It can process one macroblock that contains six 8x8 blocks. The number of blocks and the number of brightness blocks within a block can be specified. Huffman encoding can be bypassed in order to generate the quantized level or zigzag order level.

The accelerator requires memory blocks for the input/output buffer, quantization matrices, and Huffman encoding tables. The memory structure can support ordinary encoding processes, JPEG MCU (minimum coding unit), and MPEG–1 macro blocks.

Input and output both use shared memory buffer (A or B) of 4K words (1 word = 16 bits). For MCU or macro blocks, it can process a maximum of ten 8x8 blocks.

There are 384 words required for each JPEG Huffman encoding table. JPEG standards recognize two tables; therefore, a total of 768 memory words are occupied.

The quantization matrix memory is 512 words, taking into consideration the coexistence of eight quantization matrices that occupy 64 x 16 bits each. JPEG anticipates four matrices and two matrices are required in MPEG encoding.

6.6.2 Structure

Figure 6–6 shows the VLC module structure.

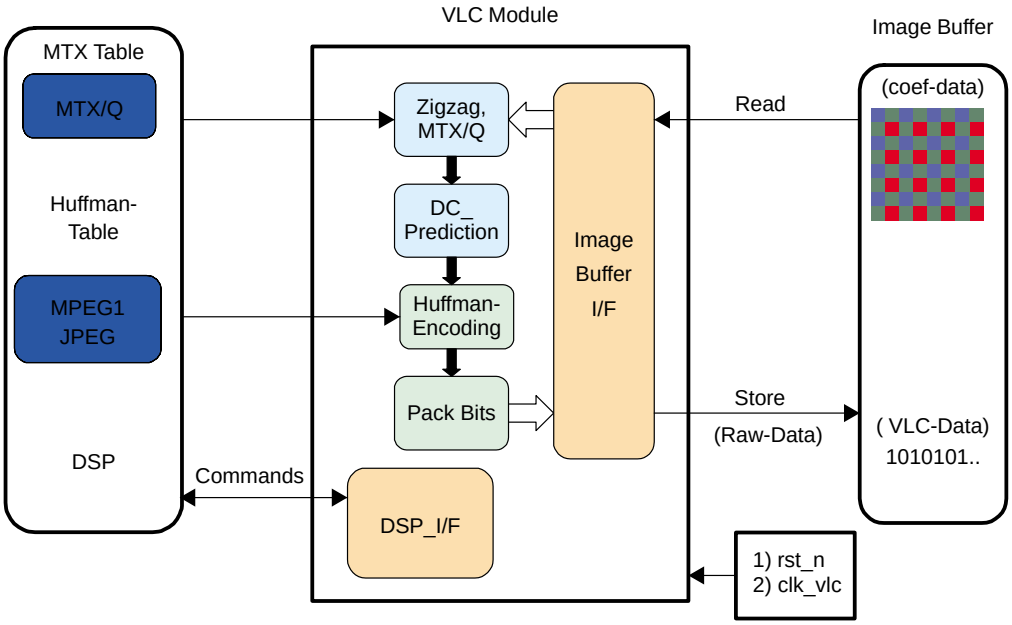


Figure 6–6. VLC Structure

6.6.3 Registers

These registers control the operation of the VLC coprocessor—start/stop execution, etc. These registers are located in the DSP I/O space.

Table 6–44. VLC Control Registers Locations

OFFSET	ADDRESS	REGISTER NAME	
0x00	port20	VLC_JPGENC	VLC JPEG encode
0x02	port22	VLC_MPGENC	VLC MPEG encode
0x04	port24	VLC_INTR	VLC interrupt enable/status
0x05	port25	VLC_DCTCF_ADDR	VLC DCT coefficient address
0x06	port26	VLC_BITS_WPTR	VLC bit stream word pointer
0x07	port27	VLC_BITS_BPTR	VLC bit stream bit pointer
0x08	port28	VLC_BITS_WORD_H	VLC bit stream insert word high
0x09	port29	VLC_BITS_WORD_L	VLC bit stream insert word low
0x0A	port2A	VLC_DC_PRED0	VLC DC predictor 0
0x0B	port2B	VLC_DC_PRED1	VLC DC predictor 1
0x0C	port2C	VLC_DC_PRED2	VLC DC predictor 2
0x0D	port2D	VLC_DC_PRED3	VLC DC predictor 3
0x0E	port2E	VLC_JPG_HUFFB0	VLC JPEG Huffman buffer 0
0x0F	port2F	VLC_JPG_HUFFB1	VLC JPEG Huffman buffer 1
0x10	port30	VLC_MPGENC_CBP	VLC MPEG encoder coded block pattern
0x11	port31	VLC_MPGENC_INVQ	VLC MPEG encoder inverse quantization
0x13	port33	VLC_MPG_NLUMABLKS	VLC MPEG number of Luma blocks
0x14	port34	VLC_JPG_ZRL_CODE_TBL0	VLC JPEG zero run length symbol: Code – Table 0
0x15	port35	VLC_JPG_ZRL_LEN_TBL0	VLC JPEG zero run length symbol: Length – Table 0
0x16	port36	VLC_JPG_ZRL_CODE_TBL1	VLC JPEG zero run length symbol: Code – Table 1

Table 6–44. VLC Control Registers Locations (Continued)

OFFSET	ADDRESS	REGISTER NAME	
0x17	port37	VLC_JPG_ZRL_LEN_TBL1	VLC JPEG zero run length symbol: Length – Table 1
0x18	port38	VLC_CONFIG0	VLC configuration for block 0
0x19	port39	VLC_CONFIG1	VLC configuration for block 1
0x1A	port3A	VLC_CONFIG2	VLC configuration for block 2
0x1B	port3B	VLC_CONFIG3	VLC configuration for block 3
0x1C	port3C	VLC_CONFIG4	VLC configuration for block 4
0x1D	port3D	VLC_CONFIG5	VLC configuration for block 5
0x1E	port3E	VLC_CONFIG6	VLC configuration for block 6
0x1F	port3F	VLC_CONFIG7	VLC configuration for block 7
0x20	port40	VLC_CONFIG8	VLC configuration for block 8
0x21	port41	VLC_CONFIG9	VLC configuration for block 9

6.6.4 VLC Control and Configuration

Before starting the VLC accelerator, the ZRL tables (length and code) must be filled. The user must also load the VLC configuration register with values to indicate which Huffman table, quantization matrix, and dc predictor they want to use for each block to be coded. The VLC_CONFIG0 to VLC_CONFIG9 registers let the user specify this.

Table 6–45. VLC Configuration (VLC_CONFIGx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	HTS	QMS2	QMS1	QMS0	DCS1	DCS0
R/W	—	—	—	—	—	—	—	—	—	—	W	W	W	W	W	W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 6–46. VLC Configuration (VLC_CONFIGx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5	HTS	Huffman table selector (1 of 2)
4–2	QMS2:0	Quantization matrix selector (1 of 8 banks)
3–0	DCS1:0	DC predictor selector (1 of 4)

When the VLC finishes processing the data, it generates an interrupt (int2) to the DSP. The interrupt is enabled or disabled with the following register. This register indicates if the VLC is currently processing data or not.

Table 6–47. VLC Interrupt Enable/Status (VLC_INTR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IRSV	RSV	INTR
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0

Table 6–48. VLC Interrupt Enable/Status (VLC_INTR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–1	RSV	Reserved bits. Not used.
0	INTR	When read, this bit correspond to the VLC status: 0: IDLE 1: BUSY When written, this bit controls the interrupt: 0: DISABLE 1: ENABLE

NOTE: Writing to this register also clears interrupt.

When the Q tables and Huffman tables have been loaded within the shared memory blocks, start the VLC engine by writing to either one of the following two registers.

- VLC_JPGENC to start a JPEG compression
- VLC_MPGENC to start a MPEG compression

Writing to either one of these registers starts the VLC (for JPEG or MPEG encode). Writing to this register also clears interrupt.

These registers allow the user to specify the number of blocks to encode.

When the MPEG mode is selected the Huffman compression can be bypassed and only the zigzag and the quantization are processed.

Table 6–49. VLC JPEG Encode (VLC_JPGENC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	BP0	RSV	RSV	RSV	RSV	NBE3	NBE2	NBE1	NBE0
R/W	—	—	—	—	—	—	—	R/W	—	—	—	—	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	0	—	—	—	—	0	0	0	0

Table 6–50. VLC JPEG Encode (VLC_JPGENC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–9	RSV	Reserved bits. Not used.
8	BP0	Bypass 00 When set to 1, bypasses inserting 00 byte after FF byte (normal = 0)
7–4	RSV	Reserved bits. Not used.
3–0	NBE3–NEB0	Number of 8x8 blocks to encode (1..10)

Table 6–51. VLC MPEG Encode (VLC_MPGENC) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	IMB	HBP	RSV	RSV	RSV	RSV	NBE3	NBE2	NBE1	NBE0
R/W	—	—	—	—	—	—	R/W	R/W	—	—	—	—	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	0	0	—	—	—	—	0	0	0	0

Table 6–52. VLC MPEG Encode (VLC_MPGENC) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–10	RSV	Reserved bits. Not used.
9	IMB	Intra-Macro Block When set to 1, specifies that the MPEG block is an intro-macro block
8	HBP	Huffman bypass When set to 1, bypasses Huffman (quantization and zigzag only)
7–4	RSV	Reserved bits. Not used.
3–0	NBE3–NBE0	Number of blocks Number of 8x8 blocks to encode (1..6)

When it starts, the VLC engine reads the data pointed by the VLC_DCTCF_ADDR register.

Table 6–53. VLC DCT Coefficient Address (VLC_DCTCF_ADDR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	DCA10	DCA9	DCA8	DCA7	DCA6	DCA5	DCA4	DCA3	DCA2	DCA1	DCA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–54. VLC DCT Coefficient Address (VLC_DCTCF_ADDR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	dca10–dca0	Input DCT Coefficient starting address (in 16-bit words)

6.6.5 Quantization and Zigzag

In JPEG or MPEG, each element F[u,v] of the 8x8 block is divided by a constant q(u,v). The table of q(u,v) is the quantization table. The JPEG and MPEG standards has two default quantization tables: one for the luminance information and one for the chrominance information. A total of four sets of tables can be used. After power on, the content of the shared memory block VLC Q is empty. Even if users decides to use the standard Q table, they need to use the DSP to initialize this memory block.

The 512 words of this block can fit eight Q tables of sixty-four 16-bit elements each.

Table 6–55. VLC Q Tables Addresses

Q TABLE	START ADDRESS	END ADDRESS
0	0xB600h	0xB63Fh
1	0xB640h	0xB67Fh
2	0xB680h	0xB6BFh
3	0xB6C0h	0xB6FFh
4	0xB700h	0xB73Fh
5	0xB740h	0xB77Fh
6	0xB780h	0xB6BFh
7	0xB7C0h	0xB7FFh

The elements need to be stored in a consecutive order.

For MPEG compression, the following register gives the inverse quantization scale.

Table 6–56. VLC MPEG Encoder Inverse Quantization (VLC_MPGENC_INVQ) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	IQ15	IQ14	IQ13	IQ12	IQ11	IQ10	IQ9	IQ8	IQ7	IIQ6	IQ5	IQ4	IQ3	IQ2	IQ0	IQ0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–57. VLC MPEG Encoder Inverse Quantization (VLC_MPGENC_INVQ) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	IQ15–IQ0	Inverse quantization: 2^15/quant_scale for MPEG quantization

A zigzag scan is then run over the result of the quantization. Figure 6–7 shows the scheme used.

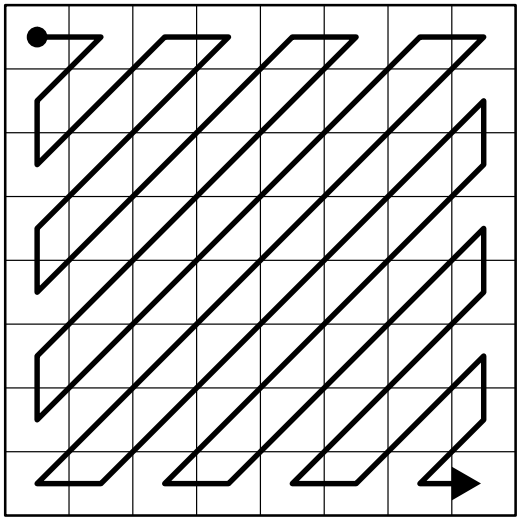


Figure 6–7. Zigzag Scan Scheme

The VLC accelerator of the DSC24 does not support the alternate-scan scheme.

6.6.6 DPCM on DC Component

In image compression, the dc component is large and varied, but often close to the previous value. The VLC engine calculates the difference between the current dc value (upper left corner of the 8x8 block) and the previous dc value.

The previous value is read from the VLC_DC_PREDx register and the current value is written back. Four different values can be used for the difference pulse code modulation on the dc value. Each one is stored in a different register. The 10 VLC configuration registers let the user choose which one they want to use.

Table 6–58. VLC DC Predictor (VLC_DC_PREDx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	DC11	DC10	DC09	DC08	DC07	DC06	DC05	DC04	DC03	DC02	DC01	DC00
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–59. VLC DC Predictor (VLC_DC_PREDx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–12	RSV	Reserved bits. Not used.
11–0		Initial/final dc predictors

6.6.7 Huffman Encoding

A Huffman code is designed by merging together the two least probable characters and repeating this process until there is only one character remaining. A code tree is thus generated and the Huffman code is obtained from the labeling of the code tree.

The Huffman table is not calculated by the VLC engine but is provided by the DSP. The VLC Huffman memory block is not initialized after power on, so your must initialize the tables with the proper values.

Two different tables can be used. The VLC configuration registers select which one has to be used for each block. The base address for the Huffman table is 0xB800h, the offset is defined by the VLC_JPG_HUFFB0 and VLC_JPG_HUFFB1 registers.

Table 6–60. VLC JPEG Huffman Buffer (VLC_JPG_HUFFBx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	HTA10	HTA09	HTA08	HTA07	HTA06	HTA05	HTA04	HTA03	HTA02	HTA01	HTA00
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 6–61. VLC JPEG Huffman Buffer (VLC_JPG_HUFFBx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0		Huffman tables address: Offset address for JPEG Huffman tables, should be even (32-bit aligned). Points to first entry of ac table, run=0, size=0. DC table is offset by 384 words.

Huffman table entries are stored as interleaving code/length 16-bit words. The code words are in even addresses and length in odd addresses.

Table 6–62. Huffman Buffer Memory Organization

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Content	Code Size								Huffman Code							

For example, if the Huffman code is 111111111110, at address 0, then 0x0FFE (code) and 0x000C (length) is stored at address 1.

6.6.8 Zero Run-Length on AC Components

In JPEG, the ac coefficients are converted to a combination (zero run length, amplitude).

Two sets of registers are used to specify which ZRL symbol is used for each Huffman table. One of the registers specifies the code (VLC_JPG_ZRL_CODE_TBLx) and the other one gives the length of it (VLC_JPG_ZRL_LEN_TBLx).

The tables used must be filled prior to VLC computation. (HW does not read Huffman table for the symbol found in these registers.)

Table 6–63. VLC JPEG Zero Run Length Symbol Code (VLC_JPG_ZRL_TBLx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	ZRLC15	ZRLC14	ZRLC13	ZRLC12	ZRLC11	ZRLC10	ZRLC9	ZRLC8	ZRLC7	ZRLC6	ZRLC5	ZRLC4	ZRLC3	ZRLC2	ZRLC1	ZRLC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–64. VLC JPEG Zero Run Length Symbol Code (VLC_JPG_ZRL_TBLx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	ZRL15–ZRL0	ZRL Code: JPEG ZRL symbol, right aligned, for associated table (0 or 1)

Table 6–65. VLC JPEG Zero Run Length Symbol Length (VLC_JPG_ZRL_LEN_TBLx) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	ZRLL4	ZRLL3	ZRLL2	ZRLL1	ZRLL0
R/W	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0

Table 6–66. VLC JPEG Zero Run Length Symbol Length (VLC_JPG_ZRL_LEN_TBLx) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–5	RSV	Reserved bits. Not used.
4–0	ZRL15–ZRL0	ZRL length: JPEG ZRL symbol length for associated table (0 or 1)

6.6.9 Bit Packing

The result of the Huffman encoding is stored in a consecutive manner. The bits are packed to reduce even more of the space needed to store the data. Because the VLC does not always have enough bits of information to complete a word after the compression of each 8x8 block, it needs to save some information that is reused to know where to start writing the result of the next compression. This information is stored in the following four registers.

- VLC_BITS_WORD_H is loaded with the high order part of the current 32-bit word.
- VLC_BITS_WORD_L is loaded with the low order part of the current 32-bit word.
- VLC_BITS_WPTR points to the insert point. The address is given in 16-bit words.
- VLC_BITS_BPTR tells the system the number of empty bits in the last 32-bit word.

Table 6–67. VLC Bit Stream Word Pointer (VLC_BITS_WPTR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	BIPA10	BIPA9	BIPA8	BIPA7	BIPA6	BIPA5	BIPA4	BIPA3	BIPA2	BIPA1	BIPA0
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

Table 6–68. VLC Bit Stream Word Pointer (VLC_BITS_WPTR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–11	RSV	Reserved bits. Not used.
10–0	BIPA10–BIPA0	Bit stream insert point's word address: 16-bit address, but must be even (32-bit aligned)

Table 6–69. VLC Bit Stream Bit Pointer (VLC_BITS_BPTR) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	EB5	EB4	EB3	EB2	EB1	EB0
R/W	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 6–70. VLC Bit Stream Bit Pointer (VLC_BITS_BPTR) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	EB5–EB0	Empty Bits: Number of empty bits in the 32-bit word pointed by VLC_BITS_WPTR. Value = 1..32

Table 6–71. VLC Bit Stream Insert Word-High (VLC_BITS_WORD_H) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LW15	LW14	LW13	LW12	LW11	LW10	LW9	LW8	LW7	LW6	LW5	LW4	LW3	LW2	LW1	LW0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–72. VLC Bit Stream Insert Word-High (VLC_BITS_WORD_H) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	LW31–LW16	Last word: High-order half of partially filled 32-bit word at VLC_BITS_WPTR. DSP should fill this word upon input and write this word back to memory upon output (if VLC_BITS_BPTR != 32)

Table 6–73. VLC Bit Stream Insert Word-Low (VLC_BITS_WORD_L) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	LW31	LW30	LW29	LW28	LW27	LW26	LW25	LW24	LW23	LW22	LW21	LW20	LW19	LW18	LW17	LW16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Default	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6–74. VLC Bit Stream Insert Word-Low (VLC_BITS_WORD_L) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–0	LW31–LW16	Last word: High-order half of partially filled 32-bit word at VLC_BITS_WPTR. DSP should fill this word upon input and write this word back to memory upon output (if VLC_BITS_BPTR ! = 32)

6.6.10 MPEG Blocks

When the VLC is used for MPEG compression, access to the following two registers is allowed. The VLC_MPG_NLUMABLKS register needs to be configured prior to launching MPEG compression.

Table 6–75. VLC MPEG Encoder Coded Block Pattern (VLC_MPGENC_CBP) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	CBP5	CBP4	CBP3	CBP2	CBP1	CBP0
R/W	—	—	—	—	—	—	—	—	—	—	R	R	R	R	R	R
Default	—	—	—	—	—	—	—	—	—	—	0	0	0	0	0	0

Table 6–76. VLC MPEG Encoder Coded Block Pattern (VLC_MPGENC_CBP) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–6	RSV	Reserved bits. Not used.
5–0	CBP5–CPB0	Coded block pattern: When bit is 1 it indicates at least one nonzero in a block. Bit 0 is for the first block, bit 1 for the second block, and so on.

Table 6–77. VLC MPEG Number of Luma Blocks (VLC_MPG_NLUMABLKS) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	NLB2	NLB1	NLB0
R/W	—	—	—	—	—	—	—	—	—	—	—	—	—	R/W	R/W	R/W
Default	—	—	—	—	—	—	—	—	—	—	—	—	—	0	0	0

Table 6–78. VLC MPEG Number of Luma Blocks (VLC_MPG_NLUMABLKS) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved bits. Not used.
2–0	NLB2–NLB0	Number of luma blocks

6.7 DSP Idle Modes

The C54x DSP embedded in the DSC24 has power-down modes in which it enters a dormant state and dissipates less power than normal operation while maintaining the CPU contents. This allows operations to continue unaltered when the power-down mode is terminated.

The user invokes one of the power-down modes either by executing the IDLE1 IDLE2 instructions, or by using the MCU DSP controller to drive the HOLD signal low with the HM status bit set to 1.

Power-down operation is summarized in Table 6–79 and described in detail in the following subsections.

Table 6–79. Operation During the Three Power-Down Modes

OPERATION/FEATURE	IDLE1	IDLE2	HOLD
CPU halted	Yes	Yes	Yes†
CPU clock stopped	Yes	Yes	No
Peripheral clock stopped	No	Yes	No
POWER-DOWN TERMINATED BY:			
HOLD driven high	No	No	Yes
Unmasked internal hardware interrupts	Yes	No	No
Unmasked interrupts from MCU/iMX/VLC/DMA	Yes	Yes	No
RS signal controlled by MCU DSP controller module	Yes	Yes	No

† Depending on the state of the HM bit, the CPU continues to execute unless the execution requires an external memory access.

6.7.1 IDLE1 Mode

The IDLE1 mode halts all CPU activities except the system clock. Because the system clock remains applied to the peripheral modules, the peripheral circuits continue operating and the CLKOUT pin remains active. Thus, peripherals such as McBSP and timers can take the CPU out of its power-down state.

Use the IDLE1 instruction to enter the IDLE1 mode. To terminate IDLE1, use a wake-up interrupt. If INTM = 0 when the wake-up interrupt takes place, the C54x DSP enters the ISR when IDLE1 is terminated. If INTM = 1, the C54x DSP continues with the instruction following the IDLE1 instruction. All wake-up interrupts must be set to enable the corresponding bits in the IMR register regardless of the INTM value. The only exceptions are the nonmaskable interrupts, RS and NMI.

6.7.2 IDLE2 Mode

The IDLE2 mode halts the on-chip peripherals as well as the CPU. Because the on-chip peripherals are stopped in this mode, they cannot be used to generate the interrupt to wake up the DSP as with IDLE1. However, power is significantly reduced because the device is completely stopped.

Use the IDLE2 instruction to enter the IDLE2 mode. To terminate IDLE2, activate any of the external interrupt signals (RS, NMI, and INT0) with a 10-ns minimum pulse. If INTM = 0 when the wake-up interrupt takes place, the C54x DSP enters the ISR when IDLE2 is terminated. If INTM = 1, the C54x DSP continues with the instruction following the IDLE2 instruction. All wake-up interrupts must be set to enable the corresponding bits in the IMR register regardless of the INTM value. Reset all peripherals when IDLE2 terminates, especially the McBSP when the CLKS pin is used.

When RS is the wake-up interrupt in IDLE2, a 10-ns minimum pulse of RS activates the reset sequence.

6.7.3 Hold Mode

The hold mode is another power-down mode. Depending on the value of the HM bit, this mode can be used to halt the CPU.

This power-down mode is initiated by the HOLD signal. The MCU with its DSP controller module controls this signal. The effect of HOLD depends on the value of HM. If HM = 1, the CPU stops executing. If HM = 0, the CPU continues to execute internally. The user can use HM = 0 with the HOLD signal when their system does not require expanded memory accesses or the use of iMX or VLC coprocessors. The C54x DSP continues to operate normally unless an off-chip access is required by an instruction; then the processor halts until HOLD is released.

This mode does not stop the operation of on-chip peripherals (such as timers and serial ports); they continue to operate regardless of the HOLD level or the condition of the HM bit.

This mode is terminated when HOLD becomes inactive.

6.7.4 Other Power-Down Capability

Switching the internal CLKOUT signal can reduce the DSC24 DSP core power consumption. This internal signal is coming out of the DSP core but it is not used within the DSC24 chip.

The CLKOUT off function allows the C54x DSP to disable CLKOUT using software instructions. The CLKOFF bit of PMST determines whether CLKOUT is enabled or disabled. At reset, CLKOUT is enabled. To gain a little bit on power consumption disable this signal.

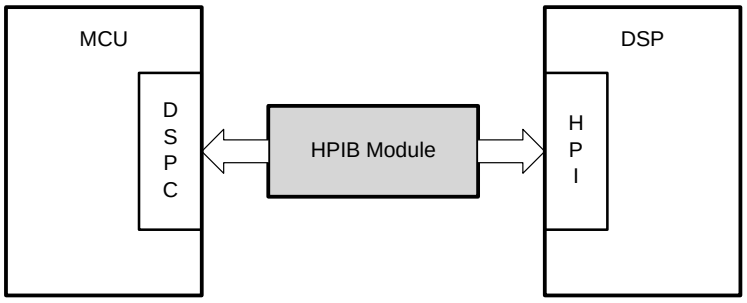
7 MCU-DSP Communication

This chapter describes the hardware interface between the ARM subsystem and the DSP subsystem.

7.1 Introduction

The MCU's DSP controller module controls data transfers and interrupts between the host and the DSP, via the host port interface (HPI). The DSC24 has the Texas Instruments TMS320VC5409 equivalent DSP built in. This DSP has an HPI as a communication port with a host CPU like the ARM7 core used in the DSC24.

The MCU accesses the DSP memory by programming the HPIB, opening a 32K-word window into the DSP memory map. The map contains the data structures shared by the MCU and the DSP for command requests, acknowledgements, and datagrams.



DSPC: DSP Controller
HPIB: HPI Bridge
HPI: Host Port Interface

Figure 7–1. HPIB Module

Unlike the C5409 DSP, the DSC24 HPI can only be used in a 16-bit mode. The HPI related registers for the MCU side and the DSP side are described below. The handshaking method is also explained.

7.2 Registers

On the MCU side, the DSP controller module uses two registers:

Table 7–1. DSP Controller Registers (MCU Side)

Offset	Address	Register Name	
0x00	0x0003:0700	HPIBCTL	HPI bridge control
0x01	0x0003:0702	HPIBSTAT	HPI bridge status

These registers determine how the MCU and the DSP communicate. Options include DSP hold and DSP wake up interrupts.

On the DSP side, only one register is used. This register is shared between the HPI interface and the DSP clock controller.

Table 7–2. HPI Controller Registers (DSP Side)

Address	Register Name	
Port 0xFFFF	CCAI	Clock controller/ARM interrupt

These registers are described with greater details later on in this chapter.

7.3 Memory Access

The DSP internal memory (DARAM block) can be read and written by the MCU through the HPI bridge. Figure 7–2 shows the DSP memory block that the MCU (or an external CPU) can access.

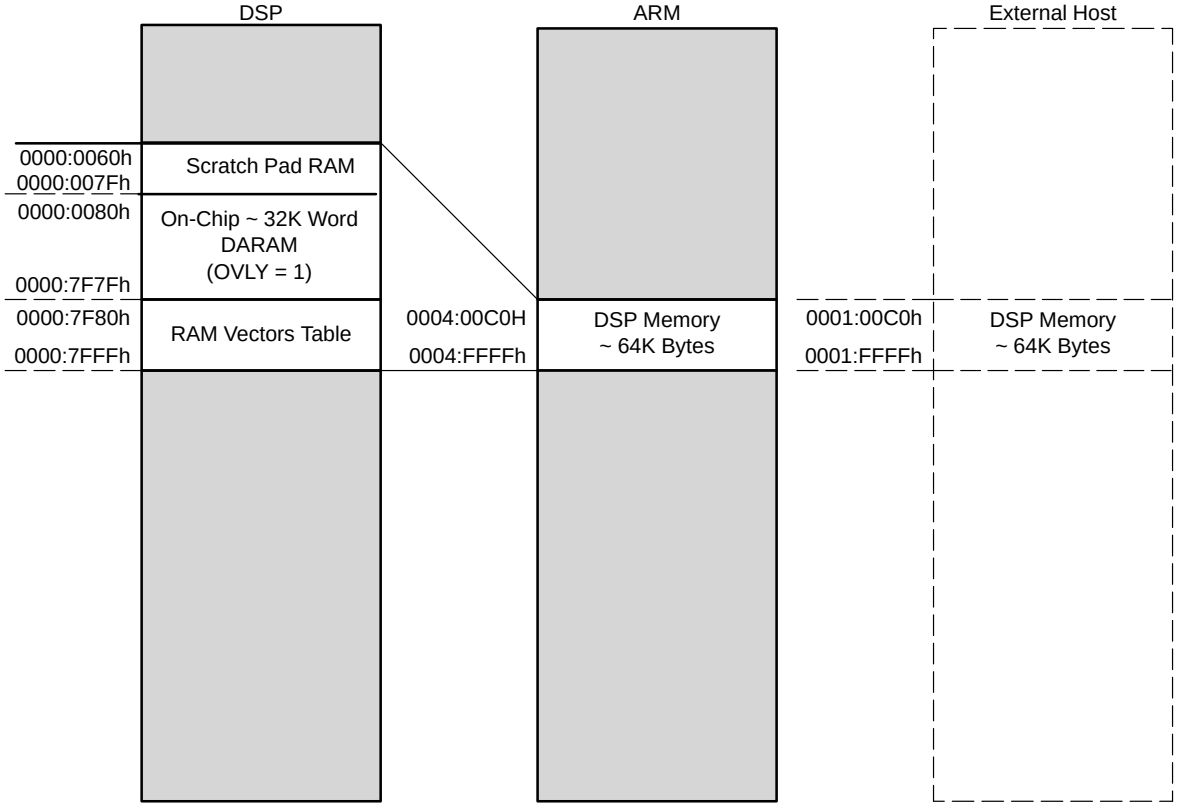


Figure 7–2. HPIB Memory Window

The DSP internal dual access memory read/write and mutual interrupts can be performed between the MCU and the DSP. When the MCU and DSP access collide, a wait is automatically inserted on the DSP side.

The HPI16 mode is an access method that uses 16-bit addresses and a 16-bit data line and is used in transmission/reception of data or downloading of DSP programs. When seen from the MCU, the DSP DARAM addresses 0x0080–0x7FFF (0x0–0x7F is reserved, 16-bit address) become the MCU addresses 0004:0100–0004: FFFF (8-bit address) and are treated as shared memory.

7.4 HPI Control and Status Registers

The registers HPIBCTL and HPIBSTAT mentioned before are used to control the HPI bridge as well as generate interrupts to the DSP. These registers are described below and the HPIBSTAT register is a read only register.

Table 7–3. HPI Bridge Control (HPIBCTL) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	TST	TST	RSV	DBIO	DHOLD	DRST	DINT0	RSV	TST	RSV	HPNMI	RSV	RSV	HPIEN
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	—	—	R/W	R/W	—	—	R/W
Default	—	—	1	1	—	1	1	1	1	—	1	1	1	—	—	1

Table 7–4. HPI Bridge Control (HPIBCTL) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–14	RSV	Reserved bits. Not used.
13–12	TST	Test bits – Set these bits to 1
11	RSV	Reserved bit. Not used.
10	DBIO	DSP BIO Connected to BIO signal of DSP. When the DSP is in the BIO status, it can branch and control the program.
9	DHOLD	DSP hold request Holds DSP. Connected to HOLDN signal of the DSP 0: DSP is kept on hold 1: Normal mode of operation
8	DRST	DSP reset This bit is connected to the reset signal of the DSP. The user must write successively a 0 then 1 in this bit to reset the DSP. 0: DSP reset signal held low 1: Reset signal not activated
7	DINT0	DSP interrupt 0: Signal input into INT0 port of the DSP. Initial value is 1. INT0 is generated in the DSP by the host setting this bit to 0. After interrupt is generated, the MCU must return this bit to 1. 0: INT0 1: Normal
6–5	RSV	Reserved bits. Not used.
4	TST	Test bit – Set this bit to 1.
3	HPNMI	HPI NMI. The MCU generates a non-maskable interrupt to the DSP. This bit is connected to the NMI signal of the DSP. After interrupt is generated, the MCU must return this bit to 1. 0: NMI 1: Normal
2–1	RSV	Reserved bits. Not used.
0	HPIEN	Enables HPI. Connected to the HPIEN signal of the DSP. When disabled, all HPI functions are invalid. 0: Disabled 1: Enabled

For proper operation of the HPIB interface, a 1 needs to be written to bits 4, 12, and 13 of the HPIBCTL register.

After an interrupt has been issued from the MCU to the DSP, it is very important to write a 1 back into bits 3, 7, and 8 to let the DSP run fine. If this does not happen, the DSP keeps getting an interrupt or stays in a reset mode.

Table 7–5. HPI Bridge Status (HPIBSTAT) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	DXF	RSV	RSV	RSV	HOLDA	RSV	RSV	RSV	RSV	RSV	RSV	RSV	HINT
R/W	—	—	—	R	—	—	—	R	—	—	—		—	—	—	R
Default	—	—	1	1	—	—	—	1	—	—	—	—	—	—	—	1

Table 7–6. HPI Bridge Status (HPIBSTAT) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–13	RSV	Reserved bits. Not used.
12	DXF	DSP XF indicator Shows status of the DSP XF signal output
11–9	RSV	Reserved bits. Not used.
8	HOLDA	DSP hold acknowledge Indicates status of the DSP HOLDA signal output. After a hold request is sent to the DSP by the DHOLD bit of the HPIBCLT register, this bit checks the acknowledgement. 0: DSP hold status 1: Normal
7–1	RSV	Reserved bits. Not used.
0	HINT	Host port interrupt Indicates status of the DSP HINT signal output 0: DSP sends interrupt request to host 1: Normal

7.5 HPI Interrupts and Handshaking

The host and the ’54x interrupts each other using bits in the HPIC and CCAI registers. The following section explains this process.

A ’54x interrupt is generated when the host (MCU) writes a 1 to the DINT0 or HPNMI bits in the HPIBCTL register. This interrupt can be used to wake up the ’54x from IDLE. After a 1 is written to DINT0 or HPNMI by the host, it is necessary for a 0 to be written to avoid another interrupt to be generated.

On the ’54x, the ARM-to-’54x interrupt vector address is either xx04h for NMI, either xx40h for INT0. Because the ’54x interrupt vectors of the DSC24 is remapped into the on-chip RAM, the host instructs the ’54x to execute preprogrammed functions by initializing the NMI or INT0 interrupt vector. When the ’54x interrupts are remapped to on-chip RAM, the host can:

- Write the opcode for a branch instruction at address xx04h or xx40h, and
- Write the start address of a function at address xx05h or xx41h in the interrupt vector table prior to interrupting the ’54x. When using this technique, care must be taken to prevent the host from corrupting the other interrupt vectors.

7.5.1 ’54x Using HINT to Interrupt the Host Device

On standard ‘C5409 DSP’s, an interrupt, called HINT, is available in the host port interface when configured for an 8-bit transfer mode (HPI–8). Because this mode is not available on the DSC24, this interrupt has been remapped to bit 0 of the CCAI register in the I/O space.

Table 7–7. Clock Controller/ARM Interrupt (CCAI) Register

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Name	RSV	RSV	RSV	DXF	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	RSV	IMBUF	IMXVLC	HINT
R/W	—	—	—	R	—	—	—	—	—	—	—		—	R/W	R/W	R/W
Default	—	—	1	1	—	—	—	—	—	—	—	—	—	0	0	0

Table 7–8. Clock Controller/ARM Interrupt (CCAI) Register Bit/Field Descriptions

BIT	REGISTER NAME	DESCRIPTION
15–3	RSV	Reserved bits. Not used.
2	IMBUF	Image buffer clock: 0 : Off 1 : On
1	IMXVLC	IMX / VLC clock 0 : Off 1 : On
0	HINT	HPI interrupt Interrupt to MCU: 0 : Off 1 : On

Bits 1 and 2 concern the DSP subsystem clock controller. These bits are described with more detail in the DSP subsystem chapter.

When the ’54x writes a 1 to the HINT bit in CCAI, the HINT output is driven low, and the HINT bit is read as a 1 by the MCU. Consequently, the HINT signal can be used as an active-low interrupt source for the MCU. It is necessary to set this bit back to 0 to avoid any additional interrupt. The following example code shows how to generate a pulse through HINT. In the example, HINT is originally 0, then set to 1, and then set back to 0, thus generating a short pulse.

In the following example, bits 1 and 2 are considered to be 0.

```
LD     #0x1,B        //Load 1 into B
STL    B,*AR2        //Store B into *AR2
PORTW  *AR2,0xFFFF   //Write *AR2 (*AR2=1) to I/O port address 0xFFFF
LD     #0,B          //Load 0 into B
STL    B,*AR2        //Store B into *AR2
RPT    #7            //Cause delay by repeating the previous instruction 7
                     //times
NOP                  //No operation
PORTW  *AR2,0xFFFF   //Write AR2 (*AR2=0) to I/O port address 0xFFFF
```

On the MCU side, HINT in HPIBSTAT (0x01) shows the current state of the I/O port at 0xFFFF. When HINT is 0, the I/O port is outputting a high signal (1), which triggers INT11. When HINT is 1, the I/O port is outputting a low signal (0).

7.6 Bootloader

7.6.1 ARM Boot Sequence

Upon power up and reset, the ARM program counter is reset to 0x000, which resides at the start of the ARM internal RAM. The ARM reset vector is hard coded in ROM at this location with an instruction to branch to 0x0100:0000 (start of the CS0 address space). The MCU should then execute a boot sequence from the CS0 region to start the DSC24 system.

The ARM boot sequence should initialize the different peripheral register as those for the clock controller module or the SDRAM interface.

Also, the ARM boot phase should reset the DSP via the DSP controller register bit. When done, the MCUtakes the DSP out of reset. At this stage, the DSP begins execution at 0xFF80, a hard coded address in ROM, which places the DSP into IDLE 1 mode (a power-down mode). This gives the MCU the opportunity to download the DSP code by using the HPI-16 interface. After it completes downloading the DSP image, the MCU can send an interrupt to the DSP, which wakes it up from IDLE1 mode and jumps to address 0x7F80 where it starts running its application code loaded by the MCU.

7.6.2 DSP Boot Sequence

On the ‘C54x DSP, the reset vector is located at address 0x0FF80. Within the DSC24, this memory location has been ROMed with a small boot loader. Figure 7–3 describes this process. It is achieved in the following six steps.

1. The DSP goes out of reset and loads its PC register program counter with 0x0FF80.
2. The ROM code found at this location branch the DSP to address 0x0F800 where the initialization routine resides.
3. At address 0x0F800, the DSP status register PMST is initialized to move the vector table to 0x07F80, the interrupts are disabled except for INT0, and the DSP is sent to a power-down mode (IDLE1).
4. While in this mode, the MCU loads the DSP internal 32K words of DARAM memory with the DSP code. There is no code predefined on the MCU side, see the DSC24 library and application note for more information.
5. When the MCU is finished downloading the DSP code, it wakes up from the IDLE mode by using the INT0.
6. The DSP then branch to address 0x7F80 where the new interrupt vector table is located. The MCU should have loaded this location with at least a branch to the start of your code.

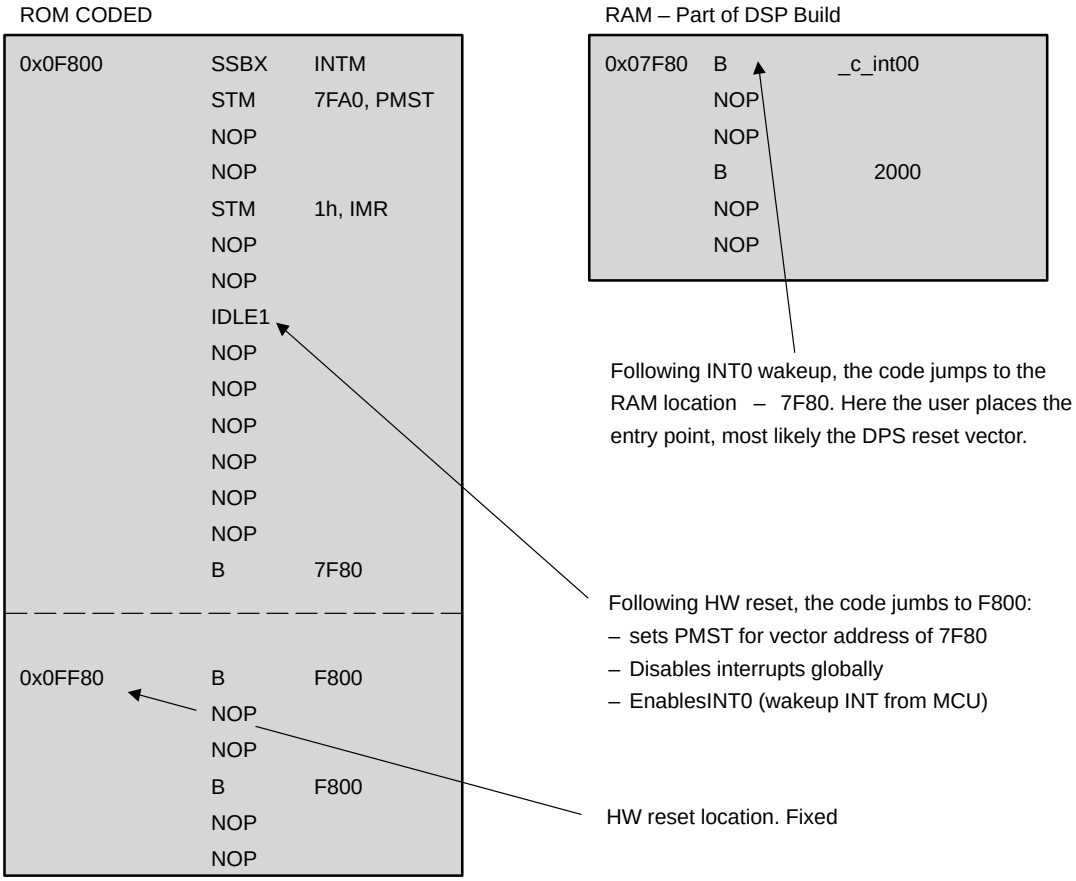


Figure 7–3. DSP Boot Load Process

A Block Diagram

This appendix shows the main block diagrams of the DSC24 chip. For more information on each item, see the appropriate chapter.

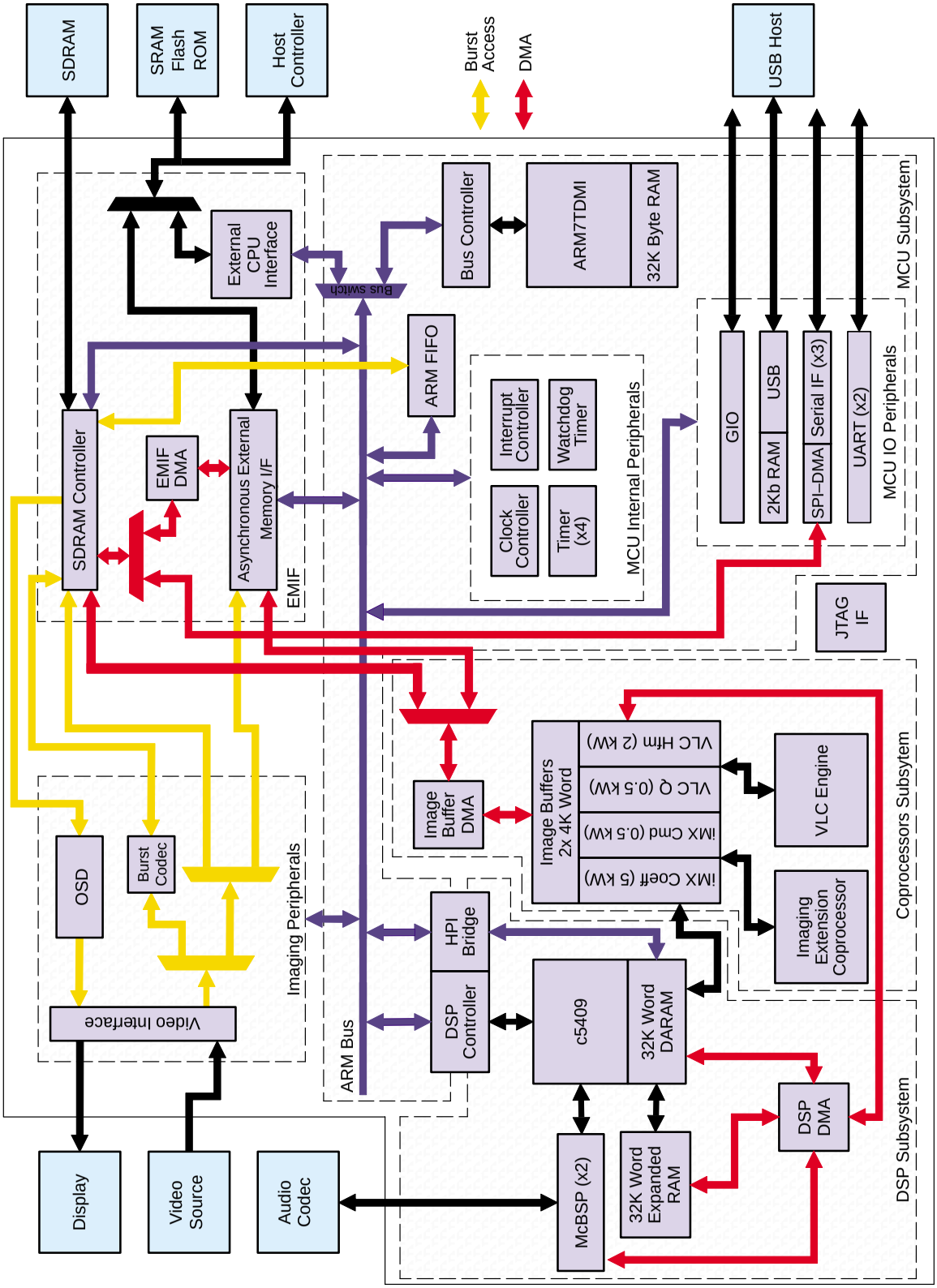


Figure A–1. Functional Block Diagram

B Memory Maps

This appendix regroups the various memory maps available in this document.

Table B–1. MCU Memory Map

AREA NAME	START ADDRESS	END ADDRESS	SIZE	BUS WIDTH	NUMBER OF WAIT STATES
Reset vector ROM	0x0000:0000	0x0000:0003	4 bytes	32 bits	0
ARM internal RAM	0x0000:0004	0x0000:7FFF	~32 Kb	8, 16, 32 bits	0
ARM peripheral	0x0003:0000	0x0003:0FFF	4 Kb	16 bits	0†
DSP memory	0x0004:00C0	0x0004:FFFF	~64 Kb	16 bits	1 minimum
DAEM	0x0010:0000	0x080F:FFFF	128 Mb	8,16, 32 bits	1 minimum

† The number of cycles for USB register access varies depending on the internal state and several cycles are required.

Table B–2. Peripheral Registers Memory Map

PERIPHERAL AREA NAME	START ADDRESS	END ADDRESS
Timer 0	0x0003:0000	0x0003:007F
Timer 1	0x0003:0080	0x0003:00FF
Timer 2	0x0003:0100	0x0003:017F
Timer 3	0x0003:0180	0x0003:01FF
Serial 0	0x0003:0200	0x0003:027F
Serial 1	0x0003:0280	0x0003:02FF
Serial 2	0x0003:0300	0x0003:037F
UART 0	0x0003:0380	0x0003:03FF
UART 1	0x0003:0400	0x0003:047F
*USB	0x0003:0480	0x0003:04FF
WDT	0x0003:0500	0x0003:057F
Interrupt controller	0x0003:0580	0x0003:05FF
GIO	0x0003:0600	0x0003:067F
OSD	0x0003:0680	0x0003:06FF
DSP controller	0x0003:0700	0x0003:077F
Video interface	0x0003:0780	0x0003:07FF
Burst mode compression	0x0003:0800	0x0003:087F
Clock controller	0x0003:0880	0x0003:08FF
Bus controller	0x0003:0900	0x0003:097F
SDRAM controller	0x0003:0980	0x0003:09FF
External memory I/F	0x0003:0A00	0x0003:0A7F

Table B–3. Memory Map of MCU External Memory After Reset

VARIABLE REGION NAME	START ADDRESS	END ADDRESS	SIZE
External memory region 0	0010:0000	008F:FFFF	8M bytes
SDRAM region	0090:0000	020F:FFFF	24M bytes
External memory region 1	0210:0000	038F:FFFF	24M bytes
External memory region 2	0390:0000	050F:FFFF	24M bytes
External memory region 3	0510:0000	068F:FFFF	24M bytes
External memory region 4	0690:0000	080F:FFFF	24M bytes

Table B–4. Memory Map in External CPU Mode (as seen from the external CPU)

AREA NAME	START ADDRESS	END ADDRESS	SIZE
MCU peripheral	0x00:0000	0x00:0FFF	4 Kb
DSP memory	0x01:0000	0x01:FFFF	~64 Kb
SDRAM	0x02:0000	0xFF:FFFF	16 Mb

Table B–5. DSP Program Memory Map

AREA NAME	START ADDRESS	END ADDRESS	SIZE	NOTE
Dual access RAM	0x00:0080	0x00:7F7F	~32K words	(1)
RAM vector Table	0x00:7F80	0x00:7FFF	128 words	
ROM	0x00:C000	0x00:FE7F	~16K words	
ROM vector Table	0x00:FF80	0x00:FFFF	128 words	
Dual access RAM	0x01:0080	0x01:7F7F	~32K words	(1)
Single access RAM	0x01:8000	0x01:BFFF	16K words	
Single access RAM	0x01:C000	0x01:FFFF	16K words	(2)

NOTES: 1. Block of memory shared with other areas noted (1)
2. Block of memory shared with other area noted (2)

Table B–6. DSP Data Memory Map

AREA NAME	START ADDRESS	END ADDRESS	SIZE	NOTE
Memory map register	0x0000	0x005F	96 words	
Scratch pad RAM	0x0060	0x007F	32 words	
Dual access RAM	0x0080	0x7F7F	~32K words	(1)
Image buffer A	0x8000	0x8FFF	4K words	
Image buffer B	0x9000	0x9FFF	4K words	
IMX coefficient buffer	0xA000	0xB3FF	5K words	(3)
IMX command buffer	0xB400	0xB5FF	512 words	
VLC Q tables	0xB600	0xB7FF	512 words	
VLC Huffman tables	0xB800	0xBFFF	2K words	
Single access RAM	0xC000	0xFFFF	~16K words	(2)

NOTES: 1. Block of memory shared with other areas noted (1)
2. Block of memory shared with other area noted (2)
3. Only 12 bits wide

Table B–7. DSP IO Memory Map

AREA NAME	START ADDRESS	END ADDRESS	SIZE
Image buffer control registers	0x0000	0x0009	16 words
iMX control registers	0x0010	0x0014	5 words
VLC control registers	0x0020	0x0037	24 words
Clock controller / ARM interrupt register	0xFFFF	0xFFFF	1 Word

C Glossary

A

AAC: *Advanced audio coding*. Standard used to compress audio data.

absolute address: An address that is permanently assigned to a memory location.

ABU: A*utobuffering unit*

ABUC: *ABU control register.* A register that controls the operation of the autobuffering unit.

AC97: *Audio codec '97*. A standard which uses a dual-phase frame feature with the first phase consisting of a single 16-bit word, and the second phase consisting of twelve 20-bit words.

accumulator: A register that stores the results of an operation and provides an input for subsequent arithmetic logic unit (ALU) operations.

accumulator shift mode field (ASM): A 5-bit field in status register 1 (ST1) that specifies a shift value (from –16 to 15) used to shift an accumulator value when executing certain instructions, such as instructions with parallel loads and stores.

ADC: *Analog-to-digital converter/conversion*. A converter with an internal sample-and-hold circuitry used to translate an analog signal to a digital signal. Also used to describe the process of conversion.

adder: A unit that adds or subtracts two numbers.

address: The location of a word in memory.

address bus: A group of connections used to route addresses. The C54x CPU has four 16-bit address busses: CAB, DAB, EAB, and PAB. The ARM7 has one 32-bit address bus.

addressing mode: The method by which an instruction calculates the location of an object in memory.

AG: *Accumulator guard bits.* An 8-bit register that contains bits 39–32 (the guard bits) of accumulator A.

AIC: *Analog interface chip.* An integrated circuit that performs serial analog-to-digital (A/D) and digital-to-analog (D/A) conversions.

alignment: A process in which the linker places an output section at an address that falls on an n-bit boundary, where n is a power of 2. You can specify alignment with the SECTIONS linker directive.

ALU: Arithmetic logic unit. The part of the CPU that performs arithmetic and logic operations.

AMIF: See asynchronous external memory interface

AR: *Automatic reload*

AR0–AR7: *Auxiliary registers 0–7.* Eight 16-bit registers that can be accessed by the CPU and modified by the auxiliary register arithmetic units (ARAUs) and are used primarily for data memory addressing.

archiver: A software program that allows the user to collect several individual files into a single file called an archive library. The archiver also allows you to delete, extract, or replace members of the archive library, as well as to add new members.

ARM: *Advanced RISC machine*

ARM Clock: Internal clock signal used to drive the ARM core and various peripherals

ARM7TDMI: Specific version of the ARM core. This version supports the thumb mode (T), it has embedded the Ice (I) debugger (D) module and includes a multiplier (M).

ARMIO: *Advanced RISC machine input/output*. See *GIO.*

ARR, ARR0, ARR1: *ABU address receive register*. A 16-bit register that specifies the destination address at which the autobuffering unit begins storing received data.

ASCII: *American Standard Code for Information Exchange*. A standard computer code for representing and exchanging alphanumeric information.

assembler: A software program that creates a machine-language program from a source file that contains assembly language instructions, directives, and macro directives. The assembler substitutes absolute operation codes for symbolic operation codes, and absolute or relocatable addresses for symbolic addresses.

ASSP: *Application Specific Silicon Product or Application Specific Standard Product*

Asynchronous External Memory InterFace: Control block to interface asynchronous external memory like RAM, ROM, flash memory or other devices

AUTOINIT: DMA auto-initialization bit. This bit configures the channel to automatically initialize after a frame by loading from the global DMA registers. Located in the DMMCR register.

auxiliary register arithmetic unit: An unsigned, 16-bit arithmetic logic unit (ALU) used to calculate indirectaddresses using auxiliary registers.

auxiliary register pointer (ARP): A 3-bit field in status register 0 (ST0) used as a pointer to the currently selected auxiliary register, when the C54x core is operating in 'C5x/'C2xx compatibility mode.

auxiliary registers: Eight 16-bit registers (AR7–AR0) that are used as pointers to an address within data space. These registers are operated on by the auxiliary register arithmetic units (ARAUs) and are selected by the auxiliary register pointer (ARP). See also auxiliary register arithmetic unit.

B

bank-switching control register (BSCR): A 16-bit register that defines the external memory bank size and enables or disables automatic insertion of extra cycles when accesses cross memory bank boundaries.

barrel shifter: A unit that rotates bits in a word.

BDRR, BDRR0, BDRR1: BSP data receive register. Two 16-bit registers used to receive data through the buffered serial ports. BDRR0 corresponds to buffered serial port 0; BDRR1 corresponds to buffered serial port 1.

BG: *accumulator B guard bits*. An 8-bit register that contains bits 39–32 (the guard bits) of accumulator B.

BGA: *Ball grid array*

big endian: An addressing protocol in which bytes are numbered from left to right within a word. More significant bytes in a word have lower numbered addresses. Endian ordering is hardware-specific and is determined at reset. See also little endian.

BIO: A general purpose, branch-control, input pin that can be used to monitor the status of peripheral devices.

BIOS: *Built-in operating system*

BIST: *Built-in self-test*

Bit Ordering: A feature that allows the LSB to be transferred to the serial port first when companded data is not used.

Blending: An OSD functionality used to mix the color of the bitmap information and the color of the video data together.

block: A group of interconnected cells. May contain instances of other blocks.

boot: The process of loading a program into program memory.

boot loader: A built-in segment of code that transfers code from an external source to program memory at power up.

BSP: *Buffered serial port.* An on-chip module that consists of a full-duplex, double-buffered serial port interface and an autobuffering unit (ABU). The double-buffered serial port of the BSP is an enhanced version of the standard serial port interface. The double-buffered serial port allows transfer of a continuous communication stream (8-,10-,12-, or 16-bit data packets).

Burst Codec: An internal module used to compress the incoming video data on the fly in order to optimized the use of the SDRAM in a multi-shot mode.

burst mode: A synchronous serial port mode in which a single word is transmitted following a frame synchronization pulse (FSX and FSR).

Bus controller: Internal arm bus controller

BUSC: See *bus controller*

BXSR: *BSP data transmit shift register*. A 16-bit register that holds serial data to be transmitted from the BDX pin.

C

C: See *carry bit*

C compiler: A program that translates C source statements into assembly language source statements.

carry bit (C): A bit in status register 0 (ST0) used by the ALU in extended arithmetic operations and accumulator shifts and rotates. The carry bit can be tested by conditional instructions.

CCD clock: Internal clock signal used to drive the video interface module.

CIF: *Common intermediate format.* A video format used in video conferencing systems that easily support both NTSC and PAL signals. CIF is part of the ITU H.261 video conferencing standard. It specifies a data rate of 30 frames per second (fps), with each frame containing 288 lines and 352 pixels per line.

CLK: *Clock*

clk_arm: See *ARM clock*

clk_ccd: See *CCD clock*

clk_dsp: See *DSP clock*

clk_osd: See *OSD clock*

clk_sdr: See *SDRAM clock*

clk_usb: See *USB clock*

clk_xi: See *XI clock*

CLKC: See *clock controller*

CLKR: *Receive clock*. A McBSP interface signal

CLKS: *External clock input*. A McBSP interface signal

CLKX: *Transmit clock*. A McBSP interface signal

Clock Controller: Internal module used to generate the various clocks needed by the chip.

CLUT: See *color look-up table*

code: A set of instructions written to perform a task; a computer program or part of a program.

codec: Coder decoder or compression decompression. A device that codes in one direction of transmission and decodes in another direction of transmission.

Coefficient buffer: Block of shared memory used to store iMX data.

Coefficient memory: See *coefficient buffer*

COFF: *Common object file format*

Color look-up table: Look-up table used to convert a color number into a YCbCr value.

Command buffer: Block on shared memory used to store the code to be executed by the iMX engine.

Command memory: See *command buffer*

compatibility mode (CMPT): A bit in status register 1 (ST1) that determines whether or not the auxiliary register pointer (ARP) is used to select an auxiliary register in single indirect addressing mode.

compiler mode (CPL): A bit in status register 1 (ST1) that determines whether the CPU uses the data page pointer or the stack pointer to generate data memory addresses in direct addressing mode.

conditional processing: A method of processing one block of source code or an alternate block of source code, according to the evaluation of a specified expression.

configured memory: Memory that the linker has specified for allocation.

constant: A numeric value that does not changed and can be used as an operand.

Coprocessors subsystem: Part of the chip that includes the iMX engine, the VLC engine, the image buffer-DMA and some blocks of shared memory

CPU: *Central processing unit.* The CPU is the portion of the processor involved in arithmetic, shifting, and Boolean logic operations, as well as the generation of data and program memory addresses. The CPU includes the central arithmetic logic unit (CALU), the multiplier, and the auxiliary register arithmetic unit (ARAU).

CS: *Chip select*

CTS: *Clear to send*

D

DAEM: *Dynamically allocated external memory*. MCU external memory region that includes the SDRAM area and five asynchronous memory spaces.

DARAM: *Dual access random access memory*. RAM that can be accessed twice in a single CPU clock cycle. For example, your code can read from and write to DARAM in the same clock cycle.

data memory: A memory region used for storing and manipulating data.

data page pointer (DP): A 9-bit field in DSP status register 0 (ST0) that specifies which of 512, 128 · 16 word pages is currently selected for direct address generation. DP provides the nine MSBs of the data-memory address; the dma provides the lower seven.

data ROM (DROM): A bit in processor mode status register (PMST) that determines whether or not part of the on-chip ROM is mapped into data space.

DCT: *Discrete cosine transform*. A fast Fourier transform used in manipulating compressed still and moving picture data.

device modes of operation: See *device-operating modes*

device operating modes: Specify the hardware configuration and functionality of the chip.

digital loopback mode: A synchronous serial port test mode in which the DLB bit connects the receive pins to the transmit pins on the same device to test if the port is operating correctly.

digital-to-analog (D/A) converter: Circuitry that translates a digital signal to an analog signal.

DIR: Direction bit. Selects the direction of a general-purpose I/O pin. Located in the GPIOCR register.

Direct memory access (DMA) controller: The direct memory access (DMA) controller transfers data between regions in the memory map without intervention by the CPU. The DMA allows movement to and from internal memory, internal peripherals, or external devices to occur in the background of CPU operation.

DLB: *Data loopback*. A synchronous serial port test mode in which the receive pins are connected internally to the transmit pins on the same device. This mode, enabled or disabled by the DLB bit, allows you to test whether the port is operating correctly.

DMA: See *direct memory access (controller)*

DP: See *data page pointer*

DPCM: *Differential pulse code modulation*. Based upon the previously encountered samples, DPCM predicts the value for the next sample and records the difference between this value and the value that really occurred.

DPLL: *Digital phase-locked loop*. Digital implementation of PLL

DROM: See *data ROM*

DSP: *Digital signal processor*. A semiconductor that manipulates discrete or discontinuous electrical impulses in a manner that implements a desired algorithm.

DSP Clock: Internal clock signal used to drive the ARM core and various peripherals

DSP subsystem: The DSP core, its memory (DARAM and SARAM) and peripherals.

DSP–DMA: DMA controller available within the DSP subsystem.

DSYN: DMA sync event control bits. Configures the DSP–DMA to synchronize to a particular event. Located in the DMSFC register.

E

EEPROM: *Electrically erasable programmable read only memory*

EMIF: See *external memory interface*

EMIF-DMA: DMA controller available within the EMIF subsystem

emulator: A hardware development system that emulates TMS470R1x and C54x operation.

entry point: The starting execution point in target memory.

EPROM: *Erasable programmable read-only memory*

EVM: *Evaluation module*

Expanded Memory: The blocks of memory available to the DSP through its XIO interface.

exponent encoder (EXP): A hardware device that computes the exponent value of the accumulator.

Extended Memory: The DSP program memory available when XPC is not zero (address 0x1000h and above)

Extended Software-Programmable Wait-State Generator: Extends external bus cycles to interface with slower off chip memory and I/O devices.

External CPU Interface: Interface used to virtually replace the internal ARM7 by an another device

external interrupt: A hardware interrupt triggered by a pin.

External Memory: Memory not implemented onto the chip

External Memory Interface: Subsystem that includes the asynchronous memory interface, the SDRAM controller, the EMIF–DAM, and the external CPU interface.

F

fast Fourier transform (FFT): An efficient method of computing the discrete Fourier transform, which transforms functions between the time domain and frequency domain. The time-to-frequency domain is called the forward transform, and the frequency-to-time domain is called the inverse transformation.

field: A software-configurable data type whose length can be programmed to be any value in the range of 1–32 bits.

FIFO: *First in first out*. A queue; a data structure or hardware buffer from which items are taken out in the same order they were put in. A FIFO is useful for buffering a stream of data between a sender and receiver, which are not synchronized; that is, the sender and receiver are not sending and receiving at exactly the same rate. If the rates differ by too much in one direction for too long, the FIFO becomes either full (blocking the sender) or empty (blocking the receiver).

FIQ: *Fast interrupt request*. See *ISR*.

FPGA: *Field programmable gate array*. A type of logic chip that can be programmed. An FPGA is similar to a PLD; whereas PLDs are generally limited to hundreds of gates, FPGAs support thousands of gates.

G

GIO: See *GPIO*

global symbol: A kind of symbol that is either 1) defined in the current module and accessed in another, or 2) accessed in the current module but defined in another.

GND: *Ground*

GPIO: *General-purpose input/output pin*. Pins that can be used to supply input signals from an external device or output signals to an external device. These pins are not linked to specific uses; rather, they provide input or output signals for a variety of purposes.

GPP: *General-purpose processor*

H

H/W: *Hardware*

Hardware cursor: Rectangular shape that can be displayed and moved around over the output video

hardware interrupt: An interrupt triggered through physical connections with on-chip peripherals or external devices.

high-level language debugging: The ability of a compiler to retain symbolic and high-level language information (such as type and function definitions) so that a debugging tool can use this information.

HINT: *C54x-to-host processor interrupt*. A bit in the HPI control register (HPIC) that enables/disables an interrupt from the C54x DSP to a host device.

HIO: *Host I/O*

HM: See *hold mode*

HMODE: *HPI mode*. Allows the HPI to operate in multiplexed/nonmultiplexed modes.

hold mode (HM): A bit in status register ST1 that determines whether the CPU enters the hold state in normal mode or concurrent mode.

hole: An area containing no actual code or data. This area is between the input sections that compose an output section.

HOM: *Host-only mode*. A mode that allows only the host to access host port interface (HPI) memory. The CPU has no access to the HPI memory block during HOM.

Host: External device that drives the HPI peripherals.

Host Port Interface (HPI-16): The DSP host port interface that provides a full 16-bit bidirectional data bus, which does not require byte identification. Memory accesses are synchronized with the direct memory access (DMA) controller providing access to the complete internal memory address range.

host-only mode (HOM): The mode that allows the host to access HPI memory while the C54x CPU is in IDLE2 (all internal clocks stopped) or in reset mode.

HPI: See *host port interface*

HSYNC: *Horizontal synchronization*. A bidirectional horizontal timing signal occurring once per line with a pulse width defined as an integral number of frame clock (FCLK) periods. Synchronization signals can be used to enable retrace of the electron beam of a display screen.

Huffman buffer: Block of memory used to store the Huffman tables needed by the VLC engine

Huffman coding: Form of encoding that creates the most efficient set of prefix codes for a given stream of data.

Huffman Table: Tables used to store the symbol—representation data used during Huffman coding

I

I/F: *Interface*

I/O*: Input/output*

I^{2}C: *Inter-integrated circuit*. A multimaster bus where multiple chips can be connected. Each chip can act as a master by initiating a data transfer.

I2S: *Inter-IC sound*. A digital audio interface standard

IB–DMA: *Image fuffer DMA controller*. The DMA cntroller that can be found within the coprocessors subsystem

IC: *Integrated circuit*

ICE: *In-circuit emulation*

ID: *Identifier data*

IDCT: *Inverse discrete cosine transform*. See *DCT*.

IDE: *Integrated development environment*. A programming environment integrated into an application.

Idle mode: Mode in which the CPU and its peripherals are in a low activity state. This mode is often used to save power.

IFR: See *interrupt flag register*

Image buffer: Block of shared memory used to transfer data between the external memory, the iMX, the VLC, and the DSP. Only one module can have access to an image buffer at a given time. The DSP controls the switching

imaging extension: DSP coprocessor designed and optimized for 2D-computation.

Imaging peripherals: Device subsystem including the various peripherals used to get image and vide data in and out of the chip.

IMR: See *interrupt mask register*

iMX: See *imaging extension*

INT: *Interrupt* . A signal sent by hardware or software to a processor requesting attention. An interrupt tells the processor to suspend its current operation, save the current task status, and perform a particular set of instructions. Interrupts communicate with the operating system and prioritize tasks to be performed.

INTC: *Interrupt controller*

Internal peripherals: ARM peripherals used internally (clock controller, interrupt controller, timers, and watchdog timer)

interrupt: A condition caused by internal hardware, an event external to the CPU, or by a previously executed instruction that forces the current program to be suspended and causes the processor to execute an interrupt service routine corresponding to the interrupt.

interrupt flag register: A 16-bit memory-mapped register that flags pending interrupts.

interrupt mask register: A 16-bit memory-mapped register that masks external and internal interrupts.

interrupt service routine (ISR): A module of code that is executed in response to a hardware or software interrupt.

Interrupt Vector Table: A table of interrupt vectors (pointers to routines that handle interrupts)

IPTR: *Interrupt vector pointer.* A 9-bit field in the DSP processor mode status register (PMST) that points to the 128-word page where interrupt vectors reside.

IR: *Instruction register*. A 16-bit register used to hold a fetched instruction.

IRQ: *(Low-priority) interrupt request*. IRQs are hardware lines over which devices can send interrupt signals to the microprocessor.

ISO: *Isochronous*. This refers to processes where data must be delivered within certain time constraints. For example, multimedia streams require an isochronous transport mechanism to ensure that data is delivered as fast as it is displayed and to ensure that the audio is synchronized with the video. Also, International Standards Organization.

ISR: *Interrupt service routine*. A function or set of functions that are called when an interrupt is encountered.

J

JPEG: *Joint Photographic Experts Group Standard*

JTAG*: Joint test action group*. The joint test action group was formed in 1985 to develop economical test methodologies for systems designed around complex integrated circuits and assembled with surface-mount technologies. The group drafted a standard that was subsequently adopted by IEEE as IEEE Standard 1149.1–1990, IEEE Standard Test Access Port and Boundary-Scan Architecture.

JTAG scan chain: Target hardware and CPUs JTAG headers connected together in series

L

LAN: *Local area network*

latency: The delay between when a condition occurs and when the device reacts to the condition. Also, in a pipeline, the necessary delay between the executions of two instructions to ensure that the values used by the second instruction are correct.

LCD*: Liquid crystal display*. A display that uses two sheets of polarizing material with a liquid crystal solution between them.

linker: A software tool that combines object files to form an object module that can be allocated into the CPU system memory and executed by the device.

little endian: An addressing protocol in which bytes are numbered from right to left within a word. More significant bytes in a word have higher numbered addresses. Endian ordering is hardware-specific and is determined at reset. See also big endian

LSB*: least significant bit*. The lowest order bit in a word.

M

macro: A user-defined routine that can be used as an instruction.

map file: An output file, created by the linker, that shows the memory configuration, section composition, and section allocation, as well as symbols and the addresses at which they were defined.

MB: *Megabyte*

MBit: *Megabit*

McBSP: See *Multi-Channel Buffered Serial Port*

MCU: *Microcontroller unit*. In this document, MCU refers to the ARM7 core

MCU subsystem: The ARM7 core and its internal and I/O peripherals

memory map: A map of target system memory space, which is partitioned into functional blocks.

memory-mapped register (MMR): The C54x CPU registers mapped into page 0 of the data memory space.

microcomputer mode: A mode in which the on-chip ROM is enabled and addressable for program accesses.

microprocessor mode: A mode in which the on-chip ROM is disabled for program accesses.

MIPS: *Million instructions per second*

mnemonic: An instruction name that the assembler translates into machine code.

modes of operation: See *device-operating modes*

MP3: *MPEG layer 3*. An audio compression format

MPEG: *Moving Pictures Expert Group*. A compression scheme for full motion video

MPEG1: The first MPEG compression scheme specification.

MPEG4: The most current MPEG compression scheme specification, intended for very narrow bandwidths.

MSB: *Most significant bit*. The highest order bit in a word. The plural form (MSBs) refers to a specified number of high-order bits, beginning with the highest order bit and counting to the right. For example, the eight MSBs of a 16-bit value are bits 15 through 8.

multichannel buffered serial port (McBSP): High-speed, full duplexed, buffered serial ports that allow direct interface to other C54x devices, codecs, and other devices in a system. The McBSPs provide full-duplex communication, multi-buffered data registers, independent framing and clocking for receive and transmit, and a flexible clock generator that can be programmed for internal or external shift clocking.

MUX: *Multiplex/multiplexer*

N

NC: *Not connected*

nested interrupt: A higher-priority interrupt that must be serviced before completion of the current interrupt service routine (ISR). An executing ISR can set the interrupt mask register (IMR) bits to prevent being suspended by another interrupt.

NMI: *Nonmaskable interrupt*. An interrupt that can be neither masked nor disabled

O

OE: *Output enable*

On-Screen Display: Module used to display bitmap information like logos, menu over the video output data.

operating modes: See *device-operating modes*

OS: *Operating system*

OSD: See *On-screen display*

OSD Clock: Internal clock signal used to drive the video interface and OSD module.

OVLY: Bit used to reconfigure the DSP memory map.

OVM: *Overflow mode bit*. A bit in status register 1 (ST1) that specifies how the ALU handles an overflow after an operation.

P

PCB: *Printed circuit board*

PCM: *Pulse code modulation*. A technique for digitizing speech by sampling the sound waves and converting each sample into a binary number.

PD: *Program data*

pipeline: A method of executing instructions in an assembly-line fashion.

PLL: *Phase-locked loop*. A closed loop frequency control system whose function is based on the phase-sensitive detection of the phase difference between the input signal and the output signal of the controlled oscillator (CO).

PMST: Processor mode status register. A DSP 16-bit status register that controls the memory configuration of the device.

PMT: *Parallel module test*. One of the test configuration modes of the device

program counter (PC): A register that indicates the location of the next instruction to be executed.

program counter extension register (XPC): A DSP register that contains the upper 7 bits of the current program memory address.

program memory: A memory region used for storing and executing programs.

Programmable Bank-Switching Module: Allows the ’54x to switch between external memory banks without requiring external wait states for memories that need several cycles to turn off. Bank-switching logic automatically inserts one cycle when accesses cross a 32K word memory-bank boundary inside program or data space.

PROM: *Programmable read-only memory*. A memory chip on which data can be written only once

Q

QCIF: *Quarter common intermediate format*. A video conferencing format that specifies data rates of 30 frames per second (fps), with each frame containing 144 lines and 176 pixels per line. This is one-fourth the resolution of CIF. QCIF support is required by the ITU H.261 video conferencing standard.

Quantization: Reduction of the number of bits used to represent a set of values.

Quantization table: 2D-matrix used during the quantization process

QVGA: *Quarter video graphics array*. One-fourth the resolution of VGA

R

RAM: *Random access memory*. A memory element that can be written to, as well as read

RAM overlay (OVLY): A bit in the processor mode status register (PMST) that determines whether or not on-chip RAM is mapped into the program space in addition to data space.

raw data: Executable code or initialized data in an output section.

register: A group of bits used for temporarily holding data or for controlling or specifying the status of a device.

relocation: A process in which the linker adjusts all the references to a symbol when the symbol’s address changes.

RESET: A means of bringing the CPU to a known state by setting the registers and control bits to predetermined values and signaling execution to start at a specified address.

RGB: Color representation using the Red Green Blue information.

RISC: *Reduced instruction set computer*. A computer whose instruction set and related decode mechanism are much simpler than those of microprogrammed complex instruction set computers. The result is a higher instruction throughput and a faster real-time interrupt service response from a smaller, cost-effective chip.

ROM: *Read only memory*. A semiconductor storage element containing permanent data that cannot be changed.

RTC: *Real-time clock*. A clock that keeps track of the time even when the device is turned off.

RTOS: *Real-time operating system*

S

S/W: *Software*

SAM: *Shared access mode*. The mode that allows both the DSP and the host to access host port interface (HPI) memory. In this mode, asynchronous host accesses are synchronized internally, and, in case of conflict, the host has access priority and the DSP waits one cycle.

SARAM: *Single access random access memory*. Memory space that only can be read from or written to in a single clock cycle; RAM that can accessed (read from or written to) once in a single CPU cycle.

SDRAM: *Synchronous DRAM*. Operates in sync with the CPU clock to avoid the delays caused by asynchronous operation

SDRAM clock: Internal clock signal to drive the SDRAM controller.

serial port interface: An on-chip full-duplex serial port interface that provides direct serial communication to serial devices with a minimum of external hardware, such as codecs and serial analog-to-digital (A/D) and digital-to-analog (D/A) converters. Status and control of the serial port is specified in the serial port control register (SPC).

shared memory blocks: blocks of memory with their access port shared between the DSP, the VLC, the iMX or the IB–DMA. The blocks of shared memory are image buffer A, image buffer B, iMX command buffer, iMX coefficient buffer, VLC Q tables, and VLC Huffman tables.

shifter: A hardware unit that shifts bits in a word to the left or to the right.

sign extension: An operation that fills the high order bits of a number with the sign bit.

sign-control logic: Circuitry used to extend data bits (signed/unsigned) to match the input data format of the multiplier, ALU, and shifter.

sign-extend: To fill the unused MSBs of a value with the value's sign bit.

software wait-state register (SWWSR): A 16-bit register that selects the number of wait states for the program, data, and I/O spaces of off-chip memory.

SP: See *stack pointer*.

SPI: *Serial port interface*. A signaling protocol for exchanging serial data

SPI-DMA: DMA controller available to the serial port interface

SRAM: *Static random access memory*. Fast memory that does not require refreshing, as DRAM does. It is more expensive than DRAM, though, and is not available in as high a density as DRAM.

stack: A block of memory used for storing return addresses for subroutines and interrupt service routines and for storing data.

stack pointer (SP): A register that always points to the last element pushed onto the stack.

Subsystem: Referred to as a group of design cells highly connected with each other.

symbol: A string of alphanumeric characters that represents an address or a value.

T

TI: *Texas instruments*

timer interrupt (TINT): A bit in the interrupt flag register (IFR) that indicates the timer counter register (TIM) has decremented past 0.

TINT: See *timer interrupt*

Transparency: OSD functionality that let a bitmap color to be selected as background information.

TRST*: Test reset*

U

UART: *Universal asynchronous receiver/transmitter.* Another name for the asynchronous serial port.

USB*: Universal serial bus*. An external bus standard that supports data transfer rates of 12M bps (12 million bits per second). A single USB port can be used to connect up to 127 peripheral devices.

USB Clock: Internal clock signal used to drive the USB interface module.

V

variable length coder: A method for the construction of minimum redundancy codes.

Video Loop Thru: Device video mode in which the signal available on the video output pins is identical to what arrives on the video input pins.

VLC: See *variable length coder*

VSYNC*: Vertical synchronization*. A bidirectional vertical timing signal occurring once per frame with a pulse-width defined as an integral number of lines (half lines for interlaced mode).

W

Watchdog timer: A timer that requires the user program or OS periodically write to the count register before the counter underflows.

WDT: See *watchdog timer*

WMA: *Windows media audio*

X

XF: A general purpose, software-controlled, DSP flag output signal

XI clock: Clock generated from external crystal input

XINT: Transmit interrupt to CPU

XIO: *External memory interface (input/output)* of the DSP core

Y

YCbCr: The color space used in the CCIR601 specification. Y is the luminance component and the Cr and Cb components are color difference signals. Cr and Cb are scaled versions of U and V in the YUV color space.

YUV: Color representation of a pixel information using the luminance and chrominance data