BLUE TREAK
Powering Tomorrow's Vision

LH79524/LH79525
User's Guide

2009 March 3

NXP founded by Philips

Specifications are subject to change without notice.

Suggested applications (if any) are for standard use; See *Important Restrictions* for limitations on special applications. See *Limited Warranty* for NXP's product warranty. The Limited Warranty is in lieu, and exclusive of, all other warranties, express or implied. ALL EXPRESS AND IMPLIED WARRANTIES, INCLUDING THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR USE AND FITNESS FOR A PARTICULAR PURPOSE, ARE SPECIFICALLY EXCLUDED. In no event will NXP be liable, or in any way responsible, for any incidental or consequential economic or property damage.

Purchase of I^2C components from NXP Corporation or one of its sublicensed Associated Companies conveys a license under the NXP I^2C Patent. Rights are granted to use these components in an I^2C system, provided that the system conforms to the I^2C Standard Specification as defined by NXP.

LH79524/LH79525 MCU User's Guide
Produced by NXP Semiconductors

Content Revisions

This document contains the following changes to content, causing it to differ from previous versions. Minor typographical changes, where they do not affect content, are not tracked here.

Record of Revisions

DATE	PAGE NO.	SECTION, TABLE, OR ILLUSTRATION	SUMMARY OF CHANGES
11-30-06	Throughout	—	Updated version number to Version 1.0.
	Throughout	—	“Preliminary” removed from all references.
	5-3	Section 5.1.1	Section added to clarify operation.
	7-16	Section 7.2.4.5	Corrected nWAIT description.
	7-32	Section 7.5.2.3	Corrected settings of bit 8.
	9-5	Section 9.1.5	Added section.
	13-3	Section 13.1.1	Section added to clarify reset of System and USB PLLs.
	13-9	Table 13-5	**“IMPORTANT”** note added.
	13-14	Section 13.2.2.4	Phrase added to clarify reset of System and USB PLLs.
	13-17	Section 13.2.2.7	Corrected equation.
	13-18	Section 13.2.2.8	Corrected equation.
	13-32	Section 13.2.2.21	Added text regarding changing from Standard Mode to FastBus Mode
	13-33	Table 13-53, Section 13.2.2.22	Definition of SYSPREDIV and equation corrected.
	13-34	Table 13-55, Section 13.2.2.23	Definition of USBPREDIV and equation corrected.
	14-8	Section 14.1.6.4	Section rewritten for clarity.
	14-13	Section 14.2.2.3	Clarified handling transactions less than, or greater than 16 bits.
	14-20	Section 14.2.2.10	Text added to clarify operation.
	15-3	Section 15.1.1	Rewrote section for clarity.
	16-27	Section 16.3.2.14	Text added to clarify operation.

Record of Revisions (Cont'd)

DATE	PAGE NO.	SECTION, TABLE, OR ILLUSTRATION	SUMMARY OF CHANGES
6-16-07	—	Throughout	Version updated to 1.1.
	5-4	Section 5.1.2.1	Section added to clarify DMA priority.
	4-18 – 4-19	Section 4.4.2	Enhanced text in Section 4.4.2.1 and Section 4.4.2.2, and added Section 4.4.2.3.
	4-21	Table 4-15	HSW clarified for AD-TFT panels.
	5-10	Section 5.2.2.4	Added second paragraph to advise users to correctly specify DMA parameters.
	9-3	Section 9.1.1	Added final paragraph describing effect of external pullup resistor to timing.
	9-7	Table 9-5	Bit descriptions enhanced for the SABT and STOP bits.
	13-34	Section 13.2.2.23	Reduced power suggestion when not using USB added; Power Down bit (bit 13) made RESERVED.
	14-8	Section 14.1.6.2	Corrected reference to SSE bit to bit position 1 in CTL1.
	17-4	Section 17.1.4.2	Added this section.
	17-21	Section 17.2.3.3	Text added to describe the STALL function for EP 0.
8-27-07	All	—	All references to Sharp replaced with NXP. Revision number rolled to Version 1.2.
3-04-09	7-48	Table 7-47	Corrected memory configuration.

Table of Contents

Chapter 3 – Boot Controller

Chapter 4 – Color Liquid Crystal Display Controller

Chapter 6 – Ethernet MAC Controller

Chapter 15 – Timers

Chapter 16 – UARTs

List of Figures

Chapter 7 – External Memory Controller

Chapter 9 – I^2C Module

Chapter 10 – I^2S Converter

Chapter 12 – Real Time Clock

Chapter 13 – Reset, Clock, and Power Controller

Chapter 14 – Synchronous Serial Port

Chapter 15 – Timers

Chapter 16 – UARTs

Chapter 17 – Universal Serial Bus Device

Chapter 19 – Watchdog Timer

List of Tables

Chapter 3 – Boot Controller

Chapter 4 – Color Liquid Crystal Display Controller

Chapter 5 – Direct Memory Access Controller

Chapter 6 – Ethernet MAC Controller

Chapter 7 – External Memory Controller

Chapter 8 – General Purpose Input/Output

Chapter 9 – I²C Module

Chapter 10 – I²S Converter

Chapter 11 – I/O Configuration

Chapter 12 – Real Time Clock

Chapter 13 – Reset, Clock, and Power Controller

Chapter 14 – Synchronous Serial Port

Chapter 15 – Timers

Chapter 16 – UARTs

Chapter 17 – Universal Serial Bus Device

Chapter 18 – Vectored Interrupt Controller

Chapter 19 – Watchdog Timer

Preface

The LH79524 and LH79525 are fully-integrated 16/32-bit MCUs based on a 32-bit ARM720T core. This User's Guide is the principal technical reference for these devices. This document assumes the reader is familiar with ARM720T programming. For more information on programming the ARM720T core, see the library of methods and downloads available from ARM Ltd., at http://www.nxp.com/redirect/arm.com.

For an abridged version of this User's Guide, consult the LH79524/LH79525 Data Sheet and the single page Product Brief. For details, contact a NXP representative or see the NXP Semiconductors website (http://www.nxp.com).

Application Notes and further information on connecting, programming and implementing the LH79524/LH79525, along with suggestions for companion parts, can be found on NXP's website (http://www.nxp.com).

IMPORTANT: The following sections contain important design information about the LH79524/LH79525. Please take a moment to read the 'Conventions and Terms' section in its entirety.

Conventions and Terms

For information on specific terms and acronyms see the Glossary in this User's Guide.

Unconnected (Floating) Inputs

Many applications employing the LH79524/LH79525 require extremely low standby and operating current consumption, especially in battery operated devices. To achieve minimum current, unused inputs must never be left floating (unconnected). Each input must be pulled up or pulled down with a 33 kΩ resistor (or smaller). In addition to terminating input pins, this also allows the designer to specify the reset state of input pins by selecting pull up (logical 1 at reset) or pull down (logical 0 at reset) resistors.

Multiplexed Pins

The LH79524 is manufactured in a CABGA package with 208 pins. The LH79525 is manufactured in a LQFP package with 176 pins. Some pins have only one function, but others are multiplexed and may carry as many as three functions. Designers must be aware that multiplexed pins cannot simultaneously support more than one function; a choice is required prior to designing the MCU into an application.

Pin Names

Package pins are named to indicate the signal(s) or functionality available at the pin. If the signal or function is active LOW, the name is prefixed with a lower-case 'n', such as nSCS2. Multiplexed pins are named to indicate all available functions, such as Pin D11 (Pin 139 for the LH79525): PE1/LCDDCLK, which can function as either GPIO Port E bit 1, or LCD Data Clock.

These naming conventions help designers recognize and avoid conflicts between multiplexed functions but can complicate explanatory text, so this User's Guide uses the name appropriate to the context. A discussion about Port E bit 1 would use PE1, for example, but information about LCD data would refer to signal LCDVD5. Readers must be aware that these are separate signals, with distinctly different functionality, which happen to be available on the same pin, although never simultaneously.

Peripheral Devices

The LH79524/LH79525 is an MCU built using the ARM720T RISC core as a base. Objects within the chip but external to the core processor and its support devices are referred to throughout this User's Guide as 'blocks' or 'Peripheral Devices'.

The LH79524/LH79525 includes two buses: an Advanced High-Performance Bus (AHB) and an Advanced Peripheral Bus (APB). The devices shown on the APB in the block diagrams are an example of Peripheral Devices in this document. Devices that are external to the chip are referred to as 'External Devices'.

Register Addresses

The LH79524/LH79525 is a memory-mapped device with programmable, internal registers that control its operation. Each internal register is located at a unique address in the memory map and the registers are generally grouped in the map by subsystem.

In this User's Guide, the addresses for all registers are expressed as a base address and an offset from that base. The base address indicates where in the map a group of registers begins and the offset locates a particular register, relative to its base address. Thus, any register's absolute address is the sum of its base address and its offset. Programmers will find this base+offset representation convenient for creating software structures to access the registers. The absolute addresses are also provided for convenient reference.

Register Tables

All Registers are presented in tabular format. A primary table presents each register's name, address, permissions, bit-field names and the register's contents at reset. Subsequent tables detail the specific names and function(s) of all bit fields in the register and explain any important variations that may exist.

An important detail to note is that all registers are not perfectly writable and readable. Some will exhibit different characteristics on a write, while a read may not return the expected result. At the same time, there will be registers whose function on a write is to clear a value or a set of stored values, while on a read will return a specific set of values. This is particularly true in registers that handle interrupts. Writing to a specific register may clear a set of interrupts, while reading that same register will yield which interrupts are set.

Similarly, not all bit fields in all registers can be written, nor can all register bit fields yield useful information when read. These restricted register bit fields will be specifically called out with three slashes (///) and the word 'Reserved', along with their special conditions in the bit field tables. See Table 1 and Table 2 for examples of this practice.

Table 1. Register Name

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///						F25	///								
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RW	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	—															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								F07	F06	F05	F04	F03	F02	F01	F00
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	REGISTERBASE + 0x0004															

Table 2. Bit Fields

BITS	FIELD NAME	FUNCTION
31:26	///	**Reserved** Reading returns 0. Values written cannot be read.
25	F25	**Field 25** A description of this bit's functionality will be found in this space.
24:8	///	**Reserved** Reading returns 0. Writing to this field will have no effect.
7:0	F7:F0	**Field Bits [7:0]** A description of these bits' functionality will be found in this space.

NOTES:
RO = Read Only
WO = Write Only
RW = Read and Write

Numeric Values

Binary values are prefixed with 0b; for example, 0b00001000.

Hexadecimal values are expressed with UPPERCASE letters and prefixed with 0x; for example, 0x0FBC.

All numeric values not specifically identified with the above prefixes as either binary or hexadecimal are decimal values.

Registers and bit fields with 0b0 in all bits are referred to as cleared or as 0. Registers and bit fields with 0b1 values in all bits are referred to as set or as the binary, hexadecimal, or decimal value of the entire field or register. When truth tables are used, the '0b' prefix is omitted for textual clarity.

Block Diagrams

The functional descriptions in this User's Guide include block diagrams with symbols representing logical or mathematical operations or selections, usually the result of writing a value to a register. Figure 1 shows one such multiplexer with three inputs and one output (the result).

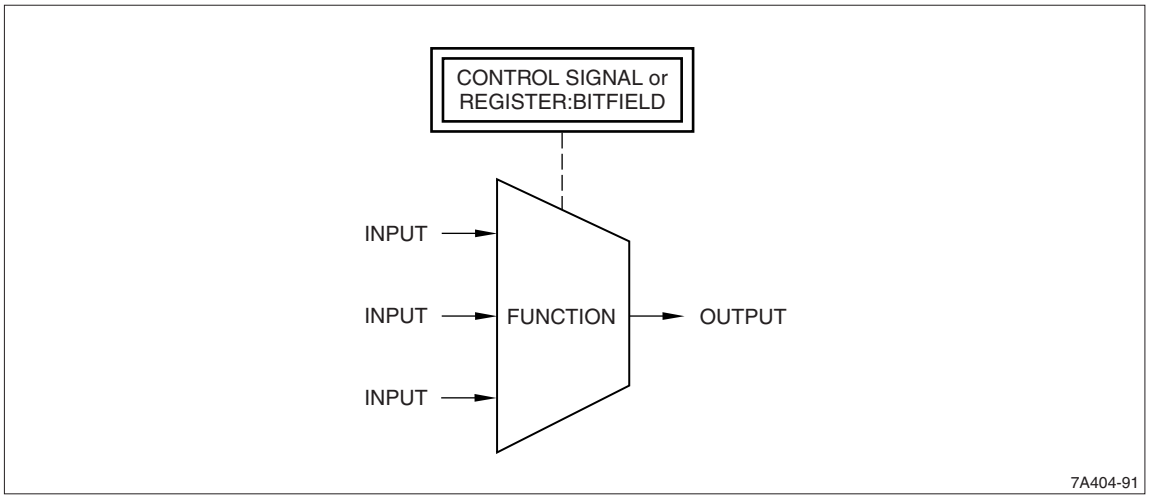


Figure 1. Multiplexer

Block diagrams can include symbols representing Registers and the bit fields within them. Figure 2 shows that the BITFIELDNAME bit field in the REGISTERNAME register enables or disables the signal named OUTPUT.

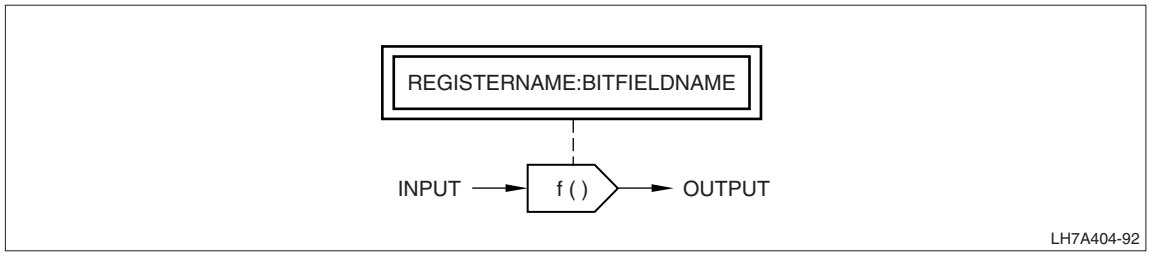


Figure 2. Register with Bit-Field Named

Figure 3 is similar to Figure 2 except that Figure 3 references multiple (different) BITFIELDS in the REGISTERNAME register.

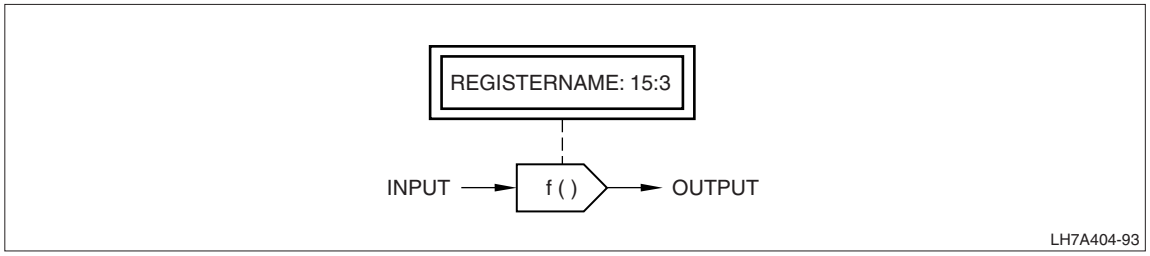


Figure 3. Register with Multiple Bit-Fields Named

Not all bit fields are named. If a bit field has no name, the Register is shown with numbers indicating the appropriate bit positions, with the least significant bit on the right, as in Figure 4. This bit ordering matches that of the Register tables, shown in Table 1.

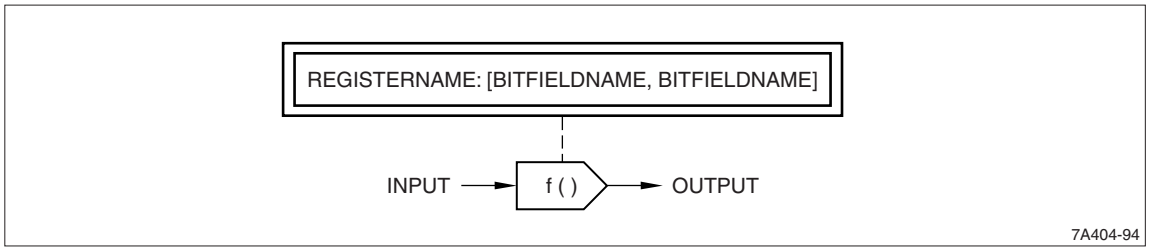


Figure 4. Register with Bit-Field Numbered

What's in This User's Guide

Chapter 1 – Overview

This Chapter lists the features of the LH79524/LH79525 MCU and presents a simplified block diagram of the device, with the major architectural features identified. Also presented is an overview of the ARM720T processor and MMU. The theory of operation covers bus architecture, bus arbitration, and the base addresses for each of the Advanced High-Performance Bus (AHB) and Advanced Peripheral Bus (APB) devices and the APB Bridge. Coverage of the memory system includes memory mapping, memory remapping, and the External Bus Interface (EBI). This Chapter provides programmer's models, programmable parameters, default memory widths, address mapping, and includes a register summary and register descriptions.

Chapter 2 – ADC and Brownout Detector

This Chapter describes the 10-channel, 10-bit Analog to Digital Converter, and its associated Brownout Detector. Theory of operation includes both touch-screen applications and traditional ADC applications. This Chapter provides programmer's models, programmable parameters, default memory widths, address mapping, and includes a register summary and register descriptions.

Chapter 3 – Boot Controller

This Chapter describes alternate booting options, their use and configuration. Also included is a programmer's model, address mapping, and a register summary and register descriptions.

Chapter 4 – Color LCD Controller

This Chapter describes the Color LCD Controller (CLCDC) and the Advanced LCD Interface Controller (ALI) functional blocks within the LH79524/LH79525. The Chapter includes a brief overview, lists the types of panels supported, and at what bit-depths. The Chapter also lists and explains the programmable parameters and includes a register summary. Register descriptions, with reset values, and horizontal timing restrictions are provided.

Chapter 5 – DMA Controller

This Chapter describes the DMA operations available in the LH79524/LH79525 MCU, latencies from one process to another, and the interrupts involved. Also included is a programmer's model, address mapping, and a register summary and register descriptions.

Chapter 6 – Ethernet MAC

Included in this chapter is a description and programming information for the MCU's Ethernet MAC. Also included is a programmer's model, address mapping, and a register summary and register descriptions.

Chapter 7 – External Memory Controller

This Chapter presents the theory of operation of the LH79524/LH79525 External Memory Controller (EMC), including programmable parameters, device selection, memory widths, and address mapping. This Chapter includes a register summary and register descriptions for the EMC.

Chapter 8 – General Purpose Input/Output

This Chapter presents the LH79524/LH79525 General Purpose Input/Output (GPIO) systems, beginning with a brief overview, and including a block diagram, programmer's model, register summary, and register descriptions.

Chapter 9 – I^2C Interface

The I^2C Interface is described in this chapter. The Chapter includes a short overview, a block diagram, programmer's model, interrupt channel list, register summaries, and register descriptions.

Chapter 10 – I^2S Converter

This Chapter describes the I^2S Converter. This peripheral converts a synchronous serial communication stream in Texas Instruments DSP-compatible mode to an I^2S-compliant synchronous serial stream. The I^2S converter operates on serial data in both master and slave mode. The Chapter includes a short overview, a block diagram, programmer's model, interrupt channel list, register summaries, and register descriptions.

Chapter 11 – I/O Configuration

This Chapter is an overview of the LH79524/LH79525 I/O Configuration and pin multiplexing. The Chapter provides a block diagram, programmer's model, register summary and descriptions.

Chapter 12 – Real Time Clock

This Chapter describes the LH79524/LH79525 Real Time Clock (RTC). The Chapter includes a short overview, a block diagram, a list of clock signals, programmer's model, signal descriptions, operating sequences, register summaries, register descriptions and interface signals.

Chapter 13 – Reset, Clock Generation and Power Control

This chapter provides a short overview of the LH79520 Reset, Clock Generation and Power Control (RCPC) system, including a block diagram, a list of clock signals, power control modes, programmer's model, signal descriptions, power sequences, register summaries, register descriptions, and descriptions of interface signals.

Chapter 14 – Synchronous Serial Port

This Chapter presents an overview of the LH79524/LH79525 Synchronous Serial Port, a block diagram, programmer's model, register summary, register descriptions, Interrupts, and register locations.

Chapter 15 – Timers

This Chapter describes the LH79524/LH79525 Timers. The Chapter includes a short overview and block diagram, signal descriptions, operation sequences, register summaries, register descriptions, and interface signals.

Chapter 16 – UARTs

This Chapter presents the LH79524/LH79525 UART blocks. The Chapter includes a brief overview, block diagram, programmer's model, programmable parameters, register summary and register descriptions.

Chapter 17 – USB Device

This Chapter presents the LH79524/LH79525 USB Device, beginning with a brief overview, and including a block diagram, programmer's model, register summary, and register descriptions.

Chapter 18 – Vectored Interrupt Controller

This Chapter describes the LH79524/LH79525 Vectored Interrupt Controller. The Chapter includes a short overview, a block diagram, programmer's model, interrupt channel list, register summaries, and register descriptions.

Chapter 19 – Watchdog Timer

This Chapter describes the LH79524/LH79525 Watchdog Timer (WDT). The Chapter includes a short overview, block diagram, programmer's model, signal descriptions, operating sequences, register summaries and register descriptions.

Appendix – Glossary

This Chapter contains an alphabetical listing of common terminology appearing in this User's Guide.

Chapter 1
Overview

The LH79524 and LH79525 are fully-integrated 16/32-bit MCUs based on a 32-bit ARM720T core. The 32-bit ARM720T RISC core provides a powerful instruction set and includes Cache RAM, a Write Buffer, Memory-Management Unit (MMU), and Translation Lookaside Buffer (TLB). Both MCUs include a Color LCD Controller, a Direct Memory Access Controller, Vectored Interrupt Controller, 16KB of internal Static RAM (SRAM), and several supporting peripherals. The External Memory Controller (EMC), provides a glueless interface to external memory.

Supporting function blocks within the LH79524/LH79525 include Serial and Parallel Interfaces, Counters/Timers, Real Time Clock, Watchdog Timer, Pulse Width Modulators, and an on-chip Phase-Locked Loop. JTAG support is provided to simplify debugging.

Table 1-1 summarizes the differences in features between the LH79524 and the LH79525. All other peripherals and functional blocks are identical (unless noted in the Chapter detailing that block's function). The block diagram for both devices appears in Figure 1-1. Refer to it when reading sections detailing bus architecture and functional block descriptions.

Table 1-1. LH79524/LH79525 Differences

FEATURE	LH79524	LH79525
Package	208 CABGA	176 LQFP
Data Bus Width	32-Bit Data Bus that includes all peripherals	16-bit Data Bus that includes all peripherals
Color LCD Controller (CLCDC)	16-bit CLCDC Data	12-bit CLCDC Data
General Purpose Input/Output (GPIO)	92 GPIO, 8 General Purpose Input only (GPI), 8 General Purpose Output only (GPO)	72 GPIO, 8 GPI, 6 GPO

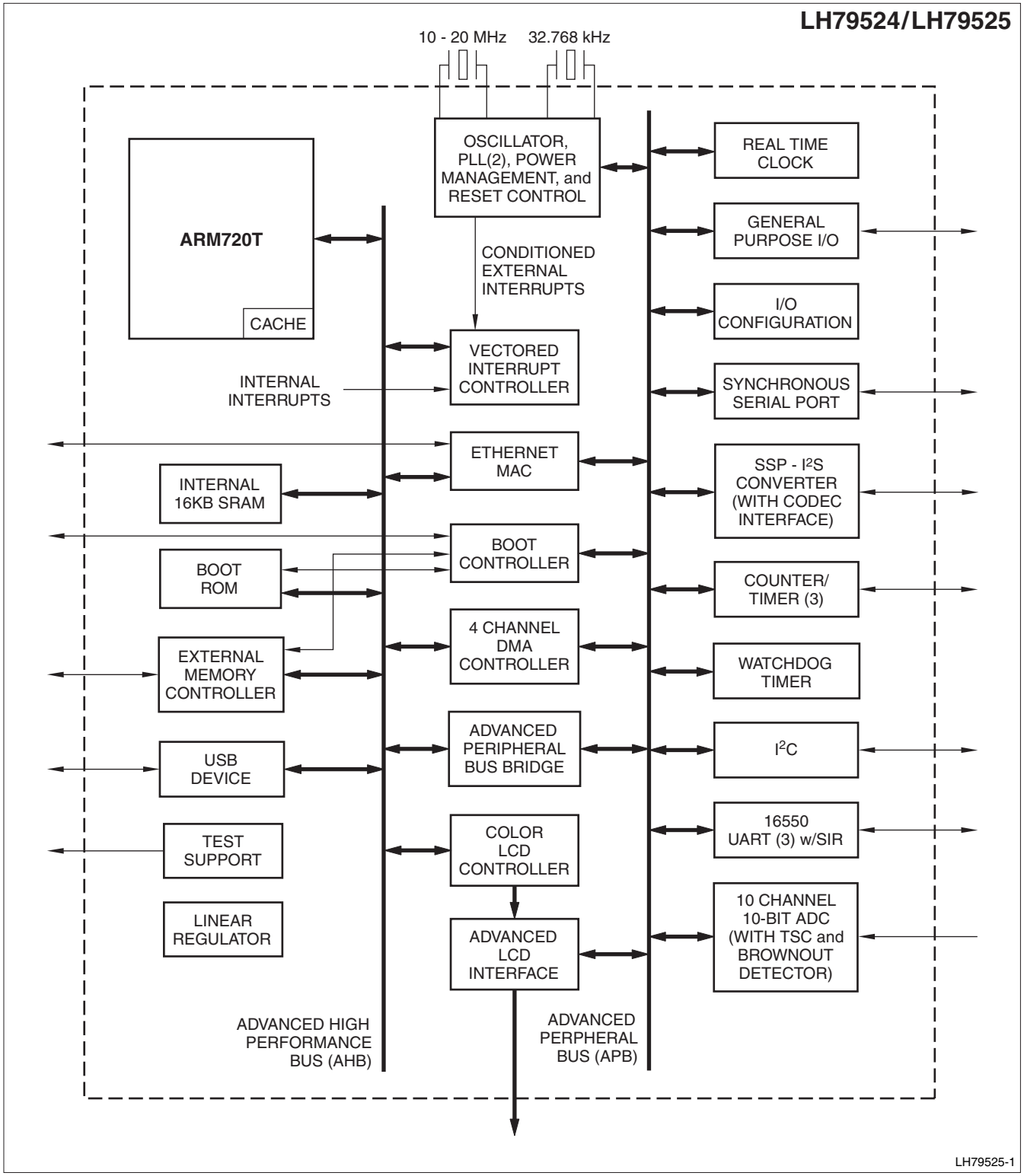


Figure 1-1. LH79524/LH79525 Block Diagram

1.1 Bus Architecture

The LH79524 and LH79525 both internally employ the ARM Advanced Microprocessor Bus Architecture (AMBA) 2.0 bus and bus protocol. They have four Bus Masters on the Advanced High-performance Bus (AHB) that control access to the external memory and the on-chip peripherals. The AHB Bus Masters are:

- The ARM720T core processor
- Direct Memory Access Controller (DMAC) for transfers between memory and an external peripheral, or memory.
- Ethernet MAC Controller (EMAC)
- Color LCD Controller (CLDCC).

Except in test mode, the ARM720T processor is the default bus master.

An Advanced Peripheral Bus (APB) bridge is provided to access the various APB peripherals. Generally, APB peripherals are serviced by the ARM core, however, if they are DMA enabled, they would also be serviced by the DMA controller to increase system performance while the ARM core is running from cache.

1.2 Power Supply

The MCU's core logic requires a 1.8 V supply. Digital Input/Output pins are 5 V tolerant and require a 3.3 V supply. They are designed to operate from a single 3.3 V supply. An on-chip 1.8 V-to-3.3 V linear regulator can be used to generate the 1.8 V needed by the core logic.

1.2.1 Linear Regulator

When the linear regulator is enabled, the 1.8 V power pins (VDDC) are outputs of the regulator. This allows regulator operation verification. In addition, an external low-ESR capacitor must be tied to the regulator output for stability. If the regulator is disabled, the 1.8 V power pins are used as inputs from an external 1.8 V supply.

The linear regulator is enabled by tying the LINREGEN pin to 3.3 V; it is disabled by holding the LINREGEN pin LOW. Proper power-up sequencing must be considered when employing the linear regulator. In order to ensure this takes place, nRESETIN must be held LOW until the linear regulator has ramped up to nominal operating voltage.

The linear regulator must only be used to power the MCU and internal devices; it is not intended to supply power to off-chip peripherals. Powering external devices can result in unpredictable behavior or device failure.

1.2.2 Phase Locked Loop Power

Two PLLs provide accurate, on-board clocks (see Section 1.3). The PLLs require a 1.8 V supply. If the linear regulator is disabled, the supply must come from an external source. If the linear regulator is enabled, the PLL power supply comes from the internal VDDC power pins.

1.3 Clock Strategy

The MCUs have two crystal oscillators. One oscillator, CLK OSC, is used to drive both PLLs and the three UARTs, among others. This oscillator supports a frequency range from 10 to 20 MHz. The second oscillator, RTC CLK, is a 32.768 kHz oscillator, also requiring a 1.8 V source. This oscillator is used to generate a 1 Hz clock for the Real-time Clock.

The clock circuitry has two PLLs — one for the system clock generation and the other for the USB clock generation. The output frequency of the PLLs ranges from 20 MHz to 304.819 MHz based on the PLL programmable dividers' values.

The system clock frequency created in the Reset, Clock, and Power Controller (RCPC) can be programmed to divide the PLL frequency by 1 or any even divisor between 2 and 30. The maximum ARM720T core operating frequency is 76.205 MHz and maximum system operating frequency of 50.803 MHz. If UARTs 0, 1, or 2 are to be used, the system clock frequency must not be set to less than 50% of the frequency applied to the crystal input pin (XTALIN) for proper UART operation.

Table 1-2 is a list of the internal clocks with maximum frequency.

Table 1-2. Clock Descriptions

NAME	FREQUENCY (MAX.)	DESCRIPTION
System Oscillator Clock (CLK OSC)	20 MHz	External crystal oscillator input.
32.768 kHz RTC OSC	32.768 kHz	External 32.768 kHz crystal oscillator input.
1 Hz Clock	1 Hz	The 1 Hz Clock is derived by dividing the RTC OSC by 32,768.
PLL System Clock (CLK PLL)	304.819 MHz	This is the output from the System PLL. The input for this clock is CLK OSC, the System Oscillator Clock. The minimum output frequency is 5 MHz.
USB PLL Clock (USB PLL)	304.819 MHz	This is the output from the USB PLL; the input is CLK OSC. It can be programmed for any frequency between 5 MHz and 304.819 MHz.
AHB Fast CPU Clock (FCLK)	76.205 MHz	This clock controls the CPU instruction execution speed. It is derived from the CLK PLL clock, and is prescaled by 2, 4, ...30. The clock is halted HIGH when the RCPC is in any power down mode other than Standby.
AHB Clock (HCLK)	50.803 MHz	This clock controls the AHB execution speed. It is derived from CLK PLL and its frequency is CLK PLL divided by 2, 4, ...30. The clock is halted HIGH when the RCPC is in any power down mode other than Standby mode. It can be programmed for power savings to turn off clock individually to DMAC, EMC, EMAC, USB, and CLCDC.
USB Clock	48.0 MHz	The USB Clock controls the 12 MHz full-speed USB Device interface. Selectable input from HCLK or USB PLL. Frequency is required to be 48 MHz for proper USB operation (hardware divides by 4).
SSP Clock	50.803 MHz	This clock controls the SSP and the I^2S interfaces. Source is either HCLK or CLK OSC. Can be divided by 2^n ($n \leq 8$). This clock can be individually halted for power savings.
ADC Clock	50.803 MHz	Controls the Touch Screen Controller (TSC) and Brownout Detector. Input source choice of HCLK or CLK OSC. Source can be divided by 2^n ($n \leq 8$). This clock can be individually halted for power savings.

Table 1-2. Clock Descriptions (Cont'd)

NAME	FREQUENCY (MAX.)	DESCRIPTION
CLCD Clock	50.803 MHz	This clock controls the data rate for pixel transfers to an external LCD panel. This clock can be separately enabled, disabled and prescaled. Source can be divided by 2^n ($n \leq 8$). This clock can be individually halted for power savings.
Serial Interface Clock (UART[2:0])	20 MHz	These clocks control the data transfer rates over the three UART interfaces. These clocks are all separate and can be separately enabled and disabled. Clock source can be selected from HCLK or CLK OSC.
Counter/Timer Clocks	25.415 MHz	These clocks control the transition rates for the internal timers. The source can be selected from HCLK or the External Timer input (CTCLK). Each timer is either clocked by CTCLK or HCLK divided by 2^n ($0 < n \leq 8$).
RTC Clock	32.768 kHz	This clock controls the transition rate for the internal real-time clock. The source can be selected from the 1 Hz Clock, RTC OSC, CLK PLL, or an External RTC Clock connected to the XTAL32IN pin.
Clock Output (CLKOUT)	50.803 MHz	This output clock is available on pin CLKOUT for use with external peripherals. Input source can be FCLK, HCLK, or CLK OSC.

1.3.1 Bus Clocking Modes

The ARM720T core (including the cache) and its AHB interface can be operated using either the Fastbus operation mode or one of two Standard clocking modes (Synchronous or Asynchronous).

The clocking modes can have significant impact on power consumption and system throughput, depending upon the application and the speed of external memory. The ARM720T core and the AHB are clocked by separate signals and the core is capable of operation at a much higher frequency than the AHB. This higher core speed benefits applications running from cache more than applications requiring frequent AHB access because each AHB access requires the core and AHB be re-synchronized. Parallel core and AHB operations can continue with buffered writes to the AHB, but all Read accesses will stall the core until the bus access is completed. Programmers can use the three bus clocking modes to maximize throughput by reducing the re-synchronization delays (the number of wait states).

1.3.1.1 Standard Bus Clocking Modes

The Standard bus clocking modes are useful for designs involving low-cost, low-speed memory, where operation of the core at a faster speed than the AHB is desired. These modes involve:

- A programmable choice of Synchronous or Asynchronous operation
- Two clocks: HCLK and FCLK.

The AHB interface is controlled by the bus clock (HCLK), qualified by an nWAIT signal. Figure 1-2 shows the Standard mode clocking arrangement. The core and cache are driven by FCLK while HCLK drives the bus. FCLK must always be greater than or equal to HCLK, on a cycle-by-cycle basis. The nWAIT signal can extend a memory access by inserting entire HCLK cycles into the bus cycle timing.

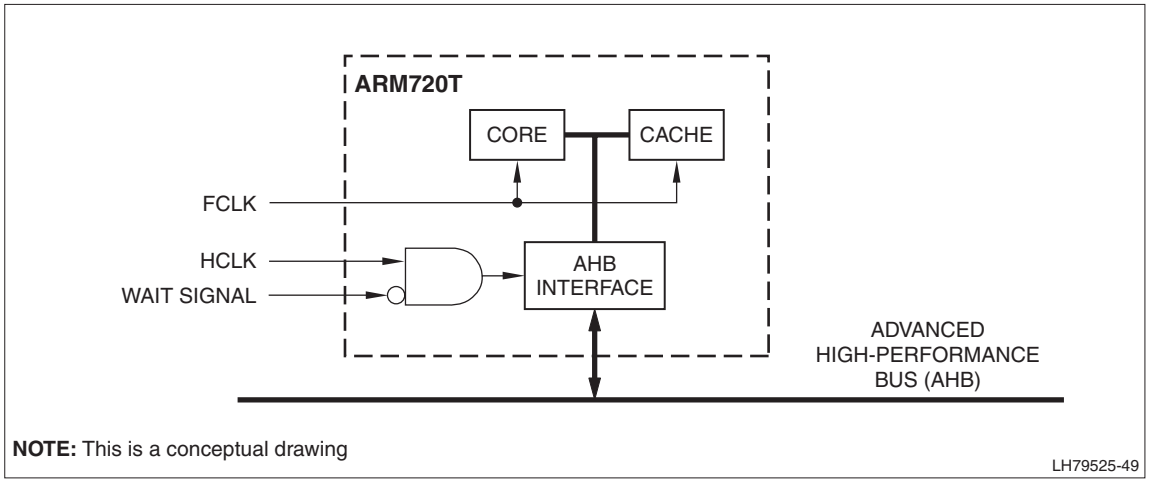


Figure 1-2. Standard Clocking Modes

1.3.1.2 Synchronous and Asynchronous Bus Clocking Modes

Although the frequency of FCLK must always be greater than (or equal to) HCLK, the two Standard modes vary the relationship between these two clock signals. In the Synchronous Mode, the FCLK frequency must be programmed to be an even integer multiple of the HCLK frequency. Bus accesses in the Synchronous Mode require a re-synchronization delay of at least one wait state. In the Asynchronous Mode the harmonic relationship between the clocks need not be maintained; the two clock signals may be of unrelated frequency. Bus accesses in the Asynchronous Mode require a minimum re-synchronization delay of two wait states.

1.3.1.3 Fastbus Extension Bus Clocking Mode

Designs involving frequent accesses of high-speed memory may benefit by using the Fastbus Extension Mode. This inherently synchronous mode clocks the core, cache, and AHB at the same frequency. Where the Standard modes utilized two different clocks, the Fastbus mode operates the core, cache, and AHB interface with two signals derived from the same source; essentially the same clock. Figure 1-3 shows the Fastbus Extension Mode clocking arrangement. The Fastbus Extension Mode does not require re-synchronization delays.

The Fastbus Extension Mode is useful for applications involving frequent AHB accesses. Although the core's frequency is limited by the AHB maximum frequency, the Fastbus Extension Mode avoids the wait-state penalties imposed by the Standard modes.

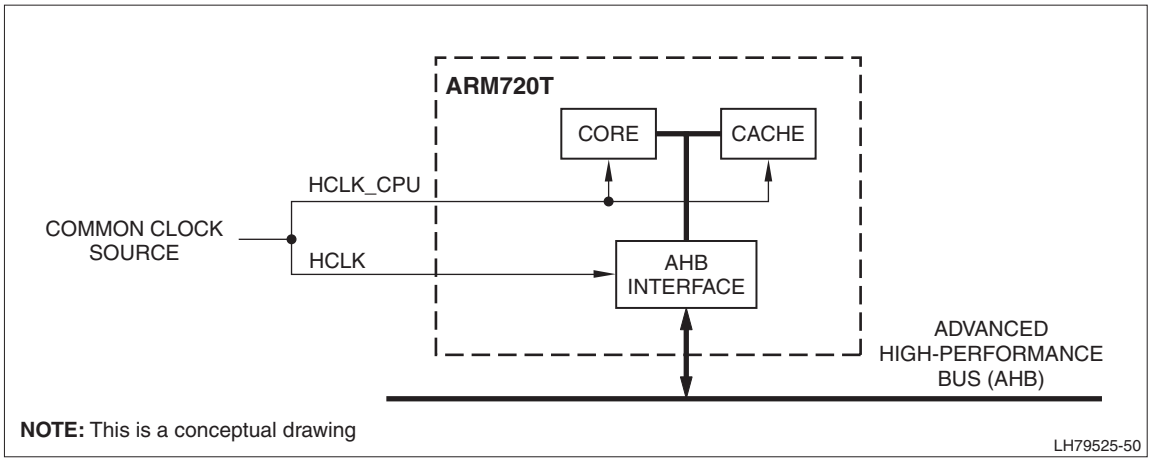


Figure 1-3. Fastbus Clocking Mode

1.4 Reset Strategy

Two external resets, nRESETIN, and nTRST, are used for the LH79524/LH79525. If nRESETIN is asserted, all internal registers EXCEPT the JTAG circuitry within the device are set to their default state. The nRESETIN signal should be held LOW for the crystal stabilization time + 200 μs (the time varies depending on crystal used) during power-up. If nTRST is asserted, only the JTAG circuitry is set to its default state.

There are two types of internal resets for the LH79524/LH79525. A software reset resets all internal registers, except the JTAG circuitry, to their default state. The other internal reset is the watchdog timer (WDT) reset, which also resets all internal registers, except the JTAG circuitry, to their default state. For more information on these internal resets, refer to the Reset, Clock, and Power Controller, and Watchdog Timer chapters.

This document uses the term 'system reset' to refer to either an nRESETIN reset, software reset, or a watchdog timer reset. The system reset is also brought out to an external pin (nRESETOUT). The nRESETOUT pin is held LOW for 8 HCLKs after HCLK becomes active following a system reset.

At power-on reset (nRESETIN), the type of memory that the CPU boots from is determined by the state that PC7, PC6, PC5, and PC4 are externally connected to, as shown in Table 1-3. If left undriven, the default value is 0x0, as determined by internal pull-down resistors. If the CPU is to boot from external memory, the nCS1 Chip Select is used. If the CPU is to boot from UART, UART0 is used.

Table 1-3. Port C Settings For Boot

PC[7:4]	BOOT CONFIGURATION
0x0	NOR Flash or SRAM; 16-bit data bus; nBLEx is LOW for reads
0x1	NOR Flash or SRAM; 16-bit data bus; nBLEx is HIGH for reads
0x2	NOR Flash or SRAM; 8-bit data bus; nBLEx is LOW for reads
0x3	NOR Flash or SRAM; 8-bit data bus; nBLEx is HIGH for reads
0x4	NAND Flash; 8-bit data bus; 3-byte address
0x5	NAND Flash; 8-bit data bus; 4-byte address
0x6	NAND Flash; 8-bit data bus; 5-byte address
0x7	NAND Flash; 16-bit data bus; 3-byte address
0x8	NOR Flash or SRAM; 32-bit data bus; nBLEx is LOW for reads
0x9	NOR Flash or SRAM; 32-bit data bus; nBLEx is HIGH for reads
0xA	Undefined
0xB	Undefined
0xC	NAND Flash; 16-bit data bus; 4-byte address
0xD	NAND Flash; 16-bit data bus; 5-byte address
0xE	I^2C
0xF	UART0

1.4.1 Resetting the Test Access Port Controller

The on-chip Test Access Port (TAP) Controller has an independent reset pin, nTRST. However, it must also be reset at power on, or any time the MCU is reset to ensure it exits the power up sequence in Normal Mode.

To ensure this, an external AND gate is necessary to AND nTRST and nRESETIN. Figure 1-4 illustrates the minimal circuit capable of guaranteeing the proper reset signals. If the application will require a push button reset, the circuit in Figure 1-5 is recommended.

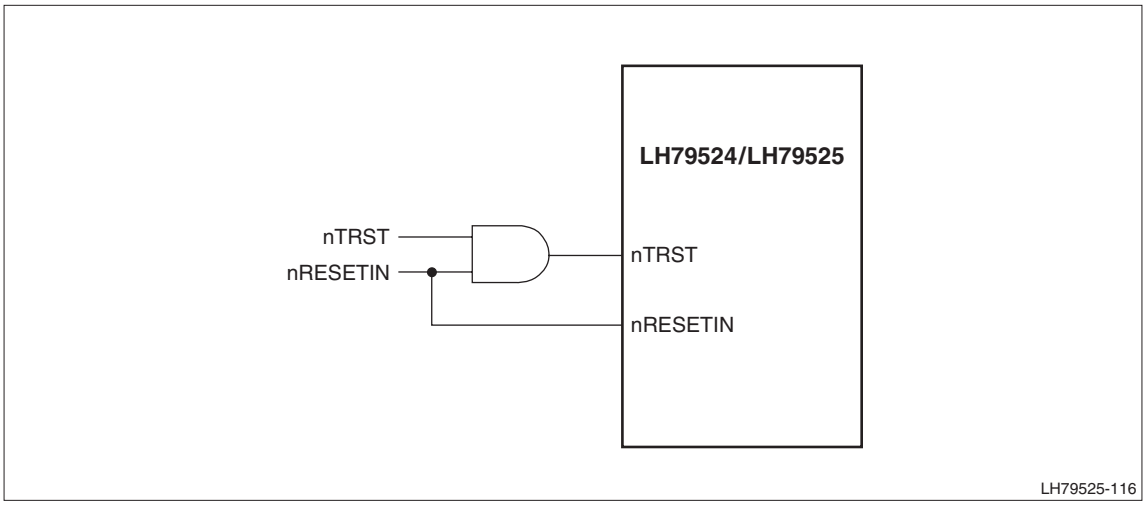


Figure 1-4. Reset Circuit for TAP Controller

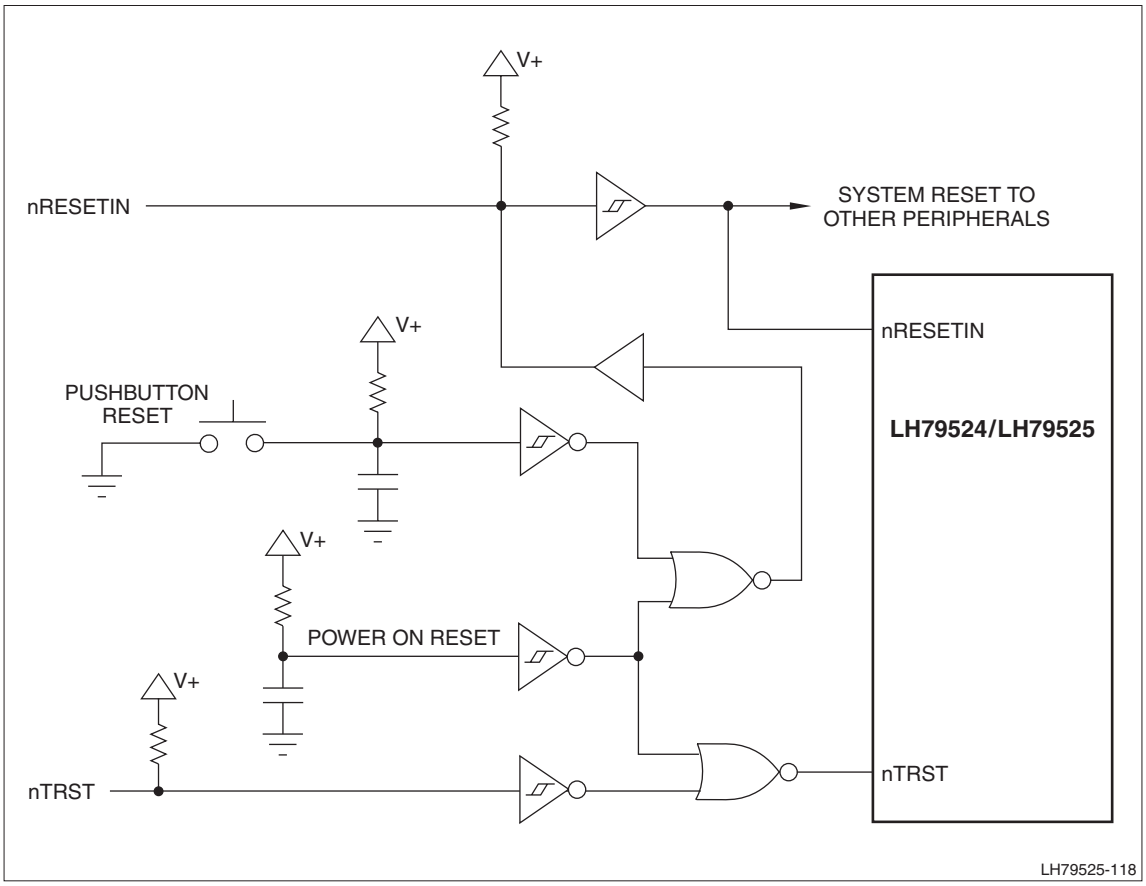


Figure 1-5. Reset Circuit for TAP Controller Including a Push Button

1.4.2 Hardware Requirements at Reset

A number of pins contain on-chip pull up or pull down resistors that provide a logic state following reset. Other pins require external pull up or pull down resistors because their state is read by the core prior to power becoming stable. Thus the state of these pins cannot be guaranteed using the internal resistors.

1.4.2.1 Floating Inputs

Many applications require extremely low standby and operating current consumption, especially in battery operated devices. For minimum current, unused inputs must never be left floating (unconnected). Each input must be pulled up or pulled down with a 33 kΩ resistor (or smaller). In addition to terminating input pins, this also allows selecting the reset state of input pins using pull up (logical 1 at reset) or pull down (logical 0 at reset) resistors.

1.4.2.2 Test Pins

The two test pins, TEST1 and TEST2, require being tied HIGH for the MCU to boot into Normal Operation Mode. Without tying these pins HIGH, the chip may boot into PLL Bypass Mode. To enter Embedded ICE Mode, TEST1 is pulled LOW and TEST2 pulled HIGH; nBLE0 has a sufficient internal pull up.

1.4.2.3 Active Pull Ups

The boot mode — NOR Flash, NAND Flash, SRAM, I2C, or UART — is selected by the value latched on the rising edge of the nRESETOUT signal from the state of Port C, pins [7:4]. Pins PC[7:6] are used during NAND Flash booting as control signals, but PC[5:4] have no other use following the end of reset. Therefore, those two GPIO pins can be used during normal operation if an active pullup is used, gated by the nRESETOUT signal.

Figure 1-6 shows a schematic representation of one active pullup circuit. One circuit is required for each PCx pin to be pulled high during reset. nRESETOUT is presented to the Gate (pin 1) of the P-Channel FET. When active (LOW), nRESETOUT causes the transistor to turn on, and pull the PCx input HIGH. When nRESETOUT transitions from LOW to HIGH at the end of the reset period, the value on PC[7:4] is latched and the FET is turned off, thus allowing those pins to be used for general purpose I/O or as address pins A[21:20]. As shown in the figure, a common pull up resistor can be used for all of the FETs.

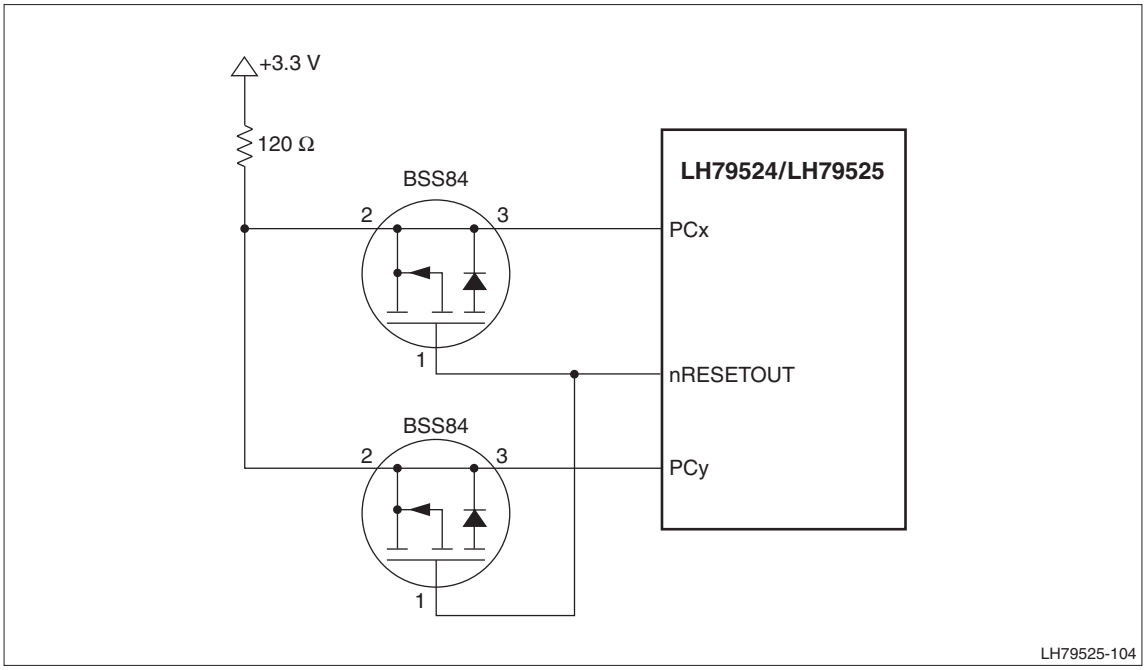


Figure 1-6. Active Pullup Circuit

1.5 AHB Bus Master Priority and Arbitration

The LH79524/LH79525 have five AHB masters - the ARM720T processor, the DMA Controller, the Color LCD Controller, USB Device, and the Ethernet Controller. Two of the masters — the ARM720T processor and the DMA controller — are capable of communicating with all the memory controllers and peripherals. The LCD Controller, USB Device, and Ethernet Controller interface to the main AHB bus via a slave interface for programming and via a master interface for accessing SDRAM and Static Memory Controllers.

The default priorities for the five different AHB masters are indicated in Table 1-4.

Table 1-4. Default Bus Master Priority

PRIORITY	BUS MASTER PRIORITY
1 (Highest)	CLCD Controller
2	Ethernet
3	USB Device
4	DMA Controller
5 (Lowest)	ARM720T Core (Default)

1.6 Memory Interface Architecture

The LH79524/LH79525 provides the following data-path-management resources on chip:

- AHB and APB data buses
- 16KB of internal SRAM accessible by the ARM720T processor, DMA Controller, Ethernet Controller, or LCD Controller
- A static and dynamic memory controller with a 24-bit address and 16/32-bit data interface
- A 4-channel general purpose DMA controller

All system resources accessible by the LH79524/LH79525 are memory mapped. These include external resources (e.g. ROM, PROM, SRAM, SDRAM, External Peripherals) and internal resources (system configuration registers, peripheral configuration registers, and internal memory).

The external memory space is partitioned into eight banks. Each bank spans 512MB. The start address of each bank is fixed and is determined by the three highest order bits of the 32-bit AHB address. These banks define the type of resource being addressed. One bank can only contain external static memory devices connected to the External Bus Interface (EBI). Another bank can only contain external SDRAM devices connected to the EBI. Another bank contains only the internal SRAM, connected to the AHB. Finally, another bank is reserved for accessing the system configuration registers themselves, as well as many of the peripheral control registers. See Table 1-5.

This memory map partition has four configurations, based on the setting of the REMAP bits in the Reset, Clock, and Power Controller.

The external static memory bank is divided into four sections, each having a Chip Select associated with it. Each section has 24 address lines. When using 32-bit wide memories each section is 64 MB, 16-bit memories have 32 MB sections, and 8-bit memories have 16 MB sections. When the Chip Select doesn't occupy the complete bank, it is aliased and doesn't cause a memory abort. The external SDRAM bank is divided into two 256 MB sections, each having a Chip Select associated with it. The peripheral register section is divided into 4KB peripheral sections. See Table 1-6 through Table 1-12.

Table 1-5. AHB Memory Mapping

ADDRESS	REMAP = 00	REMAP = 01	REMAP = 10	REMAP = 11
0x00000000 - 0x1FFFFFFF	nCS1	nDCS0	Internal SRAM	nCS0
0x20000000 - 0x3FFFFFFF	SDRAM	SDRAM	SDRAM	SDRAM
0x40000000 - 0x5FFFFFFF	Static Memory	Static Memory	Static Memory	Static Memory
0x60000000 - 0x7FFFFFFF	Internal SRAM	Internal SRAM	Internal SRAM	Internal SRAM
0x80000000 - 0x80001FFF	Boot ROM	Boot ROM	Boot ROM	Boot ROM
0x80002000 - 0xFFFBFFFF	Invalid Access			

NOTES:
1. REMAP is initialized to '00' upon system reset.
2. Right after system reset or when REMAP = 00, external static memory Chip Select 1 is mapped to lower memory (0x00000000 - 0x1FFFFFFF), which means the same Chip Select (nCS1) can be accessed from two locations: 0x00000000 and 0x44000000.
3. Programming REMAP to 01 will map SDRAM Chip Select 0 to lower memory, which means the same Chip Select (nDCS0) can be accessed from two locations: 0x00000000 and 0x20000000.
4. Programming REMAP to 10 will map internal SRAM to lower memory, which means the same physical memory can be accessed from two locations: 0x00000000 and 0x60000000.
5. Programming REMAP to 11 will map external static memory Chip Select 0 to lower memory, which means the same Chip Select (nCS0) can be accessed from two locations: 0x00000000 and 0x40000000.
6. Invalid Access memory areas will cause a memory abort.

Table 1-6. External Static Memory Section Mapping

START ADDRESS REMAP = 'XX'	EXTERNAL DEVICE	PIN	NOTES
0x40000000 - 0x43FFFFFF	Chip Select 0	nCS0	1
0x44000000 - 0x47FFFFFF	Chip Select 1	nCS1	2
0x48000000 - 0x4BFFFFFF	Chip Select 2	nCS2	
0x4C000000 - 0x4FFFFFFF	Chip Select 3	nCS3	
0x50000000 - 0x5FFFFFFF	Invalid Access	///	3

NOTES:
1. Also accessible at 0x00000000 when REMAP = 11
2. Also accessible at 0x00000000 when REMAP = 00
3. An access to this area will cause a memory abort

Table 1-7. SDRAM Memory Section Mapping

START ADDRESS REMAP = 'XX'	DEVICE	PIN
0x20000000 - 0x2FFFFFFF	Chip Select 0	nDCS0*
0x30000000 - 0x3FFFFFFF	Chip Select 1	nDCS1

NOTE: *Also accessible at 0x00000000 when REMAP = 01.

Table 1-8. Internal SRAM Memory Section Mapping

START ADDRESS REMAP = 'XX'	DESCRIPTION	NOTES
0x60000000 - 0x60003FFF	16 KB Internal SRAM	1
0x60004000 - 0x7FFFFFFF	Internal SRAM (mirrored)	2

NOTES:
1. Also accessible at 0x00000000 when REMAP = 10
2. An access to this area is mapped to the lower 16KB and will not cause a memory abort

Table 1-9. Boot ROM Memory Section Mapping

START ADDRESS REMAP = 'XX'	DESCRIPTION
0x80000000 - 0x80001FFF	8 KB Boot ROM
0x80002000 - 0x9FFFFFFF	Invalid Access*

NOTE: *An access to this area will cause a memory abort.

If, following system reset, the boot configuration is set to 0bX1XX, an override of nCS1 occurs. In this circumstance, the Boot ROM is selected for the locations in the memory map where nCS1 is normally selected. This causes the CPU to execute the predefined code contained in the Boot ROM, allowing booting from NAND Flash, UART, or I^2C; see Table 1-10. This override can be disabled by writing a 0 to the nCS1 Override bit (CS1OV:CS1O) in the Boot Controller. The override can be re-enabled by writing a 1 to CS1OV:CS1O. If on system reset the boot configuration is set to 0bX0XX, nCS1 remains mapped as described above and CS1OV:CS1O has no effect on the memory map.

Table 1-10. AHB Memory Map on Power-up when Boot Configuration = 0bX1XX

ADDRESS	REMAP = 00
0x00000000 - 0x1FFFFFFF	Boot ROM
0x20000000 - 0x2FFFFFFF	SDRAM nDCS0
0x30000000 - 0x3FFFFFFF	SDRAM nDCS1
0x40000000 - 0x43FFFFFF	Static Memory nCS0
0x44000000 - 0x47FFFFFF	Boot ROM
0x48000000 - 0x4BFFFFFF	Static Memory nCS2
0x4C000000 - 0x4FFFFFFF	Static Memory nCS3
0x50000000 - 0x5FFFFFFF	Invalid Access*
0x60000000 - 0x7FFFFFFF	Internal SRAM
0x80000000 - 0x80000FFF	Boot ROM
0x80001000 - 0xFFFBFFFF	Invalid Access*

NOTE: *An access to this area will cause a memory abort.

Table 1-11. Primary AHB Peripheral Register Mapping

ADDRESS RANGE	DEVICE
0xFFFC0000 - 0xFFFE6FFF	APB Bridge
0xFFFF7000 - 0xFFFF0FFF	Invalid Access*
0xFFFF1000 - 0xFFFF1FFF	External Memory Controller
0xFFFF2000 - 0xFFFF3FFF	Invalid Access*
0xFFFF4000 - 0xFFFF4FFF	Color LCD Controller
0xFFFF5000 - 0xFFFF5FFF	USB Device
0xFFFF6000 - 0xFFFFEFFF	Invalid Access*
0xFFFFF000 - 0xFFFFFFFF	Vectored Interrupt Controller

NOTE: *An access to this area will cause a memory abort.

Table 1-12. APB Peripheral Register Mapping

ADDRESS RANGE	DEVICE
0xFFFC0000 - 0xFFFC0FFF	UART0
0xFFFC1000 - 0xFFFC1FFF	UART1
0xFFFC2000 - 0xFFFC2FFF	UART2
0xFFFC3000 - 0xFFFC3FFF	Analog-to-Digital Convertor
0xFFFC4000 - 0xFFFC4FFF	Timer Module
0xFFFC5000 - 0xFFFC5FFF	I^2C
0xFFFC6000 - 0xFFFC6FFF	Synchronous Serial Port
0xFFFC7000 - 0xFFFC7FFF	Ethernet
0xFFFC8000 - 0xFFFC8FFF	I^2S Converter
0xFFFC9000 - 0xFFFD8FFF	Reserved*
0xFFFD9000 - 0xFFFD9FFF	GPIO Ports M&N
0xFFFDA000 - 0xFFFDAFFF	GPIO Ports K&L
0xFFFDB000 - 0xFFFDBFFF	GPIO Ports I&J
0xFFFDC000 - 0xFFFDCFFF	GPIO Ports G&H
0xFFFDD000 - 0xFFFDDFFF	GPIO Ports E&F
0xFFFDE000 - 0xFFFDEFFF	GPIO Ports C&D
0xFFFDF000 - 0xFFFDFFFF	GPIO Ports A&B
0xFFFE0000 - 0xFFFE0FFF	Real Time Clock
0xFFFE1000 - 0xFFFE1FFF	DMA Controller
0xFFFE2000 - 0xFFFE2FFF	Reset Clock and Power Controller
0xFFFE3000 - 0xFFFE3FFF	Watchdog Timer
0xFFFE4000 - 0xFFFE4FFF	LCD ICP (AD-TFT/HR-TFT/ALI support)
0xFFFE5000 - 0xFFFE5FFF	I/O Configuration Peripheral
0xFFFE6000 - 0xFFFE6FFF	Boot Controller
0xFFFE7000 - 0xFFFEFFFF	Invalid Access

NOTE: *Reads as '0', writes have no effect

1.7 Instruction and Data Cache

The ARM720T Core includes an 8KB Cache, Cache Controller, Memory-Management Unit (MMU) and Write Buffer. A single cache is used for both instructions and data. The cache is an important core feature because the AHB carries all Core, DMA, LCD Display, and Ethernet traffic. For best bandwidth utilization, software should be structured to ensure that the Core is running from within its cache whenever possible.

At reset, the Write Buffer, Cache, and MMU are disabled and the MMU's Translation Lookaside Buffer (TLB) is flushed. If the MMU is utilized, software can determine memory cachability by bank or by page. For best performance, the Color LCD frame buffer should not be located in a cachable region.

1.8 Memory Management Unit (MMU)

The ARM720T core in the LH79524/LH79525 includes an MMU that performs three primary functions: It translates virtual addresses into physical addresses, it enables cache and write buffering for particular ranges of virtual addresses, and it controls memory access permissions. When the MMU is turned off, as it is at reset, all virtual addresses are output directly onto the physical address bus (the AHB).

The MMU supports memory accesses based on 'sections' or 'pages' of memory. Sections are 1MB blocks of memory; pages can be either small or large. Small pages consist of 4KB blocks of memory. Additional access control mechanisms are extended to 1KB subpages. Large pages consist of 64KB blocks of memory. Large pages are supported to allow mapping of a large region of memory while using only a single entry in the Translation Lookaside Buffer (TLB). Additional access control mechanisms are extended to 16KB subpages.

For more information about the core, cache, and MMU, refer to the ARM document 'ARM720T Processor Data Sheet', at http://www.nxp.com/redirect/arm.com.

Chapter 2
Analog-to-Digital Converter/ Brownout Detector

The LH79524/LH79525 incorporate an analog-to-digital converter (ADC) and implements a touch screen controller (TSC) and brownout detector with interrupt.

2.1 Theory of Operation

The ADC and TSC incorporate:

- 10-bit ADC with integrated sample and hold, and fully-differential high impedance signals, and single-ended or ratiometric reference inputs
- A 10-channel multiplexer that routes user-selected inputs to the ADC in single-ended and ratiometric modes
- A 16-entry × 16-bit-wide FIFO containing the 10-bit ADC output
- Active input matrix provides a bias-and-control network for the touch screen interface and support functions, which are compatible with industry-standard 4-, 5-, 7-, and 8-wire touch-sensitive panels
- Pen-down sensing circuit and interrupt generator
- Independently-controlled voltage reference generator
- Conversion automation function to minimize controller interrupt overhead
- Three power modes: Off, Standby, and Run
- Brownout detector with interrupt.

2.1.1 Operational Summary

The ADC is an AMBA-compliant MCU peripheral that connects as a slave to the APB. The ADC block consists of an 10-channel, 10-bit Analog-to-Digital Converter with integrated Touch Screen Controller. The complete touch screen interface is achieved by combining the front-end biasing, control circuitry with analog-to-digital conversion, reference generation, and digital control. Figure 2-1 shows a block diagram of the ADC.

The ADC has a bias-and-control network that allows correct operation with 4-, 5-, 7-, and 8-wire touch panels. A 16-entry × 16-bit wide FIFO holds a 10-bit ADC output and a 4-bit tag number. When the screen is touched, it pushes the conductive coating on the coversheet against the coating on the glass, making electrical contact. The voltages produced are the analog representation of the position touched. The voltage level of the coversheet is converted continuously by the ADC and monitored by the system.

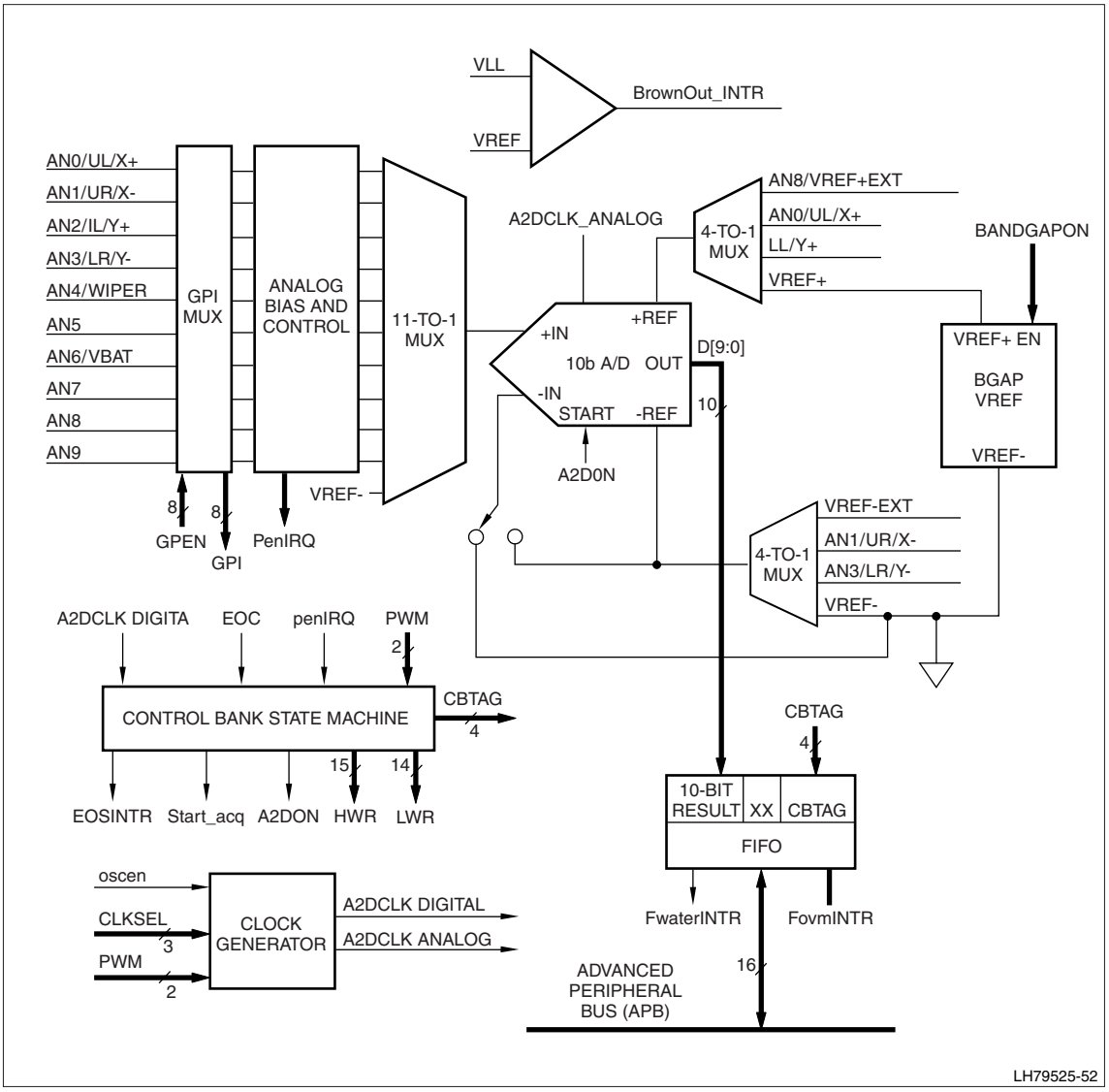


Figure 2-1. ADC Block Diagram

The ADC block can perform a sequence of measurements without intervention from the ARM core. Examples include:

- Determining touch-screen biasing switch configuration.
- Ascertaining how much settling time is required before making a measurement.
- Determining the ADC input source and ADC reference source.

From 1 to 16 different measurements can be performed in a sequence. The number of sequence steps is stored in the PC Register.

The biasing switch configuration, settling time, and ADC mux settings for each of the 1 to 16 measurements in the sequence are stored in an entry in the Control Bank. The measurement sequence can be triggered by either software or a Pen Down Interrupt.

The Control Bank state machine fetches each entry from the Control Bank and stores it in the Low Word register (LW) and High Word register (HW) for the duration of the measurement. When the measurement is complete, the Control Bank state machine stores the ADC result and the Control Bank instruction number in the measurement FIFO, then obtains the next configuration from the Control Bank and loads it into LW and HW.

When all steps of the sequence are complete, or at a programmed FIFO watermark level, the Control Bank state machine signals the ARM core to read results from the FIFO.

From the FIFO, software can read each measurement result and corresponding input configuration, as represented by the Control Bank instruction number. If the FIFO is full the control bank state machine continues to take measurements and the state machine triggers the FIFO Overrun Interrupt.

The ADC can be programmed to repeat the measurement sequence indefinitely, or to pause at the end of a sequence and wait for a new Pen Down Interrupt or software trigger. If the sequence does not repeat continuously, software programs the HW and LW registers with the contents of the Idle High Word (IHWCTRL) and Idle Low Word (ILWCTRL) register values with the bias and ADC multiplexer settings until a new measurement sequence is triggered.

2.1.2 Bias-and-Control Network

The bias-and-control network supports 4-, 5-, 7-, and 8-wire touch panels. Multiplexers on the reference inputs enable connection in both single-ended and ratiometric modes.

- For 4-wire operation, connection is to inputs AN/UL/X+, AN1/UR/X-, AN2/LL/Y+, and AN3/LR/Y-. Pull-up and pull-down FETs allow X and Y coordinate measurement in addition to pen-pressure sensing. The Pen Interrupt line is also available via the Interrupt Masking/Enabling register (see Section 2.2.2.4).
- For 5-wire operation, panel connections are to AN/UL/X+, AN1/UR/X-, AN2/LL/Y+, AN3/LR/Y-, and AN4/WIPER inputs. The Pen Interrupt line is also available in this mode.
- For 7-wire operation, connections are the same as the 5-wire touch panel with a second wire added to the Upper Left and Lower Right corners.
- For 8-wire operation, connections are the same as the 4-wire touch panel with a second wire added to each of the connections. This configuration also requires a single external MOSFET.
- Details for wiring 4-, 5-, 7, and 8-wire touch panels appear in the application note 'Using the NXP ADC with Resistive Touch Screens', available at http://www.nxp.com.

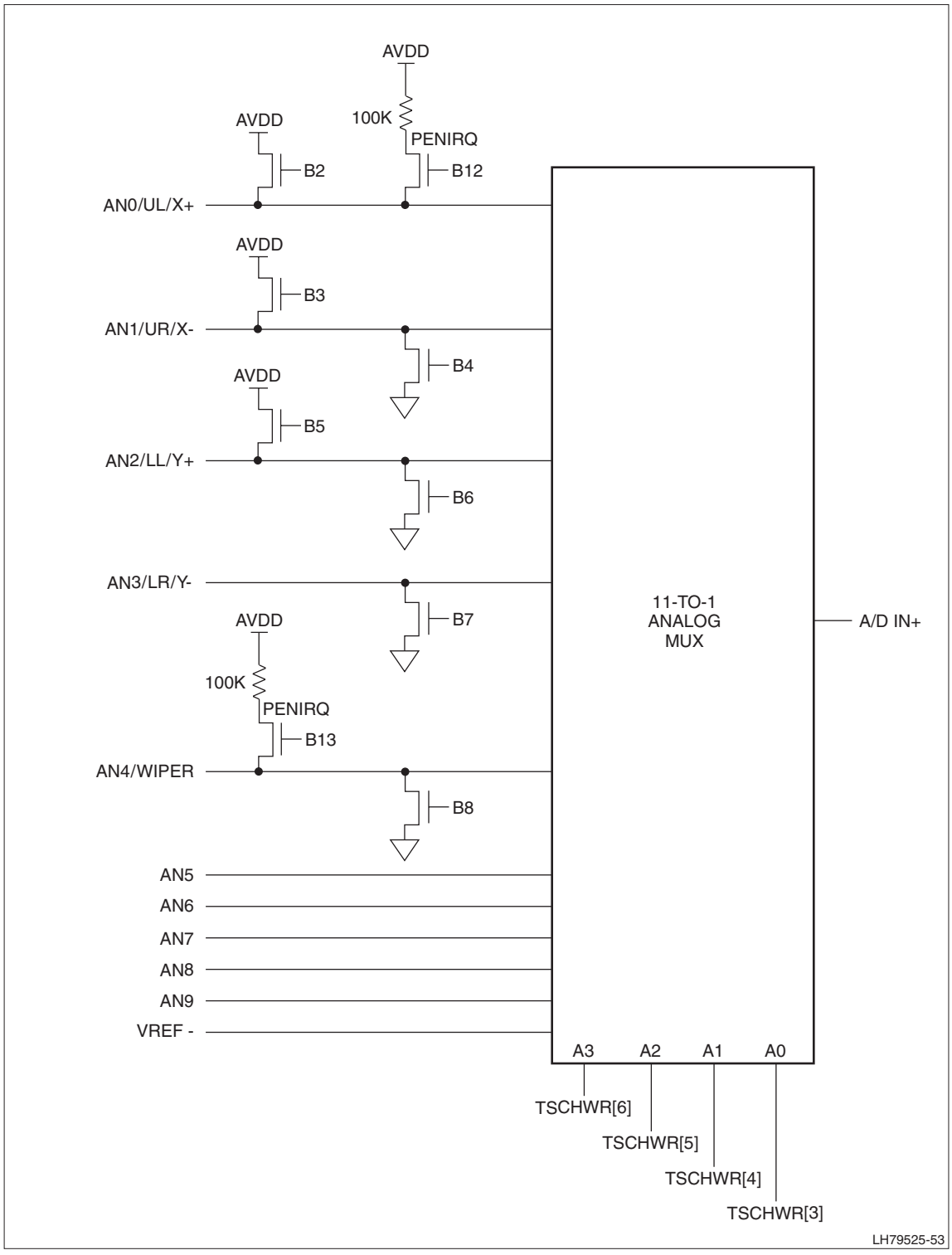


Figure 2-2. Bias-and-Control Network Block Diagram

2.1.3 Clock Generator

The ADC has a programmable measurement clock derived from the ADC peripheral clock generated by the RCPC. The clock source is selectable from HCLK or the System oscillator clock, and can be prescaled. The clock supplies the time base for the measurement sequencer and the successive-approximation circuitry. Higher clock frequencies allow faster measurement throughput. Slower clock frequencies allow more settling time for a measurement and can reduce ADC power consumption. If the clock is too slow, the sample-and-hold amplifier on the ADC input may droop before the measurement is complete.

2.1.4 Brownout Detector

The Brownout Detector is an asynchronous comparator that compares a divided version of the 3.3 V supply and a bandgap-derived reference voltage. If the supply dips below a trip point, the Brownout Detector sets a bit in the IS Register (see Section 2.2.2.8). An interrupt is directly connected to the VIC. This allows the MCU to notify peripherals of an impending shutdown and provides the ADC with time to save its state.

The Brownout detector also indicates brownout if the clock is off (PWM bits of PC register are 0b00 or 0b11). In addition, the Brownout Detector indicates a brownout condition on startup until the VDDA pin rises above the trip point.

2.1.5 SAR Architecture

While there are various SAR implementations, the basic architecture is simple. Figure 2-3 shows this architecture.

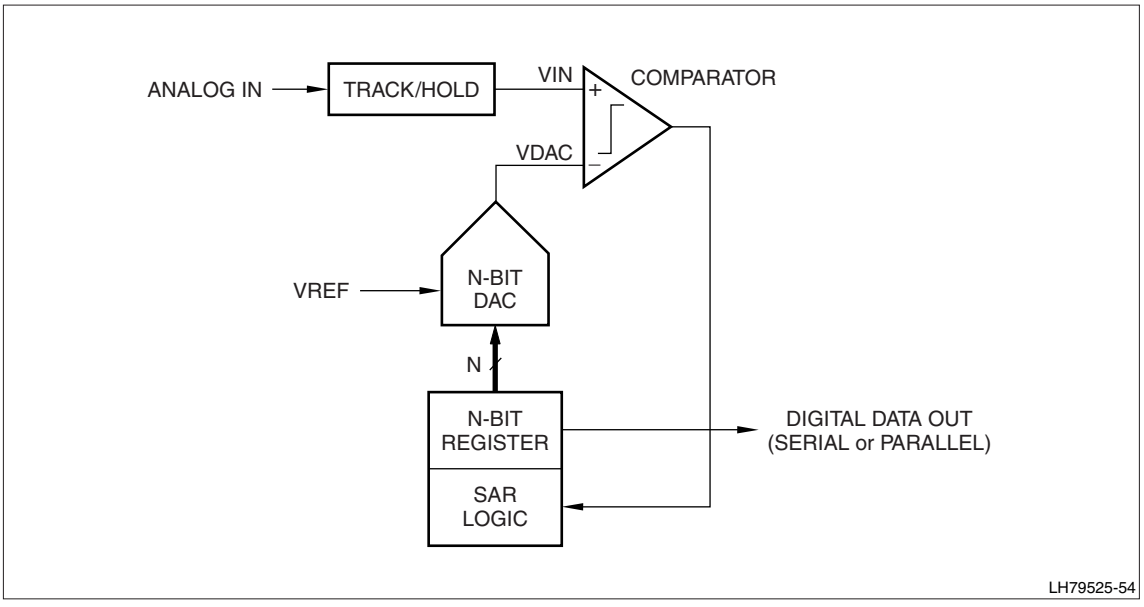


Figure 2-3. Simplified N-bit SAR Architecture

The analog input voltage (VIN) is held on a track/hold. The N-bit register is set to midscale (100...0, where the most-significant bit is set to 1) to implement the binary search algorithm. This forces the DAC output (VDAC) to be VREF ÷ 2, where VREF is the reference voltage provided to the ADC. Then a comparison is performed to determine whether VIN is less than, or greater than VDAC:

- If VIN is less than VDAC, the comparator output is a logic LOW and the most-significant bit of the N-bit register is cleared to 0.
- If VIN is greater than VDAC, the comparator output is a logic HIGH (or 1) and the most-significant bit of the N-bit register remains set to 1.

The SAR control logic then moves to the next bit down, forces that bit HIGH, and conducts another comparison. The SAR control logic repeats this sequence until it reaches the least-significant bit. When the conversion is complete, the N-bit digital word is available in the register.

Figure 2-4 shows an example of a 4-bit conversion. In this figure, the y-axis and the bold line show the DAC output voltage. In this example:

1. The first comparison shows that VIN < VDAC. Consequently, bit 3 is 0. The DAC is then set to ob0100 and the second comparison is conducted.
2. In the second comparison, VIN > VDAC, so bit 2 remains at 1. The DAC is then set to 0b0110 and the third comparison is conducted.
3. In the third comparison, bit [1] is set to 0 and the DAC is then set to 0b0101 for the last comparison.
4. In the final comparison, bit 0 remains at 1 because VIN > VDAC.

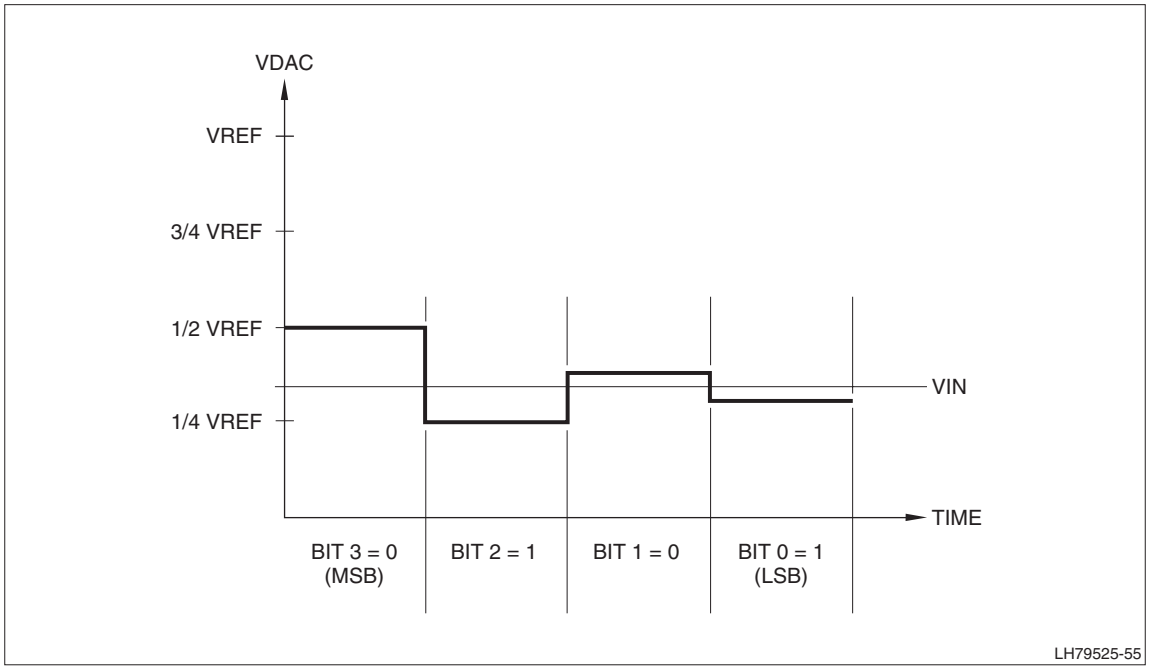


Figure 2-4. Example of a 4-bit SAR ADC Operation

Four comparison periods are necessary for a 4-bit ADC. Generally, an N-bit SAR ADC requires N comparison periods and will not be ready for the next conversion until the current conversion is completed.

Another feature of SAR ADCs is that power dissipation scales with the sample rate. By comparison, flash or pipelined ADCs usually have constant power dissipation as opposed to sample rate. This SAR ADC feature is especially useful in low-power applications or applications where data acquisition is not continuous.

2.1.6 Battery Control Feature

The battery control pin (BATCNTL) allows control of external battery circuits by the MCU. An external resistor divider allows monitoring the external battery voltage, as shown in Figure 2-5. External switches Q1 and Q2 connect the voltage divider to the battery. These switches are driven by the BATCNTL pin, which remains HIGH for the duration of the measurement. R1 and R2 should be chosen so that during normal operation (with BATCNTL LOW), the voltage at VBAT is somewhere within the common mode input range of the ADC. For example if the battery voltage is 6 V nominal, choose R1= 300 kΩ and R2 = 100 kΩ to give IN+ = 1.5 V at a load of only 15 μA on the battery.

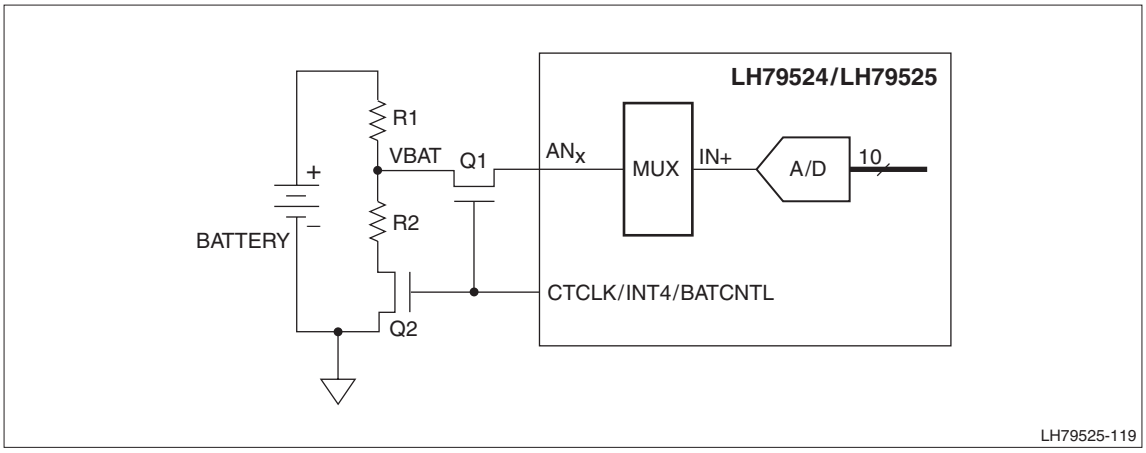


Figure 2-5. Use of the BATCNTL Pin

Note that the BATCNTL pin is *only* active when making the measurement on that particular ADC channel. All other times BATCNTL is LOW.

Software can easily configure the ADC for battery voltage measurement. Program the PC:BATLOC field to correspond to the channel to which the battery voltage measurement is connected. When HW:INP (the + Input Mux selection) equals the value in PC:BATLOC, the BATCNTL pin goes HIGH and loads the battery through the external switch as shown in Figure 2-5. In addition, PC:BATEN, the Battery Control Enable signal, must be programmed to 1. When the BATCNTL pin is not required to go HIGH, for example when the ADC is used in a general purpose application, PC:BATLOC should be programmed so that it will never equal any of the inputs that will be used in the application e.g. 0b1111. Also the PC:BATEN can be programmed to 0 to disable this function.

2.1.7 Timing Formulas

The throughput-conversion time consists of one cycle of Get Data state added to 16 cycles of measurement. Starting from the Idle state, the time for a complete measurement sequence, in clock cycles, is calculated as:

$$1CIS + MS \times (TCT + STC) + 1CEOS$$

where:

- 1CIS is one cycle in Idle state
- MS is the number of measurements in the sequence
- TCT is the throughput conversion time of 17 cycles
- STC is the number of settling time cycles per measurement
- 1CEOS is one cycle in the End of Sequence state.

This equals:

- Two cycles, plus
- The number of measurements in sequence times, plus
- The throughput conversion time (17 cycles), plus
- The number of settling time cycles per measurement.

2.1.8 Interrupts

The ADC has five interrupts:

- Brownout Interrupt (BROWNOUTINTR)
- Pen Interrupt (PENIRQ)
- End of Sequence Interrupt
- FIFO Watermark Interrupt
- FIFO Overrun Interrupt

All five interrupts make up the combined interrupt TSCIRQ, and presented to the VIC.

Each of the five individual maskable interrupts, except Brownout, is enabled or disabled by changing the mask bits in the IM Register (see Section 2.2.2.4). Software can read the interrupt status bits through the IS Register, even if corresponding mask bits are set (see Section 2.2.2.8). Clearing the mask bits does not clear the interrupt status.

2.1.8.1 Brownout Interrupt

The Brownout Interrupt (BROWNOUTINTR) is asserted when the supply voltage dips below the trip-point voltage. This interrupt status is latched in the IS Register. It remains HIGH until the BOIC bit of the Interrupt Clear (IC) register is asserted. The instantaneous raw status of the BROWNOUTINTR is stored in the GS Register (see Section 2.2.2.7). The Brownout Interrupt has its own dedicated output to the VIC.

NOTE: The latency between clearing the latched Brownout Interrupt and the time when it can be set again is one A2DCLK cycle. Polled systems should use the unlatched Brown-Out Raw Interrupt Status bit (bit [9]) in the GS register instead of the latched interrupt status in the IS register.

2.1.8.2 Pen Interrupt

The Pen Interrupt (PENIRQ) is enabled when the settings on the bias switches are switched to the Pen Interrupt Mode configuration and the ADC is set up to trigger a measurement on PENIRQ. The Pen Interrupt is used by the TSC to start the state machine. The state machine may begin a sequence of conversions, depending on the contents of the General Configuration (GC) register, when a Pen Interrupt occurs (see Section 2.2.2.6). PENIRQ is latched and remains HIGH until the PENIC bit of the Interrupt Clear (IC) register is asserted (see Section 2.2.2.14). The latched value of the Pen Interrupt is stored in the Interrupt Status register. The instantaneous raw status of the Pen Interrupt is stored in the General Status (GS) register (see Section 2.2.2.7).

NOTE: If a measurement sequence is configured to keep the Touch Screen biased for Pen detect on every measurement, PENIRQ is not generated on every sequence. If, on the other hand, the Pen detect circuit is disconnected, there will be an edge every time the system enters Idle state.

2.1.8.3 End-of-Sequence Interrupt

The End-of-Sequence Interrupt occurs after the programmed number of conversions (NOC) occurs. After the ADC converts all the data for a given sequence of conversions, this interrupt goes HIGH. The End-of-Sequence Interrupt is latched and remains HIGH until the EOSINTC bit of the IC Register is set.

2.1.8.4 FIFO Watermark Interrupt

The FIFO Watermark Interrupt occurs when the number of entries in the FIFO is greater than or equal to the programmed watermark level FIFOWMK (GC Register, bits [6:3]). This interrupt clears when the FIFO contents falls below the watermark level.

2.1.8.5 FIFO Overrun Interrupt

The FIFO Overrun Interrupt occurs when the receiving logic tries to place data into the FIFO after the FIFO has been completely filled, exceeding the FIFO's maximum capacity of 16 entries. The interrupt is cleared when the FIFO is read.

2.1.9 Application Details

An application note entitled 'Using the NXP ADC with Resistive Touch Screens' is available from NXP that provides more detailed application information dealing with use and programming of the ADC.

2.2 Register Reference

This section provides the ADC and Brownout Detector register memory mapping and bit fields.

2.2.1 Memory Map

The base address for the ADC is 0xFFFC3000.

Table 2-1 Summarizes the ADC registers. Address offsets in the table are from the base address. All registers are little endian format.

Table 2-1. ADC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	HW	High Word Register
0x04	LW	Low Word Register
0x08	RR	Results Register
0x0C	IM	Interrupt Masking Register
0x10	PC	Power Configuration Register
0x14	GC	General Configuration Register
0x18	GS	General Status Register
0x1C	IS	Interrupt Status Register
0x20	FS	FIFO Status Register
0x24 - 0x60	HWCB0 - HWCB15	High Word Control Bank Registers
0x64 - 0xA0	LWCB0 - LWCB15	Low Word Control Bank Registers
0xA4	IHWCTRL	Idle High Word Registers
0xA8	ILWCTRL	Idle Low Word Registers
0xAC	MIS	Masked Interrupt Status
0xB0	IC	Interrupt Clear Register

2.2.2 Register Descriptions

2.2.2.1 High Word Register (HW)

HW is the High Word Register. This Read Only status register shows the contents of the current conversion's high word in the control bank. There is a one-to-one correspondence between the contents of the control bank high word and the contents of this register for the current conversion in progress.

Table 2-2. HW Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SETTIME									INP				INM	REFP	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x00															

Table 2-3. HW Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:7	SETTIME	**Number of Clock Cycles** Specifies the number of clock cycles that the ADC allows for the input signal to settle to within required accuracy before beginning conversion. Used with bits [10:8] of the PC Register to set the acquire time in clock cycles (see Section 2.2.2.5). For example, if Frequency In (fIN) = 2 MHz (500 ns period): PC[10:8] = 010 (i.e., divide fIN by 4) HW[15:6] = 000100000 (i.e., 32 cycles) Therefore, acquire time is 500 ns × 4 × 32 = 64 μs
6:3	INP	**In+ Mux** Determines the signal connected to the positive input of the ADC. See Table 2-4.
2	INM	**In- Mux** Determines the signal connected to the negative input of the ADC. 1 = GND 0 = Ref- (output of the Ref- Mux)
1:0	REFP	**Ref+ Mux** Determines the signal connected to the positive reference of the ADC. 00 = VREF+ (positive terminal of the internal bandgap reference) 01 = AN0 (UL/X+) 10 = AN2 (LL/Y+) 11 = AN8

Table 2-4. In + Mux Definition

IN+	BIT6	BIT5	BIT4	BIT3
AN0 (UL/X+)	0	0	0	0
AN1 (UR/X-)	0	0	0	1
AN2 (LL/Y+)	0	0	1	0
AN3 (LR/Y-)	0	0	1	1
AN4 (Wiper)	0	1	0	0
AN5	0	1	0	1
AN6	0	1	1	0
AN7	0	1	1	1
AN8	1	0	0	0
AN9	1	0	0	1
VREF -	1	0	1	0
VREF -	1	0	1	1
VREF -	1	1	0	0
VREF -	1	1	0	1
VREF -	1	1	1	0
VREF -	1	1	1	1

2.2.2.2 Low Word Register (LW)

LW is the Control Bank Low Word Register. This Read Only status register displays the contents of the current conversion's low word in the control bank. There is a one-to-one correspondence between the contents of the control bank low word and the contents of this register for the current conversion in progress.

Table 2-5. LW Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		BIASCON												REFM	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x04															

Table 2-6. LW Fields

BIT	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13:2	BIASCON	**Bias Control** These bits turn the FETs on and off, as shown in Figure 2-2. The bit number corresponds to the FET number in the figure. IMPORTANT: bits 9-11 must always be written as 0b000. Writing a 1 to any of these three bits can cause unpredictable results. 1 = FET ON 0 = FET OFF
1:0	REFM	**Ref- Mux** Determines the signal connected to the negative reference of the ADC during Idle Mode. 00 = VREF- (negative terminal of the internal bandgap reference). 01 = AN1 (UR/X-) 10 = AN3 (LR/Y-) 11 = AN9

2.2.2.3 Results Register (RR)

RR is the Results register. This register contains the oldest entry of the 16-entry × 16-bit wide result FIFO. Its index in the FIFO's memory array is contained in the Read Pointer (RDPTR) bit field in the FIFO Status Register (see Section 2.2.2.9). This register contains the 10-bit ADC output and the 4-bit tag number from the Control Bank State Machine. When the FIFO is full, further data writes are temporarily blocked until at least one location is available for a write. Reading from RR removes the oldest entry from the result FIFO and increments the RDPTR.

Table 2-7. RR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	ADCOUT										///		CBTAG			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x08															

Table 2-8. RR Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:6	ADCOUT	**ADC Output** Contains the 10-bit digital output of the ADC.
5:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	CBTAG	**Control Bank Tag** Specifies the entry number (HWCTRLBxx or LWCTRLBxx) of the Control bank. The entry number (x) ranges from 0 to 15, corresponding to the conversion associated with the bit result.

2.2.2.4 Interrupt Mask Register (IM)

IM is the Interrupt Mask /Enable register. The active bits used in this register are Read/ Write and enable the interrupts. Software can read the status of the interrupt bits through the IS Register, even if corresponding mask bits are set in this register. The Brown Out enable is unique in that the Brown Out Interrupt can be programmed to be either an FIQ or an IRQ. That programming is done in this register. The Interrupt Status (IS) and Masked Interrupt Status (MIS) registers show only the status of the interrupt, not how it is configured. Current configuration can be read from this register. Writing a 0 to an IM bit does not clear the latched interrupt status in the IS register. The IS register is logically ANDed with the IE register to create the contents of the Masked Interrupt Status (MIS) register.

Table 2-9. IM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									INTEN	///	BOIRQ	PMSK	EOSMSK	FWMSK	FOMSK
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC3000 + 0x0C															

Table 2-10. IM Fields

BIT	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	INTEN	**Interrupt Enable** 1 = Global IRQ interrupts enabled 0 = Global IRQ interrupts masked
5	///	**Reserved** Reading returns 0. Write the reset value.
4	BOIRQ	**Brown Out IRQ Enable** Enabling this bit allows the brownout detector to generate an interrupt request as part of the combined TSCINTR input of the VIC. 1 = Enable Brown Out IRQ to VIC 0 = Disable Brown Out IRQ to VIC
3	PMSK	**Pen IRQ Interrupt Enable** 1 = Pen IRQ enabled 0 = Pen IRQ masked
2	EOSMSK	**End-of-Sequence Interrupt Enable** 1 = EOS IRQ enabled 0 = EOS IRQ masked
1	FWMSK	**FIFO Watermark Interrupt Enable** 1 = FIFO Watermark IRQ enabled 0 = FIFO Watermark IRQ masked
0	FOMSK	**FIFO Overrun Interrupt Enable** 1 = FIFO Overrun IRQ enabled 0 = FIFO Overrun IRQ masked

2.2.2.5 Power Configuration Register (PC)

In this register, the clock divider bits are programmed to set the system clock frequency for analog operation. Program bits [3:0] to the number of conversions necessary, depending on the conversion. Bit [4] can be used as an enable for external I/O pads. If this bit is set to 1, the Battery Control Logic Pin (BATCNTL) will be a valid output. If an external battery measurement circuit is not used, this bit should be set to 0.

NOTE: Allow two A2DCLK cycles between successive write cycles to this register. Otherwise, ADC behavior can become erratic.

Table 2-11. PC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	BATLOC					CLKSEL			PWM		REFEN	BATEN	NOC			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC3000 + 0x10															

Table 2-12. PC Fields

BIT	NAME	DESCRIPTION
31:15	///	**Reserved** Reading returns 0. Write the reset value.
14:11	BATLOC	**Battery Measurement Location** Program this field with the Channel number corresponding to the location programmed into the HW:INP field for battery measurement. When PC:BATLOC = HW:INP, the output pin BATCNTL goes HIGH. At all other times, the BATCNTL pin is LOW. To disable toggling the BATCNTL pin, program these bits to a value that will never appear in HW:INP, for example 0b1111.
10:8	CLKSEL	**Clock Select** If the nominal value is used, the only valid settings are 011, 100, 101, and 110. 000 = Clock oscillator (nominally 10 to 20 MHz) 001 = Clock oscillator / 2 010 = Clock oscillator / 4 011 = Clock oscillator / 8 100 = Clock oscillator / 16 101 = Clock oscillator / 32 110 = Clock oscillator / 64 111 = Clock oscillator / 128

Table 2-12. PC Fields (Cont'd)

BIT	NAME	DESCRIPTION
7:6	PWM	**Touch Screen Controller Power Mode** Tis field also affects the of the A2DCLK, Band Gap, and A2D signals (see Table 2-13). 00 = Turns off Power Mode and clock; sets the BROWNOUT field (bit [9]) of the GS Register, indicating that a brownout is detected, even if VDDA_ADC is at the correct voltage. 01 = Standby (wake on SSB or Pen Interrupt, convert, return); clears the GS:BROWNOUT bit, even if VDDA_ADC is correct voltage. 10 = Run (always on); clears the BROWNOUT field (bit [9]) of the GS Register, even if VDDA_ADC is at the correct voltage. 11 = Turns off Power Mode and clock; sets the BROWNOUT field (bit [9]) of the GS Register, indicating that a brownout is detected, even if VDDA_ADC is at the correct voltage.
5	REFEN	**Reference Enable** Enables the internal reference buffer so that the ADC can use the on-chip reference as the positive reference. 1 = Enable 0 = Disable
4	BATEN	**Battery Control Enable** 1 = Battery Control Logic Enabled 0 = Battery Control Logic Disabled
3:0	NOC	**Number of Conversions (NOC) in Sequence** Actual number of conversions is NOC + 1, and ranges from 1 to 16.

Table 2-13. Touch Screen Controller Power Modes

PWM BIT VALUES	nIDLE[1]	A2DCLK ENABLE[2]	BANDGAPON[3]	A2DON[4]
00	0	0	0	0
00	1	0	0	0
01	0	0	1	0
01	1	1	1	1
10	0	0	1	1
10	1	1	1	1
11	0	0	0	0
11	1	0	0	0

NOTES:
1. nIDLE refers to whether the state machine is in the Idle state:
   1 = Control Bank State Machine is in another state besides the Idle state.
   0 = Control Bank State Machine is in the Idle state.
2. A2DCLK ENABLE refers to whether the A2DCLK signal is enabled:
   1 = Enables the A2DCLK to the analog circuitry.
   0 = Disables the A2DCLK to the analog circuitry. (The clock is always enabled to the digital circuitry.)
3. BANDGAPON refers to whether Band Gap is turned on (required for the Brownout Detector):
   1 = Turns on the Band Gap. This setting is required for the Brownout Detector to work.
   0 = Turns off the Band Gap, disabling the Brownout Detector.
4. A2DON refers to whether the analog circuitry is enabled for the ADC:
   1 = Enables the analog circuitry for the ADC.
   0 = Disables the analog circuitry for the ADC.

2.2.2.6 General Configuration Register (GC)

In this register, the SSM field triggers the state machine to retrieve the data from the Control Bank and store it in the appropriate registers for the ADC. If the SSM bits are set to 0b11 at the end of a sequence, the state machine continues to convert data.

If the SSM bits are set to 0b10 and a value of 0b0000110 is written to the GC Register, the EOS_UM bit (bit [2]) of the Interrupt Status Register may never get set. This is normal operation. To accommodate this, wait two A2DCLK periods after setting the SSM bit to 10 before setting the SSB bit.

NOTE: Allow two A2DCLK cycles between successive write cycles to this register. Otherwise, ADC behavior can become erratic.

Table 2-14. GC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									FIFOWMK				SSB	SSM	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC3000 + 0x14															

Table 2-15. GC Fields

BIT	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6:3	FIFOWMK	**FIFO Watermark** Programmed to values between 0 and 15. This value corresponds to watermark levels between 1 and 16, respectively. When the FIFO fills to this level, the FIFO generates an interrupt.
2	SSB	**Start Sequence Bit** 1 = SSB will start the conversion sequence 0 = SSB will not start the conversion sequence
1:0	SSM	**Sequence Start Mode** To trigger continuous conversions, set these bits to 0b11, wait one A2DCLK period, and set the SSB bit to 1. Thereafter, once any conversions occur and SSM is set to 0b00 to stop the conversions, conversions can be started again by setting SSM to 0b11, without having to set SSB. Note that the Pen Interrupt can only be used when the ADC is configured to start on Pen Down. 00 = SSB or Pen Interrupt starts new conversions 01 = Pen Interrupt starts new conversions 10 = SSB starts new conversions 11 = Continuous conversions

2.2.2.7 General Status Register (GS)

GS is the General Status Register. In this Read Only register, the 4-bit signal CBSTATE field shows the current state of the Control Bank state machine. The CBTAG signal contains the control bank entry number of the conversion that is taking place.

Table 2-16. GS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						BRONOUT	PENIRQ	CBSTATE				CBTAG			
RESET	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x18															

Table 2-17. GS Fields

BIT	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9	BROWNOUT	**Brown-Out Raw Interrupt Status** 1 = Brown-out Interrupt is active. 0 = Brown-out Interrupt is not active.
8	PENIRQ*	**Pen IRQ Raw Interrupt Status** 1 = Pen IRQ Interrupt is active. 0 = Pen IRQ Interrupt is not active.
7:4	CBSTATE	**Control Bank State Machine Status** The only valid values are: 0001 = Idle state; waiting for sequence start trigger 0010 = GET_DATA state 0100 = WAIT_CONV state 1000 = END_OF_SEQ state
3:0	CBTAG	**Current Conversion Tag Number** Contains the current conversion tag number.

NOTE: *If the Idle state is configured to bias a 4-wire Touch Screen for Pen IRQ detect, the PENIRQ bit is only set during the one A2DCLK period of the GET_DATA state. To determine if the pen has been down at all, examine the IS:PENSYNC_UM bit. To determine if the pen was down at both the start and the end of a measurement sequence, use analog measurements of the pen IRQ voltage at the beginning and the end of the sequence. Then have a software Schmidt Trigger verify the logic level at the beginning and end of the coordinate-measurement sequence. Install a Pen IRQ handler function that changes the measurement mode to software triggered and disables Pen IRQ interrupts. Then stop the timer when it expires. Next, install an end-of-sequence interrupt handler that reads the measurement results and determines whether the pen is still down. If the pen is down, the handler starts the timer for triggering the next measurement. The handler discards the first set of measurements taken during the initial Pen Down detection. Otherwise, the handler posts the current pen position to some sort of OS queue. Enable Pen Triggered Measurements to start the system.

2.2.2.8 Interrupt Status Register (IS)

IS is the Interrupt Status register. This Read Only register provides the unmasked value of each interrupt. The BROWNOUT, PENSYNC, and EOS interrupts are latched and must be cleared by writing to the Interrupt Clear (IC) register. The FWATER and FOVRN interrupts are cleared when the contents of the FIFO no longer exceed their thresholds.

Table 2-18. IS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											BROWNOUT_UM	PENSYNC_UM	EOS_UM	FWATER_UM	FOVRN_UM
RESET	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x1C															

Table 2-19. IS Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	BROWNOUT_UM	**Unmasked Brown-Out Interrupt Status** 1 = Brown-out Interrupt is active. 0 = Brown-out Interrupt is not active.
3	PENSYNC_UM	**Unmasked Pen Interrupt Status** 1 = Pen Interrupt is active. 0 = Pen Interrupt is not active.
2	EOS_UM	**Unmasked End-of-Sequence Interrupt Active** 1 = EOCIA Interrupt is active. 0 = EOCIA Interrupt is not active.
1	FWATER_UM	**Unmasked FIFO Watermark Interrupt Active** 1 = FIFO Watermark Interrupt is active. 0 = FIFO Watermark Interrupt is not active.
0	FOVRN_UM	**Unmasked FIFO Overrun Interrupt Active** 1 = FIFO Overrun Interrupt is active. 0 = FIFO Overrun Interrupt is not active.

2.2.2.9 FIFO Status Register (FS)

FS is the FIFO Status Register. This Read Only register indicates the FIFO fill status.

Table 2-20. FS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				WRPTR				RDPTR				FFF	FEMPTY	FOVRNDET	FGTEWATERMRK
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0x20															

Table 2-21. FS Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11:8	WRPTR	**Write Pointer FIFO Location** Contains the index of the memory location in the result FIFO array where the next measurement result will be stored.
7:4	RDPTR	**Read Pointer FIFO Location** Contains the index to the location in the result FIFO array where the next measurement result will be read. Reads from the RR register increment this value.
3	FFF	**FIFO Full** 1 = FIFO is full 0 = FIFO is not full
2	FEMPTY	**FIFO Empty** 1 = FIFO is empty 0 = FIFO is not empty
1	FOVRNDET	**FIFO Overrun Status Bit** This bit is 1 when the receive logic tries to place data into the FIFO after it has been completely filled. When new data is received, the FOVRNDET bit is asserted and the newly received data is discarded. This process repeats for each time new data is received, until at least one empty FIFO entry exists. When FOVRNDET is set to 1, an interrupt request is generated. 1 = Logic tried to place data into a full receive FIFO and is requesting an interrupt 0 = FIFO has not experienced an overrun
0	FGTEWATERMRK	**FIFO at Watermark** 1 = FIFO is at or above watermark level 0 = FIFO has fewer entries than the watermark level

2.2.2.10 Control Bank Registers

The Control Bank is a set of 32 16-bit registers. The contents of the registers controls the switches for the ADC. These registers are typically configured once at startup, dictated by the physical system. HWCTRLBx and LWCTRLBx are used together and follow the format of the HW and LW Registers (see Section 2.2.2.1 and Section 2.2.2.2).

HWCTRLB0 (Tag 0b0000 of the High Word Control Bank) contains 16 bits of data for the 4WX (4 wire touch screen, X direction) conversion. The remaining 14 bits of data for the 4WX conversion is in LWCTRLB0 (Tag number 0b0000 of the Low Word Control Bank). Bits 15 and 14 of the low words are reserved and read as zero. The same logic is used for 4WY (4 wire touch screen, Y direction).

The same logic is used for the Control Bank Registers HWCBx and LWCBx. The High Word Registers should contain:

- The settling time
- The In+ bits
- The In- bits
- The Ref+ bits.

The Low Word Registers should contain:

- The bias control settings
- The Ref- bits.

At the end of any given conversion, a 4-digit Tag Number is stored in the FIFO along with the corresponding 10-bit output of the ADC.

For internal access into the control bank, the data writes to the registers from the APB data bus. Each entry is a 16-bit register, with its own address space. Table 2-22 shows sample entries for the Control Bank. More details, and examples can be found in NXP's Application Note 'Using the NXP ADC with Resistive Touch Screens', available at http://www.nxp.com.

Table 2-22. Sample Entries for Control Bank

TAG TYPE	TAG NAME	CONTENTS			
4WX	HWCBx0	Settling Time[15:7]	In+ [6:3]	In- [2]	Ref+ [1:0]
4WY	HWCBx1	Settling Time[15:7]	In+ [6:3]	In- [2]	Ref+ [1:0]
	HWCBx2...HWCBx15				
4WX	LWCBx0	Bias control[13:2]			Ref- [1:0]
4WY	LWCBx1	Bias Control[13:2]			Ref- [1:0]

2.2.2.11 Idle High Word Register (IHWCTRL)

IHWCTRL is the high word of the Idle Register. The active bits used in this register are Read/Write.

This register specifies the idle setting time and the inputs connected to the ADC during the Idle state. This register is used with the ILWCTRL Register (see Section 2.2.2.12).

Table 2-23. IHWCTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SETTIME_ID									INP_ID				INM_ID	REFP_ID	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC3000 + 0xA4															

Table 2-24. IHWCTRL Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:7	SETTIME_ID	**Idle Settling Time** Specifies the delay, in ADC clock cycles, from when the state machine enters the Idle state to when the Pen Interrupt signal can be activated. Prevents spurious trigger of Pen Interrupt while analog signals set up by the IDLE Register are settling.
6:3	INP_ID	**Idle In+ Mux** Specifies the connection to the positive input of the ADC during Idle Mode. See Table 2-4.
2	INM_ID	**Idle In- Mux** Specifies the connection to the negative input of the ADC during Idle Mode. 1 = GND 0 = Ref-
1:0	REFP_ID	**Idle Ref+ Mux** Specifies the connection to the positive reference of the ADC during Idle Mode. 00 = VREF+ 01 = AN0/UL/X+ 10 = AN2/LL/Y+ 11 = AN8

2.2.2.12 Idle Low Word Register (ILWCTRL)

ILWCTRL is the low word of the Idle Register. The active bits used in this register are Read/Write.

This register specifies the inputs connected to the ADC during the Idle state. This register is used with the IHWCTRL Register (see Section 2.2.2.11).

Table 2-25. ILWCTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		BIASCON_ID												REFM_ID	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC3000 + 0xA8															

Table 2-26. ILWCTRL Fields

BIT	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13:2	BIASCON_ID	**Idle Bias Control** These bits turn the FETs on and off, as shown in Figure 2-2. The bit number corresponds to the FET number in the figure. 1 = FET on 0 = FET off
1:0	REFM_ID	**Idle Ref- Mux** Specifies the connection to the negative reference of the ADC during Idle Mode. 00 = VREF- 01 = AN1 (UR/X-) 10 = AN3 (LR/Y-) 11 = AN9

IMPORTANT: Bits 9-11 must always be written as 0b000. Writing a 1 to any of these three bits can cause unpredictable results.

2.2.2.13 Masked Interrupt Status Register (MIS)

MIS is the Masked Interrupt Status register. This Read Only register gives the masked value of each interrupt. The BROWNOUT, PENSYNC, and EOS interrupts are latched and must be cleared by writing to the Interrupt Clear (IC) register. The FWATER and FOVRN interrupts are cleared when the contents of the FIFO no longer exceed their thresholds.

Table 2-27. MIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											BROWNOUT	PENSYNC	EOSINTR	FWATERINTR	FOVRNINTR
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC3000 + 0xAC															

Table 2-28. MIS Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	BROWNOUT	**Brown-Out Interrupt Status** 1 = Brown-out Interrupt is asserted 0 = Brown-out Interrupt is not active or not enabled
3	PENSYNC	**Pen Interrupt Status** 1 = Pen Interrupt is asserted 0 = Pen Interrupt is not active or not enabled
2	EOSINTR	**End-of-Sequence Interrupt Active** 1 = EOSIA Interrupt is asserted 0 = EOSIA Interrupt is not active or not enabled
1	FWATERINTR	**FIFO Watermark Interrupt Active** 1 = FIFO Watermark Interrupt is asserted 0 = FIFO Watermark Interrupt is not active or not enabled
0	FOVRNINTR	**FIFO Overrun Interrupt Active** 1 = FIFO Overrun Interrupt is asserted 0 = FIFO Overrun Interrupt is not active or not enabled

2.2.2.14 Interrupt Clear Register (IC)

IC is the Interrupt Clear Register. Bits [2:0] of this Write Only register correspond to the three latched interrupts.Writing a 1 to a bit clears the corresponding interrupt; writing a 0 to a bit has no effect. This register is self-clearing.

Table 2-29. IC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													BOIC	PENIC	EOSINTC
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFC3000 + 0xB0															

NOTE: The reset value of this register's bits is indeterminate.

Table 2-30. IC Fields

BITS	NAME	DESCRIPTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	BOIC	**Brown-Out Interrupt Clear** 1 = Clears BROWNOUTINTR. 0 = Do not clear BROWNOUTINTR
1	PENIC	**Pen Interrupt Clear** 1 = Clears PENIRQ 0 = Do not clear PENIRQ
0	EOSINTC	**End of Sequence Interrupt Clear** 1 = Clears EOSINTR 0 = Do not clear EOSINTR

Chapter 3
Boot Controller

The Boot Controller is the same for both the LH79524 and LH79525. All references in this chapter apply to both devices.

The Boot Controller provides a glueless interface to external NAND Flash devices and support for memory-mapped peripherals or NAND flash devices when performing AHB burst read accesses of undetermined length.

By monitoring external boot pins at power-on reset, the Boot Controller supports:

- Booting from 8-, 16-, or 32-bit memory
- Configuration of the byte-lane boot state for nCS1
- Booting from alternate external devices (e.g., NAND Flash, UART, I^2C).

Figure 3-1 shows the Boot Controller block diagram.

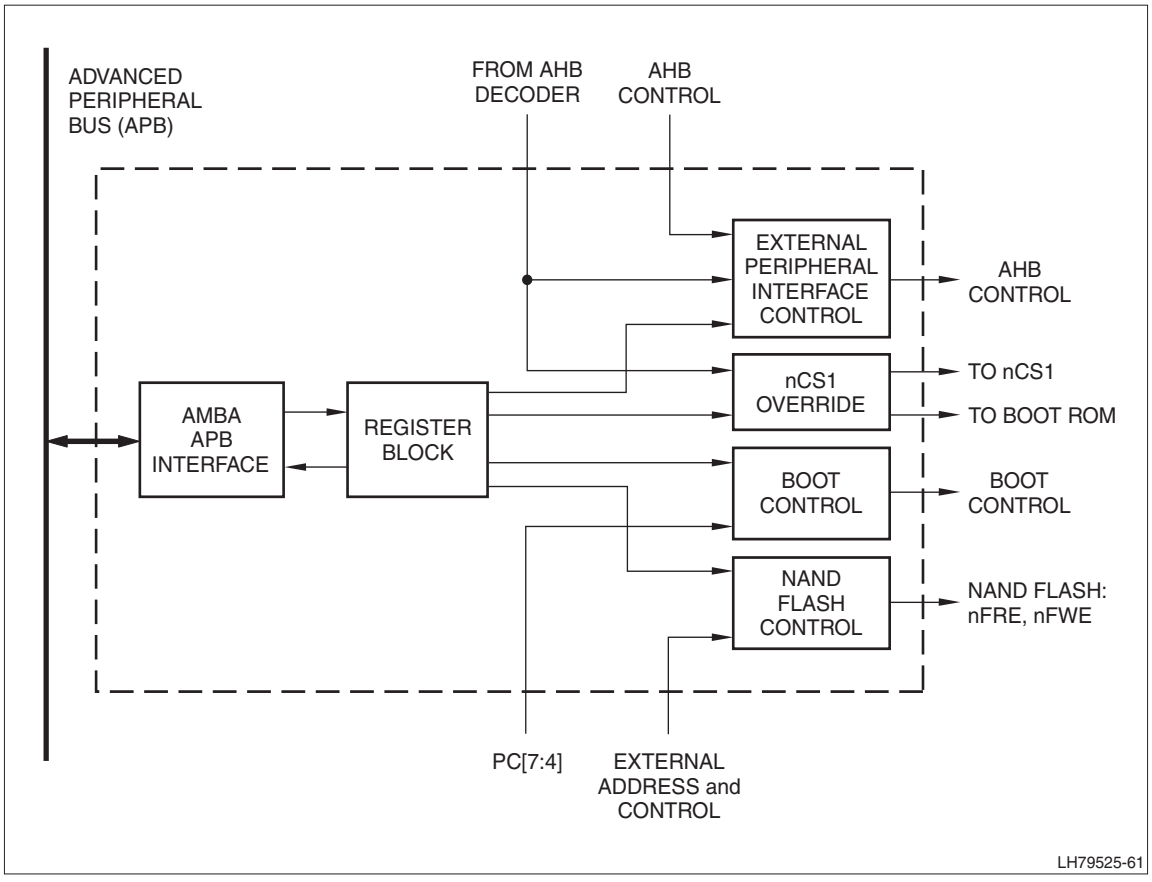


Figure 3-1. Boot Controller Block Diagram

3.1 Theory of Operation

The Boot Controller is a slave module that connects to the APB. It provides hardware support for configuring the External Memory Controller (EMC) interface on power-up, and allows multiple boot devices and scenarios to be used in different applications. The Boot Controller employs no error checking other than that specified by a protocol, if applicable, and does not utilize the MMU or caches.

Booting can occur from one of several devices. The Boot Controller reads the status of Port C and determines the type of device from which the code will be transferred. Once the device and location is determined, the Boot Controller reads exactly 4KB (small-block devices, I^2C, or UART) or 1KB (large-block devices) of code from that location and stores it at physical address 0x60000000. Finally, the Boot Controller transfers control to that code by setting the Program Counter to 0x60000000 and removes the Boot ROM from the memory map (the Boot ROM is only visible in the memory map immediatly following reset).

When using small-block devices, the Boot Controller transfers 4KB of code from the boot device. With large-block devices, the Boot Controller transfers 1KB. This is because large-block devices send an ECC value at the end of each page, which would corrupt the code stream.

When using either type device, but especially with large-block devices, larger amounts of boot code can be loaded by writing a bootstrap loader for the initial code, which would then execute and transfer the balance of the boot code to internal SRAM.

3.1.1 Boot Device Determination

The Boot Controller determines type and location of an external non-volatile device from which boot code will be loaded for MCU core execution. This location must be within the nCS1 chip select domain for all devices but NAND Flash.

The booting process is controlled by the initial values of PC[7:4]. The value on these pins at power-up determines the boot device, data bus width, and configuration of control signals.

The Boot Controller then configures the External Memory Controller for proper accesses. Table 3-1 lists the configuration ONLY for version A.0 of the MCU. For versions A.1 and later, refer to Table 3-2.

Table 3-1. Boot Configuration for Silicon Version A.0

PC[7:4]	DEVICE TYPE	DATA BUS WIDTH	CONTROL
0x0	NOR Flash or SRAM	16-bit	nBLEx LOW for Reads
0x1	NOR Flash or SRAM	16-bit	nBLEx HIGH for Reads
0x2	NOR Flash or SRAM	8-bit	nBLEx LOW for Reads
0x3	NOR Flash or SRAM	8-bit	nBLEx HIGH for Reads
0x4	NAND Flash (Small Block)	8-bit	3-byte Address
0x5	NAND Flash (Small Block)	8-bit	4-byte Address
0x6	NAND Flash (Small Block)	8-bit	5-byte Address
0x7	NAND Flash (Small Block)	16-bit	3-byte Address
0x8	NOR Flash or SRAM	32-bit	nBLEx LOW for Reads
0x9	NOR Flash or SRAM	32-bit	nBLEx HIGH for Reads
0xA	RESERVED	RESERVED	RESERVED
0xB	RESERVED	RESERVED	RESERVED
0xC	NAND Flash (Small Block)	16-bit	4-byte Address
0xD	NAND Flash (Small Block)	16-bit	5-byte Address
0xE	RESERVED	RESERVED	RESERVED
0xF	RESERVED	RESERVED	RESERVED

Table 3-2. Boot Configuration for Silicon Version A.1

PC[7:4]	DEVICE TYPE	DATA BUS WIDTH	CONTROL
0x0	NOR Flash or SRAM	16-bit	nBLEx LOW for Reads
0x1	NOR Flash or SRAM	16-bit	nBLEx HIGH for Reads
0x2	NOR Flash or SRAM	8-bit	nBLEx LOW for Reads
0x3	NOR Flash or SRAM	8-bit	nBLEx HIGH for Reads
0x4	NAND Flash (Small Block)	8-bit	3-byte Address
0x5	NAND Flash (Small Block)	8-bit	4-byte Address
0x6	NAND Flash (Large Block)	8-bit	4/5-byte Address
0x7	NAND Flash (Small Block)	16-bit	3-byte Address
0x8	NOR Flash or SRAM	32-bit	nBLEx LOW for Reads
0x9	NOR Flash or SRAM	32-bit	nBLEx HIGH for Reads
0xA	RESERVED	RESERVED	RESERVED
0xB	RESERVED	RESERVED	RESERVED
0xC	NAND Flash (Small Block)	16-bit	4-byte Address
0xD	NAND Flash (Large Block)	16-bit	4/5-byte Address
0xE	I^2C	—	—
0xF	UART	—	—

3.1.1.1 NAND Flash Operation

When NAND Flash is detected as the boot code source, the Boot Controller forces an override of nCS1. Instead, the Boot ROM is selected for the locations in the memory map where nCS1 is normally selected and the Boot Rom code is executed. The Boot ROM code manages the interface to the NAND Flash device. The nCS1 override can be disabled by writing a 0 to the nCS1 Override Control bit in the CS1OV register.

The Boot Controller generates the control signals on the nFRE and nFWE pins for external NAND Flash. nFRE is the active LOW signal to the NAND flash Read Enable pin. This signal is the External Memory Controller's nOE, enabled by the signal on external address pin A23. Note that the LH79524/LH79525 memory controller automatically indexes address signals on the address pins, depending on the width of the memory devices. For example, with 8-bit addressing, the A0 signal is presented on pin A0, and the A23 signal is presented on pin A23. For 16-bit memory, the memory controller automatically shifts the address one bit to the right so that all addresses fall on half-word boundaries. In this configuration, signal A1 is presented on pin A0, and pin A23 carries the A24 signal. Similarly, for 32-bit devices, A2 appears on pin A0, and A25 appears on pin A23. For more information on booting from NAND Flash, refer to Section 7.3.1 and Section 7.3.2.

nFRE is only active (i.e., LOW) when nOE is LOW and the A23 signal (for 8-bit), A24 signal (for 16-bit), or A25 (for 32-bit) is HIGH. nFWE is the active LOW signal to the NAND flash Write Enable pin. This signal is the External Memory Controller's nWE signal, enabled by address signal A23/A24/A25. nFWE is only active (i.e., LOW) when nWE is LOW and A23/A24/A25 is HIGH. Gating these signals allows normal memory and I/O accesses to other external devices to occur during extended NAND Flash accesses (when chip select is held active). These other devices must be mapped to external memory regions where the address signals are LOW.

3.1.2 Hardware Design Considerations

Using the Boot Controller dictates certain hardware considerations, especially when booting from NAND Flash.

3.1.2.1 Active Pullups To Signal Boot Mode

The boot mode — NOR Flash, NAND Flash, SRAM, I^2C, or UART — is selected by the value latched on the rising edge of the nRESETIN signal from the state of PC[7:4], shown in Table 3-1 and Table 3-2. PC[7:6] are used during NAND Flash booting as control signals, but PC[5:4] have no other use following the end of reset. Therefore, those two GPIO pins can be used during normal operation if an active pullup is used, gated by the nRESETOUT signal.

Figure 3-2 shows a schematic representation of one active pullup circuit. One circuit is required for each PCx pin to be pulled high during reset. nRESETOUT is presented to the Gate (pin 1) of the P-Channel FET. When active (LOW), nRESETout causes the transistor to turn on, and pull the PCx input HIGH. When nRESETOUT transitions from LOW to HIGH at the end of the reset period, the value on PC[7:4] is latched and the FET is turned off, thus allowing those pins to be used for general purpose I/O or as address pins A[21:20].

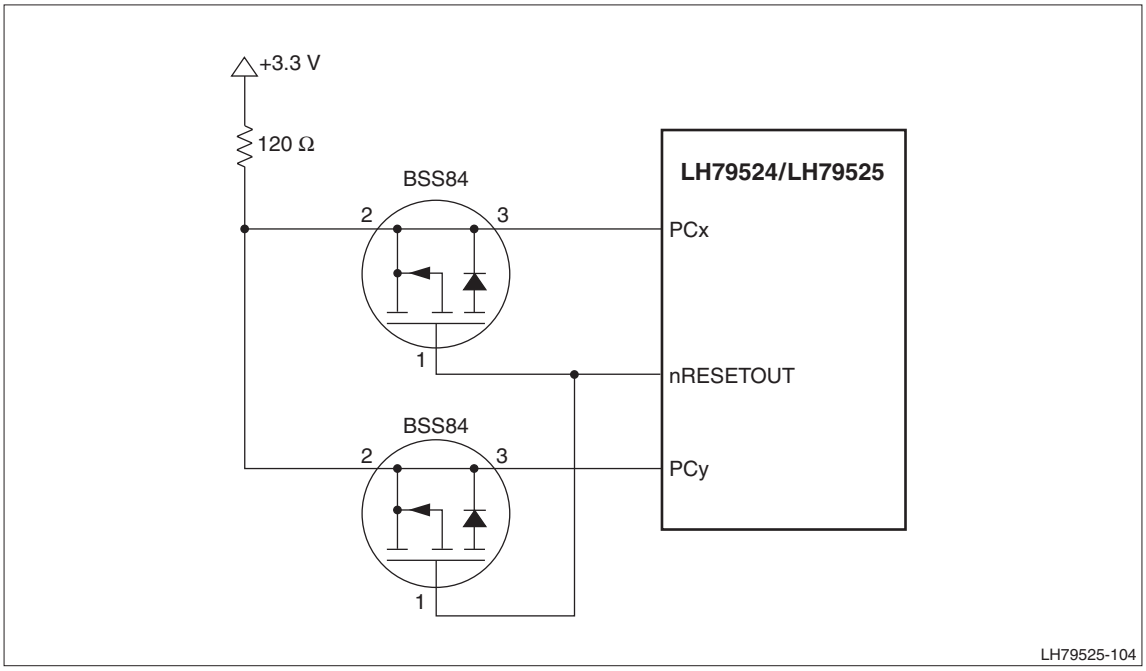


Figure 3-2. Active Pullup Circuit

3.1.2.2 NAND Flash Hardware Design

The additional NAND Flash control signals are multiplexed with Address lines. Table 3-3 shows the alternate pin functions when using NAND Flash devices.

These alternate pin functions must be considered when designing external interfaces. Further, the actual address signals presented on the A22, A23, A3 (ALE), and A4 (CLE) pins are determined by the memory width being addressed, as described in Section 3.1.2.

The A[4:3] address pins must be written with the correct address to make the ALE and CLE signals TRUE for the given transaction. These address values differ depending on the NAND Flash device width being used.

Table 3-3. Alternate Pin Function During NAND Flash Booting

PIN	PRIMARY FUNCTION	SECONDARY FUNCTION	NAND BOOT FUNCTION
PC6/A22/nFWE	PC6	A22	nFWE
PC7/A23/nFRE	PC7	A23	nFRE
A3	A3*	N/A	ALE
A4	A4*	N/A	CLE
nCS0	nCS0	N/A	nCE

NOTE: Pins A3 and A4 carry different address signals depending on the width of the memory device. For 8-bit devices, Pin A3= Address signal A3; Pin A4= Address signal A4. For 16-bit devices, Pin A3= Address signal A4; Pin A4= Address signal A5. For 32-bit devices, Pin A3= Address signal A5; Pin A4= Address signal A6. See Section 3.1 and Chapter 7: External Memory Controller.

3.1.2.2.1 NAND Flash Chip Select

Because of the hardware implementation of the NAND Flash signalling, the LH79524/ LH79525 chip select used for NAND Flash addressing *must* be nCS0 for booting; nCS1 cannot be used. Connect the nCS0 pin to the NAND Flash nCE input pin if that device is used for booting. If the NAND Flash is not used for booting, it can be located in any chip select domain.

3.1.3 Booting Using the I^2C Interface

Booting can also be done using the I^2C interface. When booting from I^2C, the device address that must be used is 0b1010000x. This address is not alterable. The Boot Controller will always boot exactly 4Kbytes when using the I2C serial EEPROM.

Interface parameters are shown in Table 3-4 and the list of supported devices is shown in Table 3-5.

Table 3-4. Boot Parameters for I^2C

PARAMETER	VALUE
Communication Speed	400 kHz
Mode of MCU	Master Mode
Addressing Mode	7 bit
I^2C EEPROM Configuration	Slave, addressed at 0b1010000x, where x=0 for Writes and x=1 for Reads

Table 3-5. Supported Devices

DENSITY	ATMEL	ST MICRO	MICROCHIP
32Kbit (4K × 8)	AT24C32	M24C32	24xx32
64Kbit (8K × 8)	AT24C32	M24C64	24xx64
128Kbit (16K × 8)	AT24C32	M24128	24xx128
256Kbit (32K × 8)	AT24C32	M24256	24xx256
512Kbit (64K × 8)	AT24C32	M24512	24xx512
1Mbit (128K × 8)	AT24C32	N/A	N/A

3.1.4 Booting from UART

Another boot option is to boot using UART0. The transfer protocol implementation is XMODEM with 128-byte packets. All UART0 parameters are summarized in Table 3-6.

The Boot Controller automatically handles initialization and setup of UART0; the source of the boot code must be compatible with the parameters in the table.

Table 3-6. UART0 Boot Parameters

PARAMETER	VALUE
Protocol	XMODEM Checksum
Bit Rate	115 kbps
Data Bits	8
Parity	None
Stop Bits	1
Packet Size	128 bytes

3.2 Register Reference

This section provides the Boot Controller register memory mapping and bit fields.

3.2.1 Memory Map

The base address for the Boot Controller is 0xFFFE6000. Table 3-7 summarizes the Boot Controller registers.

Table 3-7. Boot Controller Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	PBC	Power-up Boot Configuration Register
0x04	CS1OV	nCS1 Override Register
0x08	EPM	External Peripheral Mapping Register

3.2.2 Register Definitions

3.2.2.1 Power-up Boot Configuration Register (PBC)

Reading from the PBC register returns the value that the PC[7:4] pins were driven during a power-on reset. This value is used by software contained in the Boot ROM, as well as the Boot Controller, to determine the type and configuration of the external device from which the CPU is to boot.

Table 3-8. PBC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												PBC			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	PC[7:4]			
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE6000 + 0x00															

Table 3-9. PBC Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	PBC	**Power-up Boot Configuration** This field contains the value that the PC[7:4] pins were driven during power-on reset.

3.2.3 nCS1 Override Register (CS1OV)

Bit 0 in the CS1OV register programs the function of the nCS1 signal. This bit has different functions for read and write. Reading returns the nCS1 Override Enable current status. Writing programs the nCS1 Override Control function.

During normal boot from the internal Boot ROM, CS1OV is programmed to 1 and PBC is programmed to 0bX1XX, an override of nCS1 occurs and CS1O will read as 1, resulting in the Boot ROM using nCS1 as chip select. This override can be disabled by writing a 0 to CS1O, causing CS1O to read as 0.

If on system reset the boot configuration is set to 0bX0XX, nCS1 remains mapped as described above, CS1O has no effect on the memory map and CS1O will reads as 0.

The very last thing Boot ROM software should do before returning control to the operating system is to write a 0 to this register so that normal routing of the nCS1 signal occurs.

Table 3-10. CS1OV Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															CS1O
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	*
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW
ADDR	0xFFFE6000 + 0x04															

NOTE: *Resets to the value to which PC[6] is externally driven during power-on reset.

Table 3-11. CS1OV Fields

BITS	NAME	DESCRIPTION
31:1	///	**Reserved** Reading returns 0. Write the reset value.
0	CS1O	**Read: nCS1 Override Enable** 1 = nCS1 override is enabled. The override only occurs when PBC[3] = 1 and CS1O is programmed to 1. 0 = nCS1 override is disabled; nCS1 is routed for normal operation **Write: nCS1 Override Control** 1 = enable nCS1 override 0 = disable nCS1 override

3.2.4 External Peripheral Mapping Register (EPM)

This register determines which chip selects will have burst accesses to their address regions converted to a series of non-sequential transfers. The register provides individual selectability for each of nCS0, nCS1, nCS2, and nCS3. At reset, accesses to all four chip select regions have conversion enabled. This ensures that all external devices will be accessible following reset.

Table 3-12. EPM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												CS3EP	CS2EP	CS1EP	CS0EP
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE6000 + 0x08															

Table 3-13. EPM Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	CS3EP	**nCS3 Configured for External Peripherals** 1 = All burst accesses to nCS3 are converted to a series of non-sequential single transfers. 0 = Accesses to nCS3 are unaltered.
2	CS2EP	**nCS2 Configured for External Peripherals** 1 = All burst accesses to nCS2 are converted to a series of non-sequential single transfers. 0 = Accesses to nCS2 are unaltered.
1	CS1EP	**nCS1 Configured for External Peripherals** 1 = All burst accesses to nCS1 are converted to a series of non-sequential single transfers. 0 = Accesses to nCS1 are unaltered.
0	CS0EP	**nCS0 Configured for External Peripherals** 1 = All burst accesses to nCS0 are converted to a series of non-sequential single transfers. 0 = Accesses to nCS0 are unaltered.

Chapter 4
Color Liquid Crystal Display Controller

This chapter discusses the Color LCD Controller (CLCDC) and its Advanced LCD Interface Peripheral (ALI) for AD-TFT, HR-TFT panels, and any technology of panel compatible with this signal system. The ALI-specific description begins in Section 4.4. The only difference between the LH79524 CLCDC and the LH79525 CLCDC is the pixel bit depth. The LH79524 supports up to 16 bits-per-pixel (bpp) depth, and the LH79525 supports up to 12 bpp.

4.1 Introduction

The CLCDC provides all necessary control and data signals to interface the MCU directly to a variety of color and monochrome LCD panels, including STN and TFT panels. The ALI modifies the CLCDC output to allow the chip to connect directly to the Row and Column driver chips on superthin panels, including AD-TFT, HR-TFT, or any panel that supports this method of connection. Figure 4-1 shows a simplified diagram of the two controllers connected to the AHB, to the APB, and to each other.

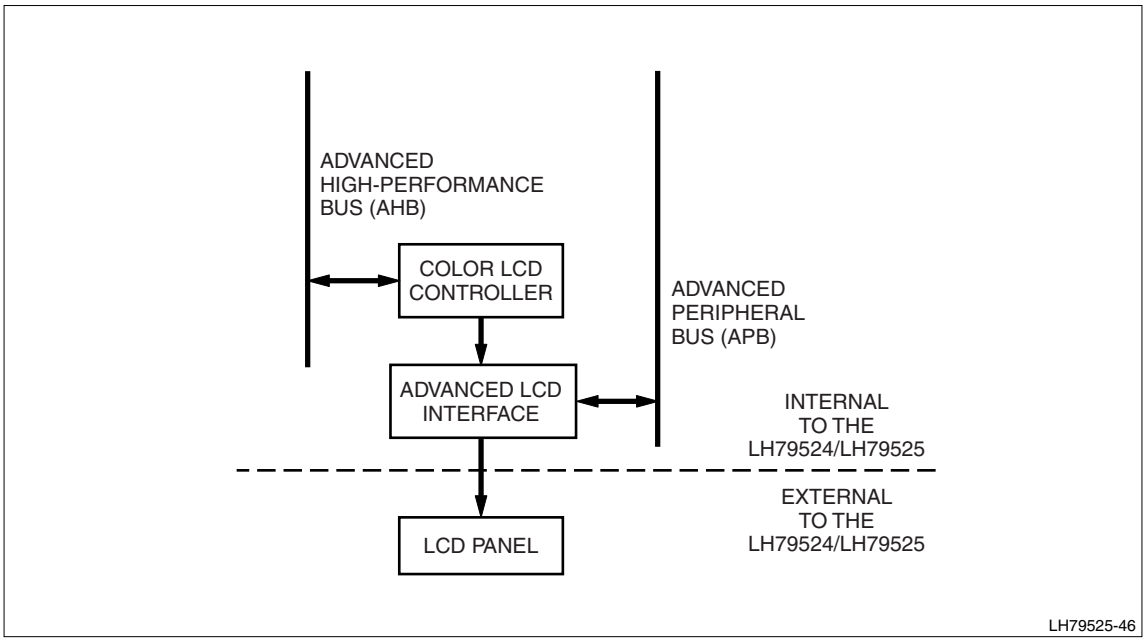


Figure 4-1. LH79524/LH79525 LCD System, Simplified Block Diagram

4.1.1 LCD Panel Architecture

Modern technology panels, including AD-TFT and HR-TFT panels, are thinner than ever. To achieve maximum space savings, they are manufactured without the large ASICs and DC-DC converter blocks built into STN and TFT panels. See Figure 4-2.

The ASIC in STN and TFT panels decodes input data into Row and Column information and builds the timing signals. It supplies this information to the panel's Row and Column driver chips to set the proper pixels at the proper intensity and at the proper times. The DC-DC converter runs the panel's power supplies and illuminator. Including these devices in STN and TFT panels, however, comes at the cost of bulk and weight.

The ALI eliminates the need for a separate Timing ASIC, since it is able to drive the panel's Row and Column driver chips directly. The DC-DC conversion is also handled off-panel, by a separate device operating the panel's high voltage supplies and illuminator. The DC-DC conversion must be handled by a separate device, since the LH79524/LH79525 do not supply this function.

Unless the behavior is different, this User's Guide uses the term TFT to discuss all types of TFT panels whether the panel requires timing support from the ALI or not.

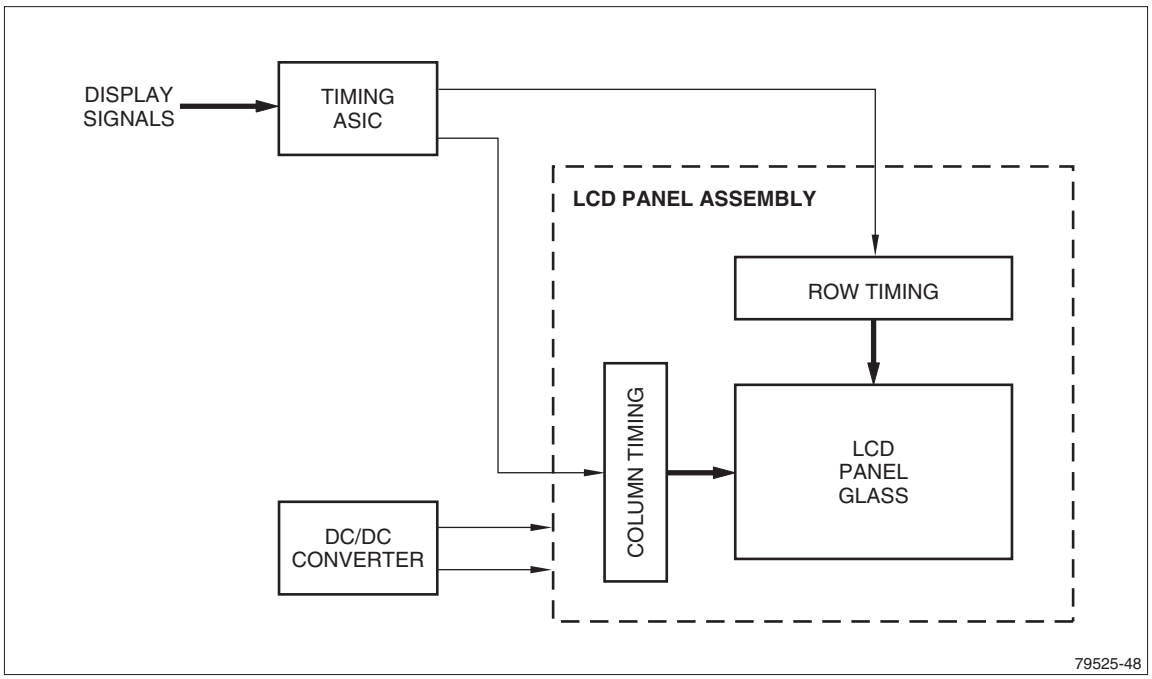


Figure 4-2. Block Diagram of a Typical Advanced LCD Panel

4.2 CLCDC Features

- Supported LCD Panels
  - Active-matrix TFT, HR-TFT and AD-TFT panels, with up to 16-bit bus interface (LH79524) or 12-bit bus (LH79525)
  - Single-panel monochrome STN panels, with 4-bit and 8-bit bus interfaces
  - Dual-panel monochrome STN panels, with 4-bit and 8-bit bus interface per panel
  - Single-panel color STN panels, with an 8-bit bus interface (LH79524 only)
  - Dual-panel color STN panels, with 8-bit bus interface per panel (LH79524 only)
- Resolution up to 1024 × 1024 dots per inch (DPI)
- Additional Features
  - Programmable timing for different display panels
  - 256-entry, 16-bit palette RAM physically arranged as a 128 × 32-bit RAM
  - AC bias signal for TFT panels and a data-enable signal for TFT panel

The following parameters can be programmed in the CLCDC:

- Horizontal front and back porch width
- Horizontal synchronization pulse width
- Number of pixels per line
- Vertical front and back porch width
- Vertical synchronization pulse width
- Number of horizontal lines per panel
- Number of panel data clocks per line
- Programmable signal polarities, active HIGH or active LOW
- AC panel bias
- Panel data clock frequency (LCDDCLK)
- Bits-per-pixel
- Little-endian, big-endian, and WinCE™ data formatting
- Interrupt generation.

4.3 Theory of Operation

The CLCDC is an AMBA master-slave module that connects to the AHB. Figure 4-3 is a detailed block diagram of the CLCDC. Packets of pixel-coded data are sent, via the AHB interface, to two independently programmable, 32-bit-wide DMA FIFOs. Each FIFO is 16 words deep by 32 bits wide. In Single Panel STN Mode, the LCD DMA FIFOs appear as a single FIFO of twice the size. The buffered pixel-coded data is then unpacked via a pixel serializer.

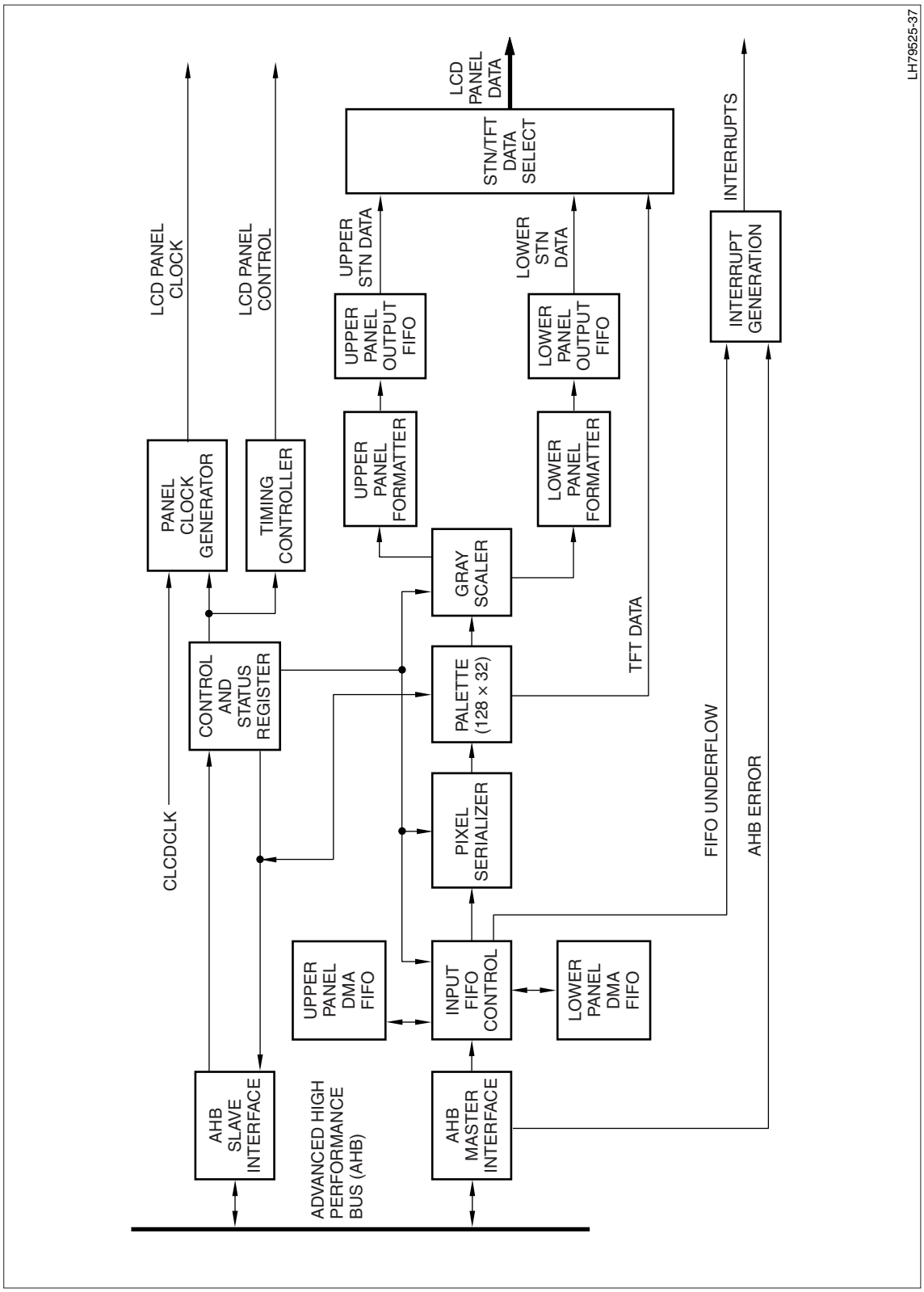


Figure 4-3. Color LCD Controller Block Diagram

In 12 or 16-bit-per-pixel Mode, the CLCDC uses the unpacked data directly to generate the pixel value. In all other bit-per-pixel modes, the CLCDC uses the unpacked data to index its palette RAM; the CLCDC uses the value indexed from the palette to generate the pixel value. For STN displays, this value passes to the gray-scaling generator. For TFT displays, this value bypasses the gray-scaling generator and goes directly to the output display drivers.

The CLCDC generates a single combined interrupt to the Vectored Interrupt Controller (VIC) when an individual interrupt condition becomes true for upper/lower panel DMA FIFO underflow, base address update signification, vertical compare, or bus error.

4.3.1 Supported Displays and Panels

The CLCDC retrieves image data from system memory (the frame buffer), formats the data for the LCD panel, and writes the data to the panel. The CLCDC also creates the control signals that cause the panel to display the formatted data.

The CLCDC translates pixel-coded data into the formats and timings required to drive:

- Single monochrome panels
- Dual monochrome panels
- Color LCD panels
- Super Twisted Nematic (STN) displays
- Active Thin Film Transistor (TFT) LCD displays.

4.3.2 Frame Buffer

A set of numbers representing the color or gray scale of each pixel the CLCDC displays is stored in a region of static memory called a frame buffer. The CLCDC uses its DMA Controller to fetch data from the frame buffer into its FIFO (or FIFOs for dual-panel interfaces) when the amount of pixel data falls below the FIFO watermark.

Data moves from the frame buffer into the FIFO via the AHB. The frame buffer can reside on either the external memory bus or on the internal SRAM.

4.3.3 LCD DMA FIFOs

The CLCDC has an upper LCD DMA FIFO and a lower LCD DMA FIFO. These FIFOs can be independently controlled to cover single- and dual-panel LCDs. Each FIFO is 16 words deep by 32 bits wide. In single-panel modes the LCD DMA FIFOs are made to appear as a single FIFO of twice the size.

The watermarks within each FIFO are set so that each FIFO requests data when the level of data in a FIFO falls below the programmed watermark (either four or eight locations, as specified by bit [16] of the LCD Panel Pixel Parameter Register). An interrupt signal is asserted, if enabled, if either of the two LCD DMA FIFOs is read when they are empty.

4.3.4 Pixel Serializer

The pixel serializer reads the 32-bit-wide LCD data from the output port of the LCD DMA FIFO and extracts 12, 8, 4, 2, or 1 bits per pixel (bpp) data, depending on the operating mode. In Dual Panel Mode, data alternately is read from the upper and lower LCD DMA FIFOs. Depending on the operating mode, the extracted data is either used to point to a color/grayscale value in the palette RAM or directly applied to an LCD panel input.

4.3.5 How Pixels are Stored in Memory

Table 4-1 shows the pixel arrangement on a display, with the first 32 pixels labeled p0 through p31. Table 4-2 and Table 4-3 show the data structure in each DMA FIFO word corresponding to the bpp combinations. The required data for each panel display pixel must be extracted from the data word. The first pixel value in the frame corresponds to the color value encoded in P0 (see Table 4-3), the second corresponds to P1, the third to P2, and so on (continuing in Table 4-2).

Table 4-1. Pixel Display Arrangement

p0	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10	p11	p12	p13	p14	p15	p16	p17	p18	p19	p20	p21	p22	p23	p24	p25	p26	p27	p28	p29	p30	p31

Table 4-2. Frame Buffer Pixel Storage Format [31:16]

BPP	DMA FIFO OUTPUT BITS															
1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	p31	p30	p29	p28	p27	p26	p25	p24	p23	p22	p21	p20	p19	p18	p17	p16
2	p15		p14		p13		p12		p11		p10		p9		p8	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p7				p6				p5				p4			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p3								p2							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
12[1]	p1															
		11	10	9	8		7	6	5	4		3	2	1	0	
16[2]	p1															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTES:
1. LH79525 with 12-Bit CLCDC
2. LH79524 with 16-Bit CLCDC

Table 4-3. Frame Buffer Pixel Storage Format [15:0]

BPP	DMA FIFO OUTPUT BITS															
1	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0
2	p7		p6		p5		p4		p3		p2		p1		p0	
	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	p3				p2				p1				p0			
	3	2	1	0	3	2	1	0	3	2	1	0	3	2	1	0
8	p1								p0							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
12[1]	p0															
		11	10	9	8		7	6	5	4		3	2	1	0	
16[2]	p0															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

NOTES:
1. LH79525 with 12-Bit CLCDC
2. LH79524 with 16-Bit CLCDC

4.3.6 Palette RAM

The CLCDC includes a 256 × 16 bit dual-port RAM-based palette. The least-significant bit of the serialized pixel data selects the upper or lower half of the palette RAM, based on the Byte Ordering Mode. Table 4-4 and Table 4-5 show the representations of each word.

- One port of the dual-port palette RAM is used as a Read/Write port and is connected to the AHB slave interface. Palette entries can be written and verified through this port.
- The second port of the dual-port palette RAM is used as a Read Only port and is connected to the unpacker and grayscaler.
- In 12 and 16 bpp TFT mode, the palette is bypassed and the output of the pixel serializer is used as the TFT panel data.

Table 4-4. Palette Data Storage (LH79525 with 12-Bit CLCDC)

BIT	NAME*	DESCRIPTION
31	///	Unused
30:27	MB[3:0]	Most Significant Blue palette data
26	///	Unused
25:22	MG[3:0]	Most Significant Green palette data
21	///	Unused
20:17	MR[3:0]	Most Significant Red palette data
16:15	///	Unused
14:11	LB[3:0]	Least Significant Blue palette data
10	///	Unused
9:6	LG[3:0]	Least Significant Green palette data
5	///	Unused
4:1	LR[3:0]	Least Significant Red palette data
0	///	Unused

NOTE: ***Blue and red palette data can be swapped by programming CTRL:BGR.**

Table 4-5. Palette Data Storage (LH79524 with 16-Bit CLCDC)

BIT	NAME*	DESCRIPTION
31	I	Intensity
30:26	MB[4:0]	Most Significant Blue palette data
25:21	MG[4:0]	Most Significant Green palette data
20:16	MR[4:0]	Most Significant Red palette data
15	I	Intensity
14:10	LB[4:0]	Least Significant Blue palette data
9:5	LG[4:0]	Least Significant Green palette data
4:0	LR[4:0]	Least Significant Red palette data

NOTE: *Blue and red palette data can be swapped by programming CTRL:BGR.

4.3.6.1 Grayscale Algorithm

A patented grayscale algorithm drives the monochrome and color STN panels.

- For monochrome displays, the gray-scale algorithm provides 15 gray scales.
- For color displays, the 3-color components (red, green, and blue) are grayscaled simultaneously. This results in 3,375 colors (15 × 15 × 15) being available for Color STN Mode.

The grayscaler transforms each 4-bit gray value into a sequence of activity-per-pixel over several frames, relying partially on the display characteristics to provide the representation of gray scales and color.

Red, green, and blue pixel data bit values from the gray scaler are shifted concurrently into the respective registers in the upper and lower panel formatter. When sufficient data is available, a byte is constructed by assembling the data to the correct bit position for the LCD panel's RGB data pattern.

4.3.6.2 Interrupts

The CLCDC has four individually maskable interrupt conditions associated with a single combined interrupt. The single combined interrupt is asserted if any of the combined interrupt conditions are asserted and unmasked.

4.3.6.3 LCD Panel Resolutions

LCD panel resolution is expressed as the total number of horizontal pixels multiplied by the total number of vertical pixels. The CLCDC supports STN, TFT, HR-TFT, and AD-TFT LCD panels. The only requirement for supporting these displays is for the width to be divisible by 16; there is no limit on height. Examples of resolutions supported include:

- 160 × 120
- 320 × 200
- 320 × 240
- 640 × 200
- 640 × 240
- 640 × 480 8-bit (this resolution requires significant processor bandwidth)

TFT, HR-TFT, and AD-TFT LCD panels utilize color palette RAM. For these panels, each 16-bit palette entry is composed of 5 bpp, plus a common intensity bit. In addition, the total number of supported colors can be doubled from 32,768 to 65,536 if the Intensity bit is utilized and applied simultaneously to all three color components (R, G, and B).

Table 4-6 and Table 4-7 show the supported bpp for TFT, HR-TFT, AD-TFT and STN panels. Table 4-8 shows the bpp for monochrome STN panels.

Table 4-6. Supported TFT, HR-TFT, and AD-TFT LCD Panels

BPP	SOURCE	HR-TFT (UP TO 16-BIT BUS)	NOTES
1	Palletized	2 colors selected from 65,536 available colors	
2	Palletized	4 colors selected from 65,536 available colors	
4	Palletized	16 colors selected from 65,536 available colors	
8	Palletized	256 colors selected from 65,536 available colors	
12	Direct	4:4:4 RGB	1
16	Direct	5:5:5 RGB + Intensity. The Intensity bit can be unused, or it can be used to connect to the LSB of the R, G, and B components of a 6:6:6 TFT panel.	2

NOTES:
1. LH79525 with 12-Bit CLCDC
2. LH79524 with 16-Bit CLCDC

Table 4-7. Supported Color STN LCD Panels (LH79524 only)

BPP	SOURCE	COLOR STN (SINGLE AND DUAL PANEL, 8-BIT BUS)	NOTE
1	Palletized	2 colors selected from 3,375 available colors	1
2	Palletized	4 colors selected from 3,375 available colors	1
4	Palletized	16 colors selected from 3,375 available colors	1
8	Palletized	256 colors selected from 3,375 available colors	1

NOTE:
1. 3,375 colors = (15 RED) × (15 BLUE) × (15 GREEN) .

Table 4-8. Supported Mono-STN LCD Panels

bpp	SOURCE	MONO STN (SINGLE AND DUAL, 4- AND 8-BIT BUS)
1	Palletized	2 gray scales selected from 15
2	Palletized	4 gray scales selected from 15
4	Palletized	15 gray scales selected from 15

Table 4-9 shows the intensity that can be obtained from each of the 16 possible 4-bit palette combinations. Only 15 of the combinations are useful because the values 0b0110 and 0b1000 produce the same result.

Table 4-9. Color STN Intensities From Gray-Scale Modulation

4-BIT PALETTE VALUE	DUTY CYCLE[1]	RESULTING INTENSITY[2]
0b0000	0/90	00.0%
0b0001	10/90	11.1%
0b0010	18/90	20.0%
0b0011	24/90	26.7%
0b0100	30/90	33.3%
0b0101	36/90	40.0%
0b0110	40/90	44.4%
0b0111	45/90	50.0%
0b1000	45/90	50.0%
0b1001	50/90	55.6%
0b1010	54/90	60.0%
0b1011	60/90	66.6%
0b1100	66/90	73.3%
0b1101	72/90	80.0%
0b1110	80/90	88.9%
0b1111	90/90	100.0%

NOTES:
1.Duty cycle is determined by (pixels on ÷ (pixels on + pixels off)).
2.Resulting intensity: 000% = black, 100% = white.

4.3.7 LCD Data Multiplexing

When LCD data is written to a LCD panel, the manner in which the LCD data is multiplexed onto the external data bus varies for STN, TFT, AD-TFT, or HR-TFT panels. Table 4-10 and Table 4-11 show the data multiplexing for each supported panel.

Table 4-10. LH79524 LCD Data Multiplexing

CABGA BALL NO.	CABGA BALL NAME	STN						TFT
		MONO 4-BIT		MONO 8-BIT		COLOR		COLOR
		SINGLE PANEL	DUAL PANEL	SINGLE PANEL	DUAL PANEL	SINGLE PANEL	DUAL PANEL	SINGLE PANEL
C2	LCDVD15	MUSTN0	MUSTN0	MUSTN0	MUSTN0	CUSTN0	CUSTN0	INTENSITY
C1	LCDVD14	X	X	X	MLSTN4	X	CLSTN4	BLUE4
C10	LCDVD13	X	X	MUSTN6	MUSTN6	CUSTN6	CUSTN6	BLUE3
A10	LCDVD12	X	X	X	MLSTN7	X	CLSTN7	BLUE2
A11	LCDVD11	X	X	X	MLSTN6	X	CLSTN6	BLUE1
B10	LCDVD10	X	X	X	MLSTN5	X	CLSTN5	BLUE0
C9	LCDVD9	X	MLSTN3	X	MLSTN3	X	CLSTN3	GREEN4
B9	LCDVD8	X	MLSTN2	X	MLSTN2	X	CLSTN2	GREEN3
A9	LCDVD7	X	MLSTN1	X	MLSTN1	X	CLSTN1	GREEN2
A8	LCDVD6	X	MLSTN0	X	MLSTN0	X	CLSTN0	GREEN1
B8	LCDVD5	X	X	MUSTN7	MUSTN7	CUSTN7	CUSTN7	GREEN0
C8	LCDVD4	X	X	MUSTN5	MUSTN5	CUSTN5	CUSTN5	RED4
A7	LCDVD3	X	X	MUSTN4	MUSTN4	CUSTN4	CUSTN4	RED3
B7	LCDVD2	MUSTN3	MUSTN3	MUSTN3	MUSTN3	CUSTN3	CUSTN3	RED2
C7	LCDVD1	MUSTN2	MUSTN2	MUSTN2	MUSTN2	CUSTN2	CUSTN2	RED1
A6	LCDVD0	MUSTN1	MUSTN1	MUSTN1	MUSTN1	CUSTN1	CUSTN1	RED0

NOTES:
'X' is Don't Care.
'MLSTNx' = Mono Lower panel STN data bit 'x'
'MUSTNx' = Mono Upper panel STN data bit 'x'
'CLSTNx' = Color Lower panel STN data bit 'x'
'CUSTNx' = Color Upper panel STN data bit 'x'

Table 4-11. LH79525 LCD Data Multiplexing

PIN NO.	PIN NAME	STN MONO 4-BIT	
		SINGLE PANEL	DUAL PANEL
145	LCDVD11	MUSTN1	MUSTN1
146	LCDVD10	MUSTN0	MUSTN0
147	LCDVD9		
149	LCDVD8		
151	LCDVD7		MLSTN3
153	LCDVD6		MLSTN2
154	LCDVD5		MLSTN1
155	LCDVD4		MLSTN0
156	LCDVD3		
157	LCDVD2		
158	LCDVD1	MUSTN3	MUSTN3
159	LCDVD0	MUSTN2	MUSTN2

NOTES:
'X' is Don't Care.
'MLSTNx' = Mono Lower panel STN data bit 'x'
'MUSTNx' = Mono Upper panel STN data bit 'x'
'CLSTNx' = Color Lower panel STN data bit 'x'
'CUSTNx' = Color Upper panel STN data bit 'x'

4.3.8 LCD Interface Timing Signals

LCD interface timing signals are categorized as either horizontal or vertical timing signals. These signals are created by the CLCDC, optionally modified by the ALI, and applied directly to an external LCD panel with no additional external hardware required, except for Continuous Grain Silicon (CGS) panels.

4.3.8.1 LCD Horizontal Timing Signals

The horizontal components of LCD timing describe the process of writing one line of LCD data to a LCD panel and include programmable delays before and after the data is written to the panel. A line of data is composed of all pixel information for one displayed line. See Section 4.6 for timing diagrams.

4.3.8.1.1 STN Horizontal Timing Restrictions

The CLCDC's dedicated DMA system requests new data at the start of each horizontal display line. Time must be allowed for the DMA transfer operation to occur. Time must also be allowed for the data to propagate down the FIFO path within the LCD interface. These delays constitute LCD data path latency. The data path latency imposes some restrictions on the usable minimum values for horizontal back porch width when operating in the STN modes. The value restrictions are listed in Table 4-12.

Table 4-12. Usable Minimum Values Affecting STN Back Porch Width

HORIZONTAL TIMING VALUE	SINGLE-PANEL MODE	DUAL-PANEL MODE
TIMING0:HSW	3	3
TIMING0:HBP	5	5
TIMING0:HFP	5	5
TIMING2:PCD	1 × (CLCD CLOCK/3)	5 × (CLCD CLOCK/7)

NOTE: The minimum value for PCD is 4.

4.3.8.2 LCD Vertical Timing Signals

Data is written to an LCD panel in frames. Each frame is composed of a number of horizontal lines. The vertical components of LCD timing describe the process of writing one full frame to an LCD panel.

Each frame begins with a frame pulse or vertical synchronization pulse of programmable duration. Each frame pulse is followed by a programmable delay, the vertical back porch. When the vertical back porch expires, all line information for the frame is presented to the LCD panel. See Section 4.3.8.1. The line information is followed by another programmable delay, the vertical front porch.

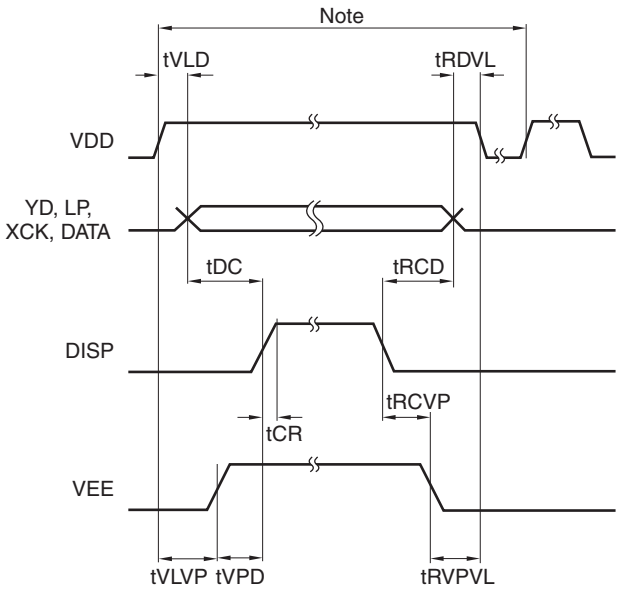
In STN mode, the vertical front porch, pulse width, and vertical back porch are not programmable.

4.3.9 LCD Power Sequencing at Turn-On and Turn-Off

Many LCD panels require ground, power for the digital logic, and high-voltage power supplies. To extend the life of these panels, the digital power must be applied before the high voltage is applied, and removed after they are removed. The logic signals driving the panel must be active before the panel voltages are applied, and the panel voltages must be removed before the logic signals are removed. This sequencing ensures that the panel is always operated with a net DC bias of 0 VDC.

Software must ensure that these conditions are met. The requisite delay is usually specified in the LCD panel's data sheet. If the proper power sequencing is not followed, the LSI drivers in the panel can latch and the display will freeze. Typically when this happens, the colors will be incorrect on STN panels. In addition the power down sequence must be followed or LCD life can be degraded.

Figure 4-4 is an example of these timing requirements for the NXP LM057QCTT03 Color STN LCD Panel, and the accompanying timing specifications. Always refer to your specific LCD panel's Data Sheet to determine the specific turn-on and turn-off requirements for the panel being used in your application.



POWER ON

SYMBOL	ALLOWABLE VALUE	
tVLD	0 ms MIN.	1 sec. MAX.
tDC	20 ms MIN.	
tCR		100 ns MAX.
tVPD	0 ms MIN.	
tVLVP	0 ms MIN.	

POWER OFF

SYMBOL	ALLOWABLE VALUE	
tRDVL	0 ms MIN.	1 sec. MAX.
tRCD	20 ms MIN.	
tRCVP	20 ms MIN.	
tRVPVL	0 ms MIN.	

NOTE: 500 ms MIN. ON/OFF cycle time. All signals and power must be switched in this sequence during power ON/OFF cycles.

79525-47

Figure 4-4. LCD Panel Power Sequencing

4.3.9.1 Minimizing a Retained Image on the LCD

While it is very important to follow the power turn-off sequence to ensure longevity of the LCD panel, this sequence alone will not ensure there is no retained image (ghosting) left on the LCD panel after the LCD has been powered down.

This ghost bleeds away slowly after powering down the LCD panel. It is most noticeable with LCD panels utilizing HR-TFT and AD-TFT technologies to light the LCD. With these types of LCD panels the ambient light alone is enough to make the retained image visible. TFT-type LCD panels also have a retained image, but it is typically not as visible once the backlight source is turned off.

To minimize the appearance of a retained image on HR-TFT and AD-TFT LCD panels, software should write a complete frame of all 1s (white) to the LCD just prior to initiating the turn-off sequence.

To minimize the appearance of a retained image on a TFT LCD panel, software should write a complete frame of all 0s (black) just prior to initiating the turn-off sequence.

In all cases the illumination source should be turned off prior to initiating the turn-off sequence.

4.3.10 Interrupts

The single combined interrupt, CLCDINTR, is used to drive the VIC. If any of the four interrupt conditions occurs, this signal is asserted. CLCDINTR drives the VIC.

Each of the four individual maskable interrupt conditions is enabled or disabled by changing the mask bits in the INTREN Register. Provision of individual outputs, along with a combined interrupt output, allows the use of either a global interrupt service routine or modular device drivers to handle interrupts. The status of the individual interrupt sources can be read from the Status Register.

4.4 Advanced LCD Interface

The Advanced LCD Interface (ALI) provides the additional processing required to interface the LH79524 and LH79525 to AD-TFT, HR-TFT, or any display technology that uses this method of connection. Figure 4-5 shows the ALI between the CLCDC and the LCD output pins.

The ALI is programmed via its16-bit APB interface and receives control signals and display data from the CLCDC. The ALI converts the display data to a format suitable for direct connection to the Row and Column driver ICs in AD-TFT, HR-TFT displays, or any display using similar technology.

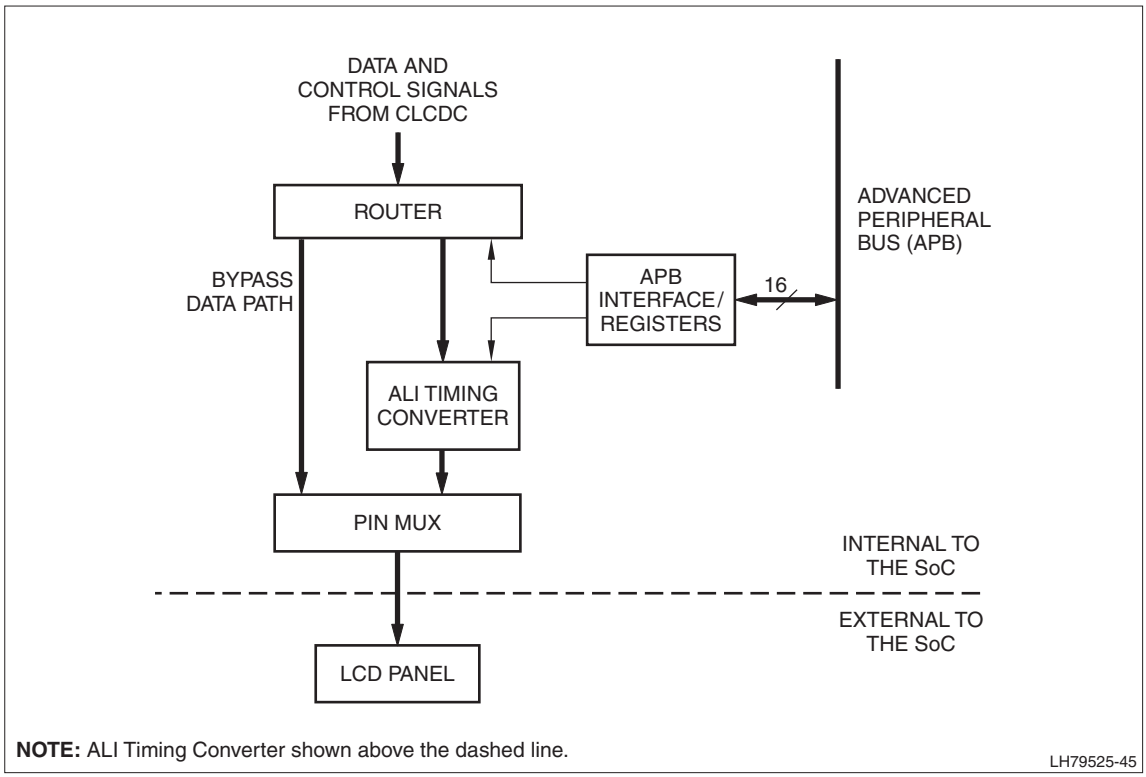


Figure 4-5. ALI Simplified Block Diagram

4.4.1 ALI Theory of Operation

All ALI Control and Status Registers can be accessed through the APB. One of the registers, the ALI Setup Register, can be programmed to select Bypass Mode or Active Mode.

- In Bypass Mode, LCD Controller signals are passed directly to the pins.
- In Active Mode, TFT data and control signals from the LCD Controller generate a set of signals for driving an AD-TFT or HR-TFT display.

Selecting Active Mode re-times the data to the falling edge of the output clock. The formatter also provides the:

- Features of normal scanning signals for vertical and horizontal scan
- Generation of source driver, gate driver, and voltage-preparation control signals.

The timing parameters for the Active Mode are register-programmable. When using Active Mode, program the ALI Setup Register first, followed by ALI Timing registers 1 and 2. After these registers are programmed, the LCD Controller can be enabled and the ALI Control Register can be used.

The ALI generates the MOD signal automatically. By default, activation of MOD occurs 2 SPS rising edge clocks after activation of the controller. This can be reprogrammed for a longer or shorter wait, or can be overridden via the ALI Control Register.

4.4.2 ALI Operating Modes

The ALI has two operating modes: Bypass Mode or Active Mode. The ALI Setup Register setting specifies the operating mode of the ALI (see Section 4.5.6.1).

4.4.2.1 Bypass Mode

The ALI defaults to Bypass mode following reset. In this mode, signals and data received from the CLCDC simply pass unaltered through the ALI to the LCD output pins. Select the Bypass mode when the CLCDC is driving STN or TFT LCD panels that contain a timing ASIC.

4.4.2.2 Active Mode

In Active mode, the ALI reformats TFT data and control signals from the CLCDC to drive the Row and Column driver circuitry on the LCD panel. When ALISETUP:CR is programmed to select the Active mode, the ALI re-times TFT data from the CLCDC so that it is output on the falling edge of the output clock (LCDDCLK). In the Active mode the ALI also generates source driver, gate driver, and voltage-preparation control signals appropriate for AD-TFT and HR-TFT LCDs.

The correct programming sequence for the CLCDC and ALI registers in Active mode is:

1. Ensure that the CLCDC is not enabled
2. Program the ALISETUP register
3. Program the ALITIMING1 Register
4. Program the ALITIMING2 register
5. Program the ALICONTROL register
6. Enable the CLCDC.

4.4.2.3 CLCDC Setup for AD-TFT or HR-TFT Operation

To supply the correct waveforms, the ALI must receive the correct signals from the CLCDC. Software must:

- Program the CLCDC to scale the internal clock signal routed to the ALI from the CLCDC to a frequency appropriate for the AD-TFT or HR-TFT panel being connected. The ALI will modify this signal's timing but not its frequency.
- Program the CLCDC to operate in TFT mode.
- Program the CLCDC to provide an enable signal (either LCDSPR or LCDSPL).
- Program the CLCDC to provide a continuous clock signal (LCDDCLK).
- Program TIMING2:IVS to select the appropriate polarity for the LCDSPS (Row reset) signal.
- Program TIMING2:IHS to select the appropriate polarity for the LCDLP (Horizontal Sync) signal.
- Program TIMING2:IPC to drive data on the falling edge of the LCDDCLK signal.
- Program TIMING2:IOE to 0 to select the appropriate polarity for the LCDSPR/LCDSPL signal.

4.5 CLCDC Register Reference

This section contains the register definitions for the CLCDC. ALI registers are found in the next section.

4.5.1 Enabling the CLCDC

Following reset, the CLCDC Data Clock is gated OFF. Prior to using the CLCDC, it must be enabled by turning on the LCD Data Clock in the PCLKCTRL1 register of the Reset, Clock, and Power Controller block (see Section 13.2.2.10). *Registers cannot be programmed until the LCD Data Clock is enabled.*

4.5.2 CLCDC Memory Map

The base address for the CLCDC is: 0xFFFF4000

Table 4-13. CLCDC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	TIMING0	Horizontal Axis Timing Control
0x004	TIMING1	Vertical Axis Timing Control
0x008	TIMING2	Clock and Signal Polarity Control Register
0x00C	///	Reserved — Do not access
0x010	UPBASE	Upper Panel Frame Buffer Base Address Register
0x014	LPBASE	Lower Panel Frame Buffer Base Address Register
0x018	INTREN	Interrupt Enable Register
0x01C	CTRL	Panel Parameters, Panel Power, and Control
0x020	STATUS	Raw Interrupt Status Register
0x024	INTERRUPT	Masked Interrupt Status Register
0x028	INTCLR	Interrupt Clear Register
0x02C	UPCURR	Upper Panel Frame Buffer Current Address Register
0x030	LPCURR	Lower Panel Frame Buffer Current Address Register
0x034 - 0x1FC	///	Reserved — Do not access
0x200 - 0x3FC	PALETTE	256 × 16-bit Color Palette Register. Palette is addressed at 32 bits.
0x400 - 0x7FF	///	Reserved — Do not access

4.5.3 CLCDC Register Descriptions

4.5.3.1 Horizontal Timing Panel Control Register (TIMING0)

The TIMING0 Register controls:

- Horizontal Synchronization Pulse Width (HSW)
- Horizontal Front Porch (HFP) period
- Horizontal Back Porch (HBP) period
- Pixels-Per-Line (PPL)

Table 4-14. TIMING0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	HBP								HFP							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	HSW								PPL						///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO
ADDR	0xFFFF4000 + 0x00															

Table 4-15. TIMING0 Fields

BIT	NAME	DESCRIPTION
31:24	HBP	**Horizontal Back Porch** HBP specifies the number of LCDDCLK periods between the end of the LCDLP signal and the beginning of valid data. HBP can be programmed for a delay of 1 to 256 pixel clock cycles. HBP = (LCDDCLK periods) – 1
23:16	HFP	**Horizontal Front Porch** HFP specifies the number of LCDDCLK periods between the end of valid data and the beginning of the LCDLP signal. HFP can be programmed for a delay of 1 to 256 pixel clock cycles. HFP = (LCDDCLK periods) – 1
15:8	HSW	**Horizontal Synchronization Pulse Width** HSW specifies the width of the LCDLP signal, in LCDDCLK periods. In STN modes, this signal is referred to as the 'line clock'. In TFT modes, this signal is referred to as the 'horizontal synchronization pulse'. When using AD-TFT panels, HSW always inserts 1 LCDDCLK period, regardless of the value programmed to this field. HSW = (LCDDCLK periods) – 1
7:2	PPL	**Pixels-Per-Line** PPL specifies the number of pixels in each line of the LCD panel. The PPL value sets the number of pixel clocks that occur before the value in the HFP bit field is applied (that is, before the LCDLP signal is asserted). The PPL bit field is a 6-bit value that represents a number corresponding to the actual pixels-per-line, which can range from 16 to 1,024. At reset this field is 0, which corresponds to 16 actual pixels-per-line. PPL = (Actual pixels-per-line/16) – 1 Actual pixels-per-line = 16 × (PPL + 1)
1:0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.1.1 Horizontal Timing Restrictions

The LCD DMA requests new data at the start of a horizontal display line. Some time must be allowed for the DMA transfer and for the data to propagate down the FIFO path in the LCD interface. The data path latency forces some restrictions on the usable minimum values for horizontal porch width in STN Mode.

The minimum values are HSW = 2 and HBP = 2.

Single Panel Mode:

- HSW = 3
- HBP = 5
- HFP = 5
- Panel Clock Divisor (PCD) = 1 (CLCDCLK/3)

Dual Panel Mode:

- HSW = 3
- HBP = 5
- HFP = 5
- PCD = 5 (CLCDCLK/7)

If sufficient time is given at the start of the line (for example, setting HSW = 6, HBP = 10), data will not get corrupted for PCD = 4 (minimum value).

NOTE: CLCDCLK is a separate clock provided by the Reset Clock and Power Controller (RCPC).

4.5.3.2 Vertical Timing Panel Control Register (TIMING1)

The TIMING1 Register controls the:

- Number of Lines-Per-Panel (LPP)
- Vertical Synchronization Pulse Width (VSW)
- Vertical Front Porch (VFP) period
- Vertical Back Porch (VBP) period

Table 4-16. TIMING1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	VBP								VFP							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	VSW						LPP									
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF4000 + 0x04															

Table 4-17. TIMING1 Fields

BIT	NAME	DESCRIPTION
31:24	VBP	**Vertical Back Porch** VBP defines the number of inactive lines at the start of a frame, after the vertical synchronization period (after the framing pulse, LCDFP, is de-asserted). The VBP bit field specifies the number of horizontal line-clocks (LCDLP) inserted at the beginning of each frame. The value in the VBP bit field can generate from 0 to 255 additional line-clock cycles. TFT modes: The VBP delay begins after the vertical synchronization signal for the previous frame (LCDFP) has been deasserted. STN modes: Program to zero. The vertical back porch is not programmable and always a duration of zero. Programming any other value other than zero will have no effect.
23:16	VFP	**Vertical Front Porch** VFP defines the number of inactive lines at the end of a frame, before the vertical synchronization period (before the framing pulse, LCDFP, is asserted). VFP specifies the number of horizontal line-clocks (LCDLP) inserted at the end of each frame. The value in the VFP bit field can generate from 0 to 255 line-clock cycles. TFT modes: The vertical synchronization signal (LCDFP) is asserted after the VFP delay has expired. STN modes: Program small value. As higher values are programmed, the contrast of the display will be reduced.
15:10	VSW	**Vertical Synchronization (Pulse) Width** TFT Modes: VSW is the width of the LCDFP signal. VSW is specified in terms of the number of horizontal synchronization lines (LCDLP pulses). VSW = (LCDLP Periods) – 1 STN modes: Program to 0. The vertical sync pulse width is not programmable and always a duration of 1 line. Programming any other value than zero will led to unexpected behavior. When using AD-TFT panels, VSW always inserts 3 LCDDCLK period, regardless of the value programmed to this field.

Table 4-17. TIMING1 Fields (Cont'd)

BIT	NAME	DESCRIPTION
9:0	LPP	**Lines Per Panel** LPP specifies the number of active lines (rows of pixels) per panel. The LPP bit field is a 10-bit value allowing between 1 and 1,024 lines. LPP = (Active Lines) – 1 Dual-Panel displays: The two panels in dual-panel displays are assumed to be of identical sizes. For dual-panel displays, the LPP bit field should be programmed to describe either the upper or the lower panel, and not be doubled. ALI Active modes: LPP and LCDREV must be considered together. Program LPP to an odd number of lines for AD-TFT and HR-TFT panels. AD-TFT and HR-TFT displays utilize the LCDREV signal as an AC bias signal (it does not alter LCDDCLK). The LCDREV signal oscillates, driven HIGH during one frame and driven LOW during the next frame. To avoid long-term LCD damage, the AC bias applied to any line of an LCD panel should average to a net 0 VDC. When correctly programmed, the LCDREV signal will be HIGH for a line during one frame and LOW for the same line during the next frame. If the LPP bit field specifies an even number of Lines Per Panel, this oscillation will be mismatched to the display's lines and the lines will not receive a net 0 VDC bias. See ALITIMING1:REVDEL for additional information.

4.5.3.3 Clock and Signal Polarity Control Register (TIMING2)

The TIMING2 Register controls the CLCDC timing.

Table 4-18. TIMING2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	PCD_HI					BCD	CPL									
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///	IOE	IPC	IHS	IVS	ACB					///	PCD_LO				
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF4000 + 0x08															

Table 4-19. TIMING2 Fields

BIT	NAME	DESCRIPTION
31:27	PCD_HI	**Panel Clock Divisor (upper five bits)** The ten-bit PCD field, comprising PCD_HI and PCD_LO (bits [4:0]) is used to derive the LCD Panel clock frequency CLCP from the CLCDCLK frequency: CLCP = CLCDCLK/(PCD + 2). For mono STN displays with a four or eight-bit interface, the panel clock is a factor of four and eight of the actual individual pixel clock rate. For color STN displays, 2 2/3 pixels are output per CLCP cycle, therefore the panel clock is 0.375 times. For TFT displays the pixel clock divider can be bypassed by setting the TIMING2:BCD bit.
26	BCD	**Bypass Pixel Clock Divider** 1 = Bypass the pixel clock divider logic 0 = Use the pixel clock divider logic See the description of the PCD bit field, below.
25:16	CPL	**Clocks Per Line** CPL specifies the number of LCDDCLK pulses fed to the LCD panel during each horizontal line. The TIMING2:CPL and TIMING0:PPL fields work together; both must be programmed correctly in order for the CLCDC to function correctly. Actual Pixels Per Line (APPL) = 16 x (TIMING0:PPL – 1) TFT panels: CPL = (APPL – 1) 4-bit mono STN panels: CPL = ((APPL/4) – 1) 8-bit mono STN panels: CPL = ((APPL/8) – 1) Color STN panels: CPL = (((3 x APPL) / 8) – 1)
15	///	**Reserved** Reading returns 0. Write the reset value.
14	IOE	**Invert Output Enable** IOE applies only to TFT modes and should be programmed to 0 for all other modes. In the TFT mode, the LCDEN pin indicates to the LCD panel that valid display data is available. IOE selects the active polarity of this output enable signal. In the TFT mode, data is driven onto the LCD data lines at the programmed edge of LCDDCLK when LCDEN is asserted. 1 = The LCDEN output pin is active LOW 0 = The LCDEN output pin is active HIGH
13	IPC	**Invert Panel Clock** IPC selects the active edge of the LCDDCLK signal. 1 = Data is driven on the LCD data lines on the falling-edge of LCDDCLK 0 = Data is driven on the LCD data lines on the rising-edge of LCDDCLK
12	IHS	**Invert Horizontal Synchronization** IHS selects the polarity of the LCDLP signal. 1 = The LCDLP pin is active LOW 0 = The LCDLP pin is active HIGH

Table 4-19. TIMING2 Fields (Cont'd)

BIT	NAME	DESCRIPTION
11	IVS	**Invert the Vertical Synchronization Signal** IVS selects the polarity of the LCDFP signal. 1 = LCDFP is active LOW 0 = LCDFP is active HIGH
10:6	ACB	**AC Bias Signal Frequency** ACB sets the frequency of the LCDEN signal. STN modes: ACB applies to the CLCDC when it is operating in the STN mode. STN displays require periodic reversal of the pixel voltages in order to prevent damage to the STN panel due to DC charge accumulation. Program this field to select the required number of line clocks (the LCDLP signal) between each toggle of the AC bias signal (LCDEN). ACB = (line clocks) – 1 TFT modes: This field has no effect if the CLCDC is operating in TFT mode because the LCDEN pin is instead utilized for a Data Enable signal.
5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	PCD_LO	**Panel Clock Divisor** Program this field and the PCD_HI field to select the LCD panel clock frequency (LCDDCLK frequency) from the input CLCDC CLOCK frequency. LCDDCLK = (CLCDC CLOCK)/(PCD+2) Mono STN modes: LCDDCLK for mono STN panels with a four- (or eight-) bit interface should be programmed to be 1/4 (or 1/8) the desired individual pixel clock rate. Color STN modes: Color STN displays receive multiple pixels during each clock cycle. The pixel data for Color STN displays is stored and transferred in packed format, with each pixel represented by three bits (R,G and B). Therefore, one byte contains the pixel data for 2 2/3 pixels (RGB, RGB, RG) and three bytes contain the pixel data for eight complete pixels. For Color STN panels, each LCDDCLK cycle transfers one byte, containing 2 2/3 pixels, to the panel. LCDDCLK should be programmed to be as close as possible to 3/8 the desired individual pixel clock rate. TFT mode: For TFT displays, the pixel clock divider can be bypassed by programming TIMING2:BCD = 1.

4.5.3.4 Upper Panel Frame Buffer Base Address Register (UPBASE)

The UPBASE Register is one of two Color LCD DMA Base Address Registers (the other is LPBASE, described in Section 4.5.3.5). Together with LPBASE, this Read/Write register programs the base address of the frame buffer.

UPBase is used for:

- TFT displays
- Single-panel STN displays
- The upper panel of dual-panel STN displays.

UPBASE (and LPBASE for dual panels) must be initialized before enabling the CLCDC. Optionally, the value can be changed mid-frame to allow double-buffered video displays to be created. These registers are copied to the corresponding current registers at each LCD vertical synchronization. This event causes the BUI bit and an optional interrupt to be generated. The BUI bit indicates that it is safe to update both the UPBASE and LPBASE Registers. The interrupt can be used to reprogram the base address when generating double-buffered video.

Table 4-20. UPBASE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	LCDUPBASE															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	LCDUPBASE														///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO
ADDR	0xFFFF4000 + 0x10															

Table 4-21. UPBASE Fields

BIT	NAME	DESCRIPTION
31:2	LCDUPBASE	**LCD Upper Panel Base Address** This is the start address of the upper panel frame data stored in memory and is word-aligned.
1:0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.5 Lower Panel Frame Buffer Base Address Register (LPBASE)

The LPBASE Register is one of two Color LCD DMA Base Address Registers (the other is UPBASE, described in Section 4.5.3.4). Together with UPBASE, this Read/Write register programs the base address of the frame buffer.

LPBASE is used for the lower panel of dual-panel STN displays. UPBASE must be initialized (and LPBASE for dual panels) before enabling the CLCDC. Optionally, the value can be changed mid-frame to allow double-buffered video displays to be created. These registers are copied to the corresponding current registers at each LCD vertical synchronization. This event causes the BUI bit and an optional interrupt to be generated. The BUI bit indicates that it is safe to update both the UPBASE and LPBASE Registers. The interrupt can be used to reprogram the base address when generating double-buffered video.

Table 4-22. LPBASE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	LCDLPBASE															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	LCDLPBASE														///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO
ADDR	0xFFFF4000 + 0x14															

Table 4-23. LPBASE Register Fields

BIT	NAME	DESCRIPTION
31:2	LCDLPBASE	**LCD Lower Panel Base Address** This is the start address of the lower panel frame data in memory and is word-aligned.
1:0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.6 Interrupt Enable Register (INTREN)

INTREN is the Interrupt Enable Register. Setting bits within this register enables the corresponding Raw Interrupt Status bit values to be passed to the Raw Interrupt Status Register (see Section 4.5.3.8).

Table 4-24. INTREN Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											MBEIEN	VCIEN	BUIEN	FUIEN	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF4000 + 0x18															

Table 4-25. INTREN Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	MBEIEN	**Bus Master Error Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
3	VCIEN	**Vertical Compare Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
2	BUIEN	**Next Base Update Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
1	FUIEN	**FIFO Underflow Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.7 CLCDC Control Register (CTRL)

CTRL controls the CLCDC operating mode. All registers should be set up prior to programming LCDEN to 1.

Note that the operating mode (color/mono, bits-per-pixel, etc.) can only be changed between frames to avoid corruption of the current frame data. To ensure this is done properly, use the Vertical Compare Interrupt to detect that the current frame is complete. To do this, follow these steps when changing operating mode:

1. Program the CTRL: VCI bit to 0b00 to generate the Vertical Compare Interrupt on entry to the Sync State. Enable this interrupt by programming the INTREN:VCIEN bit to 1.
2. Program the CTRL:LCDEN bit to 0. The controller will complete the current frame before sampling this bit.
3. Wait for the Vertical Compare Interrupt. Upon assertion of the interrupt, program the new mode (e.g. from color to monochrome using the CTRL:BW bit).

Then program the CTRL:LCDEN bit to 1 and clear the Vertical Compare Interrupt by writing a 1 to the STATUS:VCI bit. The CLCDC will resume operating, using the newly programmed parameters.

Table 4-26. CTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															WATERMARK
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		VCI		PWR	BEPO	BEBO	BGR	DUAL	MONO8L	TFT	BW	BPP			LCDEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF4000 + 0x1C															

Table 4-27. CTRL Fields

BIT	NAME	DESCRIPTION
31:17	///	**Reserved** Reading returns 0. Write the reset value.
16	WATERMARK	**LCD DMA FIFO Watermark Level** 1 = Requests data when either of the two DMA FIFOs have eight or more empty locations. 0 = Requests data when either of the two DMA FIFOs have four or more empty locations.
15:14	///	**Reserved** Reading returns 0. Write the reset value.
13:12	VCI	**LCD Vertical Compare** Program to generate an interrupt at: 00 = start of vertical synchronization 01 = start of back porch 10 = start of active video 11 = start of front porch
11	PWR	**LCD Power Enable** This bit controls power to the LCD panel. For this bit to be functional, the CONTROL:LCDEN bit must be programmed to 1. If programmed to 0, LCD power is off regardless of the setting of the PWR bit. 1 = LCD power ON 0 = LCD power OFF
10	BEPO	**Big-Endian Pixel Ordering** The BEPO bit selects between little and big-endian pixel packing for 1, 2 and 4 bpp display modes. The BEPO bit has no effect on 8, 12, or 16 bpp pixel formats. 1 = Big-endian pixel ordering within a byte 0 = Little-endian ordering within a byte
9	BEBO	**Big-Endian Byte Ordering to the LCD** 1 = Big-endian byte order 0 = Little-endian byte order
8	BGR	**RGB or BGR Format Selection** 1 = Bits 14:10 and 4:0 swapped (blue and red swapped) 0 = Display data normal output
7	DUAL	**Dual Panel STN LCD** 1 = Select Dual Panel LCD Mode 0 = Select Single Panel LCD Mode
6	MONO8L	**Monochrome LCD** LCD is Monochrome with an 8-bit interface. This bit controls whether a monochrome STN LCD uses a 4 or an 8-bit parallel interface. It should be programmed to 0 for all other types of displays. 1 = Mono LCD uses 8-bit interface 0 = Mono LCD uses 4-bit interface
5	TFT	**TFT LCD** 1 = LCD is TFT — do not use grayscaler 0 = LCD is an STN display — use grayscaler

Table 4-27. CTRL Fields (Cont'd)

BIT	NAME	DESCRIPTION
4	BW	**Monochrome STN LCD** LCD is Monochrome (Black and White) STN. This bit has no effect in TFT mode. 1 = STN LCD is monochrome 0 = STN LCD is color
3:1	BPP	**LCD Bits-Per-Pixel** For the LH79525, 12 bpp is selected by 0b100. See Table 4-2 and Table 4-3 for 12-bit mapping in the 16-bit frame buffer. 000 = 1 BPP 001 = 2 BPP 010 = 4 BPP 011 = 8 BPP 100 = 16 BPP 101 = Invalid 110 = Invalid 111 = Invalid
0	LCDEN	**Color LCD Controller Enable** LCD displays usually require that their logic signals be operating before the high voltages are applied to the display. Thus, the LCDVDDEN output signal is not asserted unless both the LCDEN and PWR bit fields have been programmed to 1. Most LCD displays require that the controller be enabled (LCDEN = 1) approximately 20 ms before power is applied to the LCD (PWR = 1). Most LCD displays also specify that the power-down sequence be the reverse of the power-up sequence. 1 = Color LCD Controller enabled 0 = Color LCD Controller disabled

4.5.3.8 Raw Interrupt Status Register (STATUS)

STATUS is the Raw Interrupt Status Register. The status of the interrupts without masking applied is contained in this register.

Table 4-28. STATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											MBEI	VCI	BUI	FUI	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF4000 + 0x20															

Table 4-29. STATUS Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	MBEI	**AMBA AHB Master Bus Error Status** Indicates that the CLCDC AHB master has encountered a bus error response from a slave. 1 = Interrupt asserted 0 = No interrupt
3	VCI	**Vertical Compare** Set to 1 when one of the four vertical regions selected in the CONTROL register is reached. 1 = Interrupt asserted 0 = No interrupt
2	BUI	**LCD Next Base Address Update** Mode dependent; set to 1 when the Current Base Address registers have been successfully updated with the data from Next Address registers. Signifies that a new Next Address can be loaded if double buffering is in use. 1 = Interrupt asserted 0 = No interrupt
1	FUI	**FIFO Underflow** Set to 1 when either the upper or lower DMA FIFOs have been accessed when empty, resulting in an underflow condition 1 = Interrupt asserted 0 = No interrupt
0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.9 Masked Interrupt Status Register (INTERRUPT)

The INTERRUPT Register is a Read Only register. It is a bit-by-bit logical AND of the Raw Interrupt Status Register (see Section 4.5.3.8) and the INTREN Register (see Section 4.5.3.6). Interrupt lines correspond to each interrupt. A logical OR of all interrupts is provided to the Vectored Interrupt Controller.

Table 4-30. INTERRUPT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											MBEIM	VCIM	BUIM	FUIM	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF4000 + 0x24															

Table 4-31. INTERRUPT Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	MBEIM	**Masked AHB Master Error Interrupt** 1 = Interrupt asserted and enabled 0 = No interrupt
3	VCIM	**Masked Vertical Compare Interrupt** 1 = Interrupt asserted and enabled 0 = No interrupt
2	BUIM	**Masked LCD Next Base Address Update Interrupt** 1 = Interrupt asserted and enabled 0 = No interrupt
1	FUIM	**Masked FIFO Underflow Interrupt** 1 = Interrupt asserted and enabled 0 = No interrupt
0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.10 Interrupt Clear Register (INTCLR)

Writing a 1 to an active bit in this register causes that interrupt to be cleared. This is a write-only register.

Table 4-32. INTCLR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											CMBEI	CVCI	CBUI	CFUI	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	RO
ADDR	0xFFFF4000 + 0x24															

Table 4-33. INTCLR Fields

BIT	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	CMBEI	**Clear Masked AHB Master Error Interrupt** 1 = Interrupt cleared 0 = No change
3	CVCI	**Clear Masked Vertical Compare Interrupt** 1 = Interrupt cleared 0 = No change
2	CBUI	**Clear Masked LCD Next Base Address Update Interrupt** 1 = Interrupt cleared 0 = No change
1	CFUI	**Clear Masked FIFO Underflow Interrupt** 1 = Interrupt cleared 0 = No change
0	///	**Reserved** Reading returns 0. Write the reset value.

4.5.3.11 LCD Upper Panel and Lower Panel Frame Buffer Current Address Register (UPCURR and LPCURR)

UPCURR and LPCURR are registers that contain an approximate value of the upper and lower panel data DMA addresses when read. The registers can change at any time and provide a coarse indication of the current LCD DMA memory pointer.

Table 4-34. UPCURR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	A31	A30	A29	A28	A27	A26	A25	A24	A23	A22	A21	A20	A19	A18	A17	A16
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF4000 + 0x2C															

Table 4-35. UPCURR Fields

BIT	NAME	DESCRIPTION
31:0	A31:A0	A31:A0 of the current lower panel data DMA address. Values change dynamically. Read only.

Table 4-36. LPCURR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	A31	A30	A29	A28	A27	A26	A25	A24	A23	A22	A21	A20	A19	A18	A17	A16
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF4000 + 0x30															

Table 4-37. LCDLPCURR Fields

BIT	NAME	DESCRIPTION
31:0	A31:A0	A31:A0 of the current lower panel data DMA address. Values change dynamically. Read-only.

4.5.3.12 256 × 16-bit Color Palette Register (PALETTE)

The PALETTE Registers contain 256 palette entries organized as 128 locations of two entries per word. TFT displays use 12 of the palette entry bits. Each word location contains two palette entries. This means that 128 word locations are used for the palette. The color that is displayed is based on the setting of bit [8] of the Control Register (see Section 4.5.6.2).

For color STN displays, only bits [4:1] of red, blue, and green palettes are used. For monochrome STN displays, only the red palette bits [4:1] are used.

Table 4-38. PALETTE Register (LH79525 with 12-Bit CLCDC)

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///	MB[3:0]				///	MG[3:0]				///	MR[3:0]				///
RESET	—	—				—	—				—	—				—
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///	LB[3:0]				///	LG[3:0]				///	LR[3:0]				///
RESET	—	—				—	—				—	—				—
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF4000 + 0x200 to 0xFFFF4000 + 0x3FC															

Table 4-39. PALETTE Fields (LH79525 with 12-Bit CLCDC)

BIT	NAME	DESCRIPTION
31	///	Unused. Writes have no effect.
30:27	MB[3:0]	Most Significant Blue palette data
26	///	**Reserved** Reading returns 0. Write the reset value.
25:22	MG[3:0]	Most Significant Green palette data
21	///	**Reserved** Reading returns 0. Write the reset value.
20:17	MR[3:0]	Most Significant Red palette data
16:15	///	**Reserved** Reading returns 0. Write the reset value.
14:11	LB[3:0]	Least Significant Blue palette data
10	///	**Reserved** Reading returns 0. Write the reset value.
9:6	LG[3:0]	Least Significant Green palette data
5	///	**Reserved** Reading returns 0. Write the reset value.
4:1	LR[3:0]	Least Significant Red palette data
0	///	**Reserved** Reading returns 0. Write the reset value.

Table 4-40. PALETTE Register (LH79524 with 16-Bit CLCDC)

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	MI	MB[4:0]					MG[4:0]					MR[4:0]				
RESET	—	—					—					—				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	LI	LB[4:0]					LG[4:0]					LR[4:0]				
RESET	—	—					—					—				
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF4000 + 0x200 to 0xFFFF4000 + 0x3FC															

Table 4-41. PALETTE Fields (LH79524 with 16-Bit CLCDC)

BIT	NAME	DESCRIPTION
31	MI	**Most Significant Intensity Bit** See the bit 15 description in this table.
30:26	MB[4:0]	**Most Significant Blue Palette Data** See the bit 14:10 description in this table.
25:20	MG[4:0]	**Most Significant Green Palette Data** See the bit 9:5 description in this table.
19:16	MR[4:0]	**Most Significant Red Palette Data** See the bit 4:0 description in this table.
15	LI	**Least Significant Intensity Bit** Unused for STN displays. Can be used as the LSB of the R, G and B inputs to a 6:6:6 TFT display, effectively doubling the number of available colors from 32 k to 64 k, where each color has two different intensities.
14:10	LB[4:0]	**Least Significant Blue Palette Data** For color STN displays, only the four MSBs (bits 4:1) of each color are used.
9:5	LG[4:0]	**Least Significant Green Palette Data** For color STN displays, only the four MSBs (bits 4:1) of each color are used.
4:0	LR[4:0]	**Least Significant Red Palette Data** For color STN displays, only the four MSBs (bits 4:1) of each color are used. For monochrome STN displays, only the four MSBs (bits 4:1) of the red palette data is used.

4.5.4 ALI Register Reference

The base address for the ALI is: 0xFFFE4000

Locations at offsets 0x010 through 0xFFF are reserved and must not be used during normal operation.

4.5.5 ALI Memory Map

Table 4-42. ALI Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	ALISETUP	ALI Setup Register
0x004	ALICTRL	ALI Control Register
0x008	ALITIMING1	ALI Timing Register 1
0x00C	ALITIMING2	ALI Timing Register 2
0x010 - 0xFFF	///	Reserved — Do not access

4.5.6 ALI Register Descriptions

4.5.6.1 Setup Register (ALISETUP)

The ALISETUP Register puts the ALI into Bypass Mode and configures the basic behavior of the control signals.

Table 4-43. ALISETUP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///			PPL									///			CR
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
RW	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO	RO	RW
ADDR	0xFFFE4000 + 0x000															

Table 4-44. ALISETUP Fields

BITS	NAME	DESCRIPTION
31:13	///	**Reserved** Reading returns 0. Write the reset value.
12:4	PPL	**Pixels Per Line** PPL = (Actual Pixels per line) – 1.
3:1	///	**Reserved** Reading returns 0. Write the reset value.
0	CR	**Conversion Mode Select** This bit selects the conversion mode for the ALI. Change the ALI mode only when the CLCDC is disabled. 1 = ALI Mode 0 = Bypass Mode

4.5.6.2 Control Register (ALICTRL)

ALICTRL is the Control Register. It enables and controls output signals.

Table 4-45. ALICTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						MODOVRD	MODVAL	///			EN0	DISP	///	CLSEN	SPSEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RW	RW	RO	RO	RO	RW	RW	RO	RW	RW
ADDR	0xFFFE4000 + 0x004															

Table 4-46. ALICTRL Fields

BITS	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9	MODOVRD	**MOD Signal Override Enable** Puts the value of MODVAL directly onto the MOD signal. 0 = LCDMOD pin goes HIGH after the SPS periods specified by the MODDEL field of the TIMING1 Register. 1 = LCDMOD pin equals the state of MODVAL bit in this register.
8	MODVAL	**Mod Signal Value** Specifies the value to force onto the MOD signal.
7:5	///	**Reserved** Reading returns 0. Write the reset value.
4	EN0	**LCDVEEEN Output Enable** Specifies the general-purpose output enable to LCDVEEEN (only in Bypass Mode). 1 = LCDVEEEN signal enabled 0 = LCDVEEEN signal disabled
3	DISP	**Display Control Signal Output** Controls the output of the Display Control signal, LCDDSPLEN (only in Bypass Mode). 1 = LCDDSPLEN signal enabled 0 = LCDDSPLEN signal disabled
2	///	**Reserved** Reading returns 0. Write the reset value.
1	CLSEN	**CLS Enable** STN or TFT (Bypass) modes: Reserved Reading returns 0. Values written cannot be read. ALI mode: Enables or disables the generation of the LCDCLS (Gate Driver Clock) signal. 1 = LCDCLS signal enabled 0 = LCDCLS signal disabled
0	SPSEN	**SPS Enable** STN or TFT (Bypass) modes: Reserved Reading returns 0. Values written cannot be read. ALI mode: Enables or disables the generation of the LCDSPS (Row Reset) signal. 1 = LCDSPS signal is enabled 0 = LCDSPS signal is disabled

4.5.6.3 Timing Delay Register 1 (ALITIMING1)

The ALITIMING1 Register is used for various delays values for output signals. All delays are specified in number of LCD clock (LCDDCLK) periods.

Table 4-47. ALITIMING1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		MODDEL		PSCLS				REVDEL				LPDEL			
RESET	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE4000 + 0x008															

Table 4-48. ALITIMING1 Fields

BITS	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13:12	MODDEL	**LCDMOD LOW Delay** Controls the delay (number of LCDSPS rising edges) to hold LCDMOD LOW before transitioning HIGH. MODDEL = (LCDSPS rising edges) – 1
11:8	PSCLS	**LCDPS and LCDCLS Delay** Controls the delay in LCDDCLK periods from the first rising edge of the internal CLCDC clock after the leading edge of the internal LCDLP signal (not the LCDLP pin no. 137), to the leading edge of the LCDREV signal. The value of this field must be greater than 0. PSCLS = (LCDDCLK periods) – 1
7:4	REVDEL	**Polarity-Reversal Delay*** Controls the delay in LCDDCLK periods from the first rising edge of the internal CLCDC clock after the leading edge of the internal LCDLP signal (not the LCDLP pin no. 137), to the falling edge of the LCDREV signal. The value of this field must be greater than 0. REVDEL = (LCDDCLK periods) – 1
3:0	LPDEL	**LCDLP Delay** Controls the delay in LCDDCLK periods from the first rising edge of the internal CLCDC clock after the leading edge of the internal LCDLP signal (not the LCDLP pin no. 137), to the leading edge of the LCDLP signal. The value of this field must be greater than 0. LPDEL = (LCDDCLK periods) – 1

4.5.6.4 Timing Delay Register 2 (ALITIMING2)

The ALITIMING2 Register is used for various delay values for output signals. All delays are specified in number of LCD clock (LCDDCLK) periods.

Table 4-49. ALITIMING2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPLDEL							PS2CLS2								
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE4000 + 0x00C															

Table 4-50. ALITIMING2 Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:9	SPLDEL	**LCDSPL Delay** Controls the delay in LCDDCLK periods, from the rising edge of the internal CLCD clock after the leading edge of the internal LCDLP signal (not the LCDLP, pin 137), to the leading edge of the LCDSPL signal during the vertical front and back porches. This field must be programmed to a value greater than the sum of (TIMING0:HSW + TIMING0:HBP). SPLDEL = (LCDDCLK periods) – 1
8:0	PS2CLS2	**LCDSPL and LCDCLS Delay 2** Controls the delay in LCDDCLK periods from the first rising edge of the LCDSPL signal to the trailing edge of the LCDCLS and LCDPS signals. The value of this field must be greater than 0. PS2CLS2 = (LCDDCLK periods) – 1

4.6 Timing Waveforms

This section describes typical output waveform diagrams for the CLCDC and the ALI.

4.6.1 STN Horizontal Timing

Figure 4-6 shows typical horizontal timing waveforms for STN panels. In this figure, the CLCDC Clock (an input to the CLCDC) is scaled within the CLCDC and used to produce the LCDDCLK output. Programmable registers in the CLCDC set the timings (in terms of LCDDCLK pulses) to produce the other signals that control an STN display.

For example, Figure 4-6 shows that the duration of the LCDLP signal is controlled by TIMING0:HSW. Figure 4-6 also shows that the polarity of the LCDLP signal is set by TIMING2:IHS.

4.6.2 STN Vertical Timing

Figure 4-7 shows typical vertical timing waveforms for STN panels.

4.6.3 TFT Horizontal Timing

Figure 4-8 shows typical horizontal timing waveforms for TFT panels.

4.6.4 TFT Vertical Timing

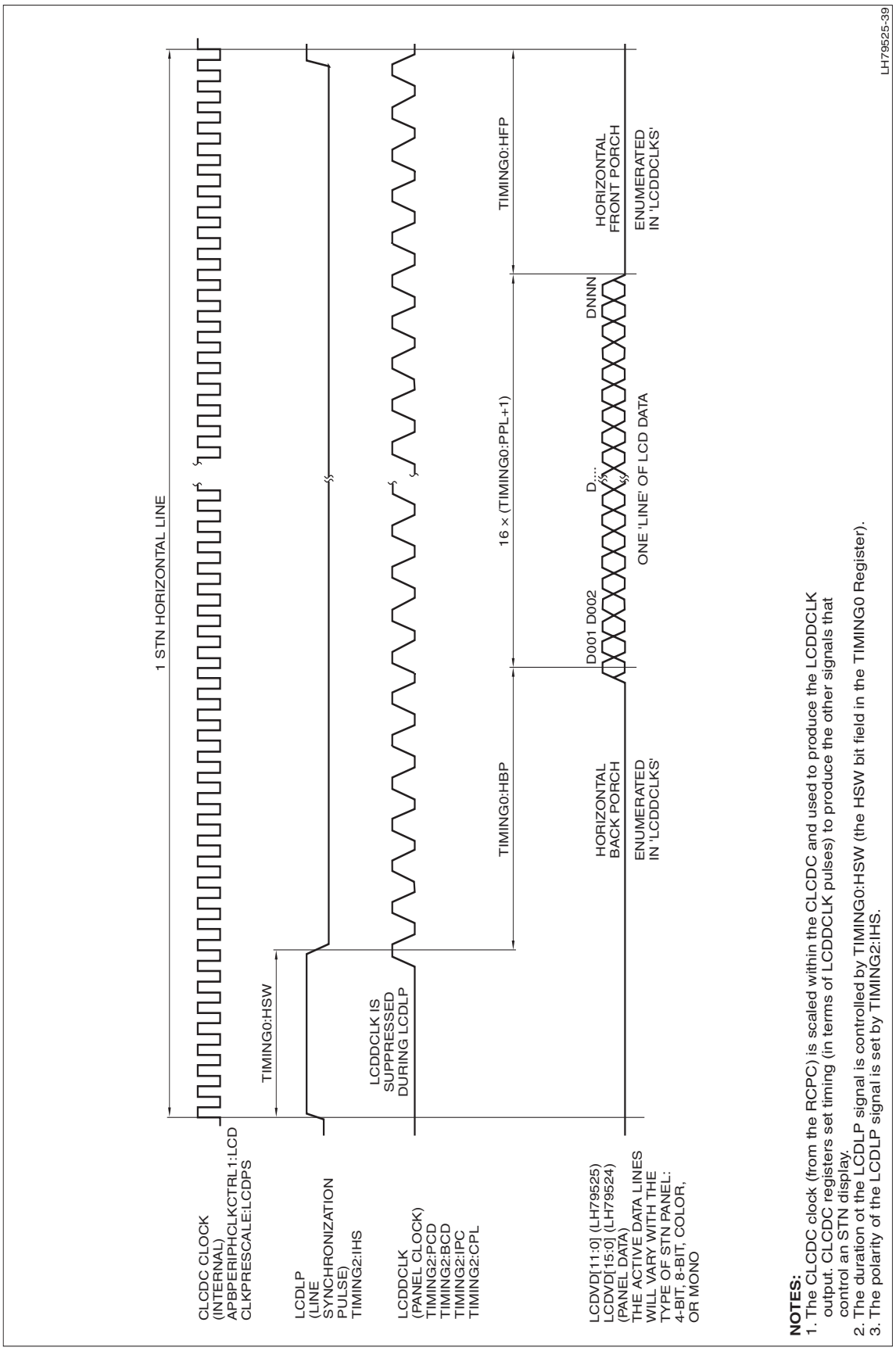
Figure 4-9 shows typical vertical timing waveforms for TFT panels.

4.6.5 AD-TFT/HR-TFT Horizontal Timing Waveforms

Figure 4-10 shows typical horizontal timing waveforms for AD-TFT and HR-TFT panels. The ALI adjusts the normal TFT timing to accommodate these panels.

4.6.6 AD-TFT/HR-TFT Vertical Timing Waveforms

Figure 4-11 shows typical vertical timing waveforms for AD-TFT and HR-TFT panels. The power sequencing and register information is the same as for TFT vertical timing.



NOTES:
1. The CLCDC clock (from the RCPC) is scaled within the CLCDC and used to produce the LCDDCLK output. CLCDC registers set timing (in terms of LCDDCLK pulses) to produce the other signals that control an STN display.
2. The duration ot the LCDLP signal is controlled by TIMING0:HSW (the HSW bit field in the TIMING0 Register).
3. The polarity of the LCDLP signal is set by TIMING2:IHS.

Figure 4-6. STN Horizontal Timing Diagram

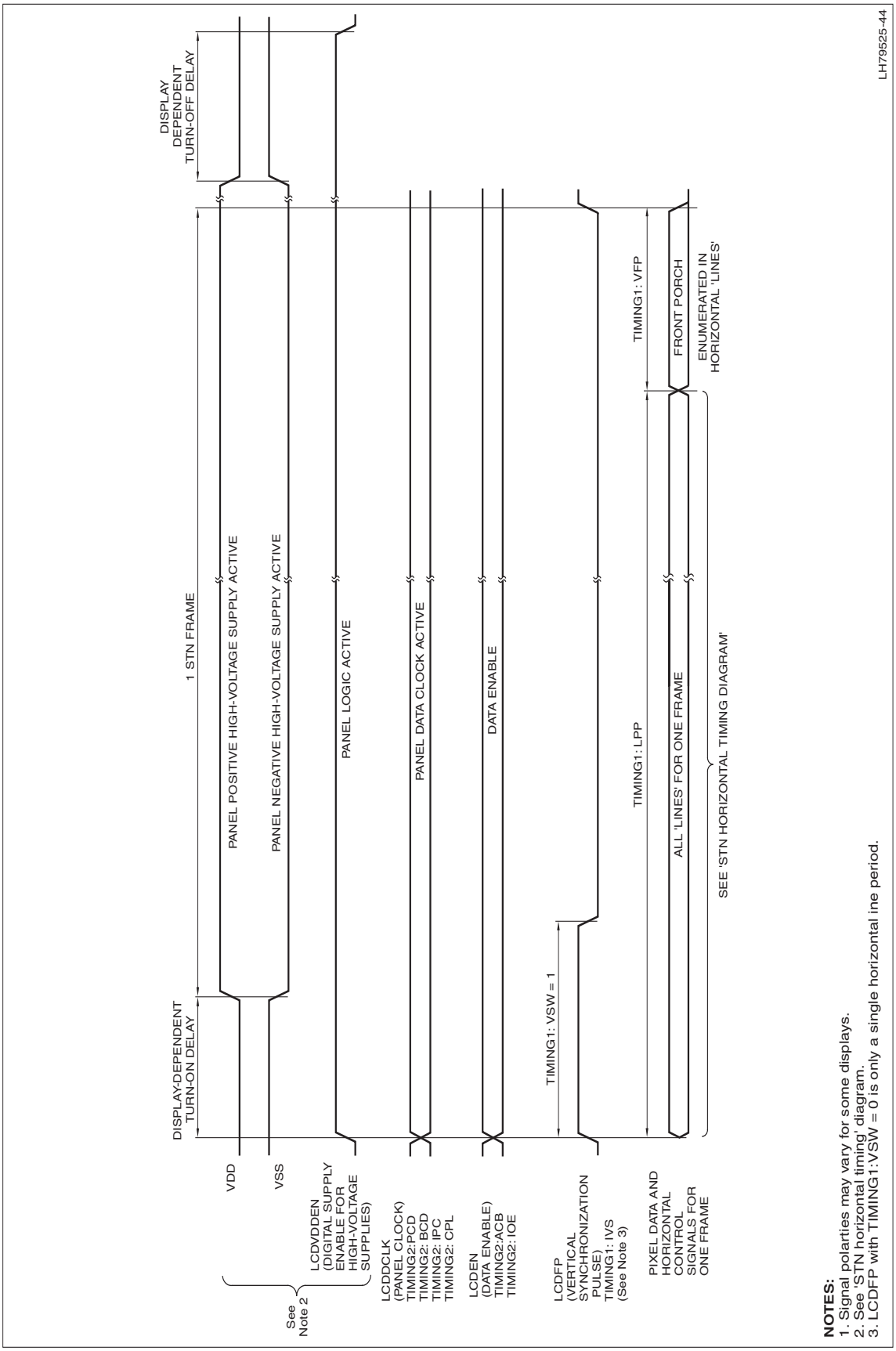


Figure 4-7. STN Vertical Timing Diagram

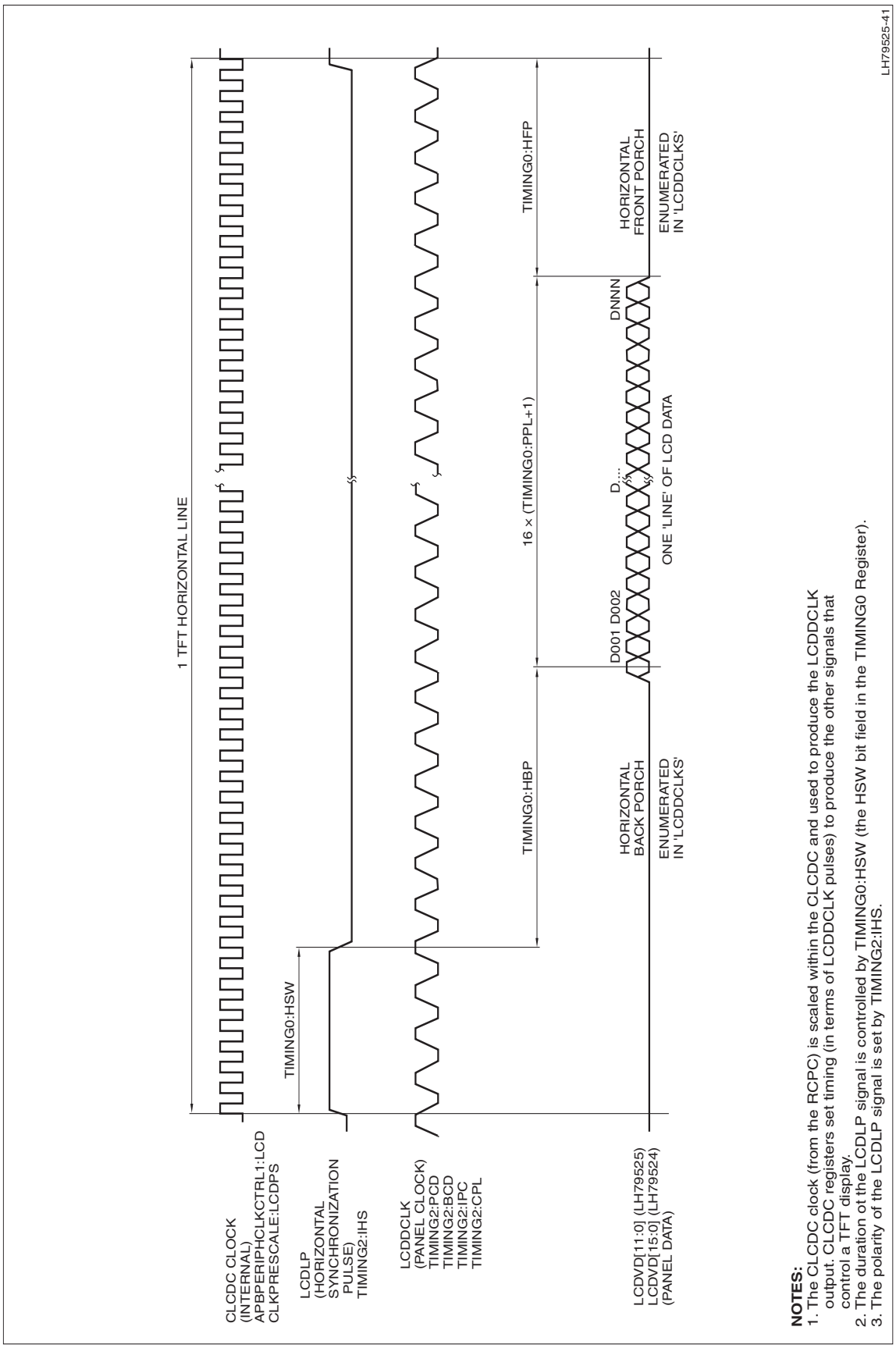


Figure 4-8. TFT Horizontal Timing Diagram

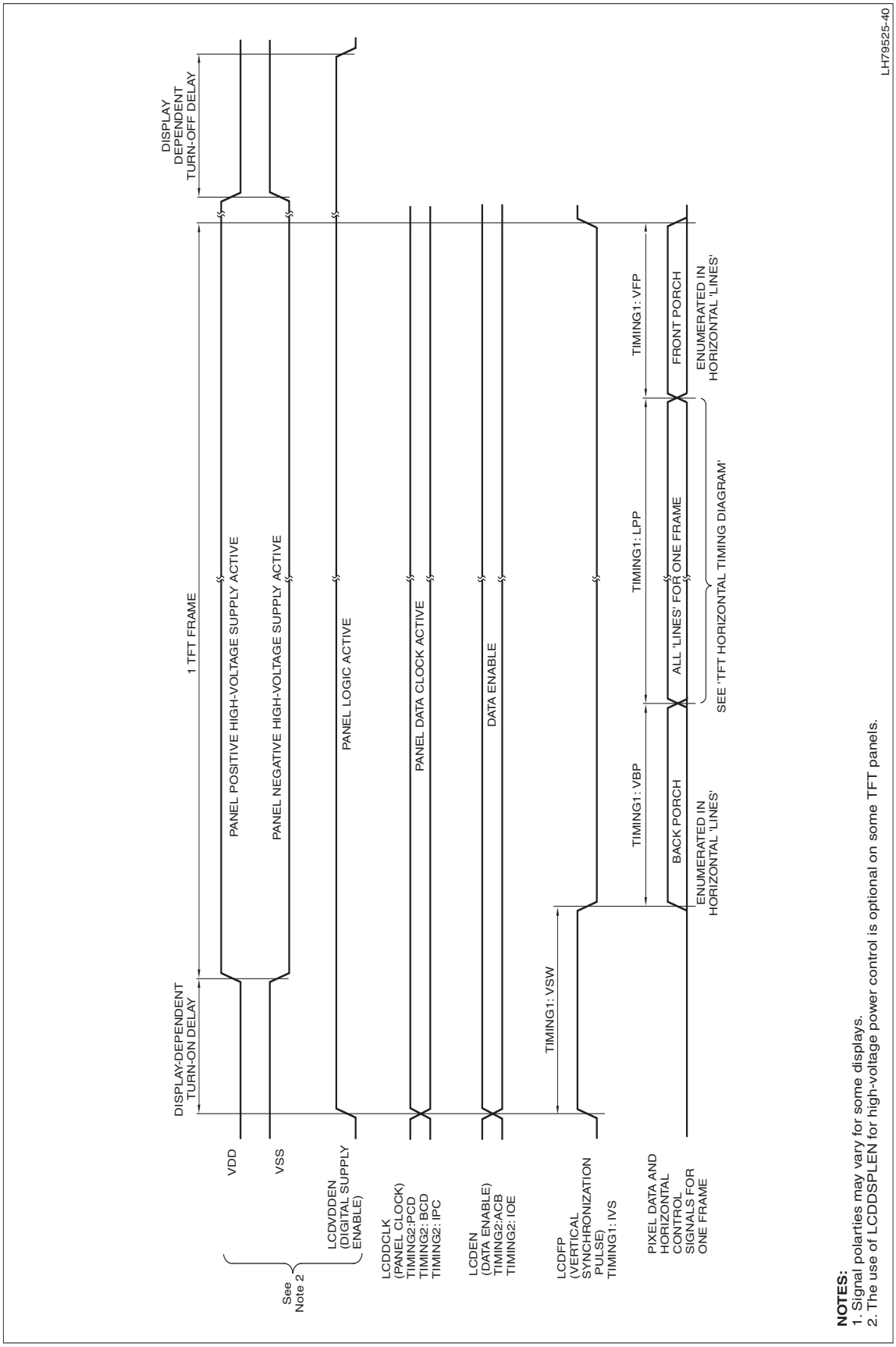


Figure 4-9. TFT Vertical Timing Diagram

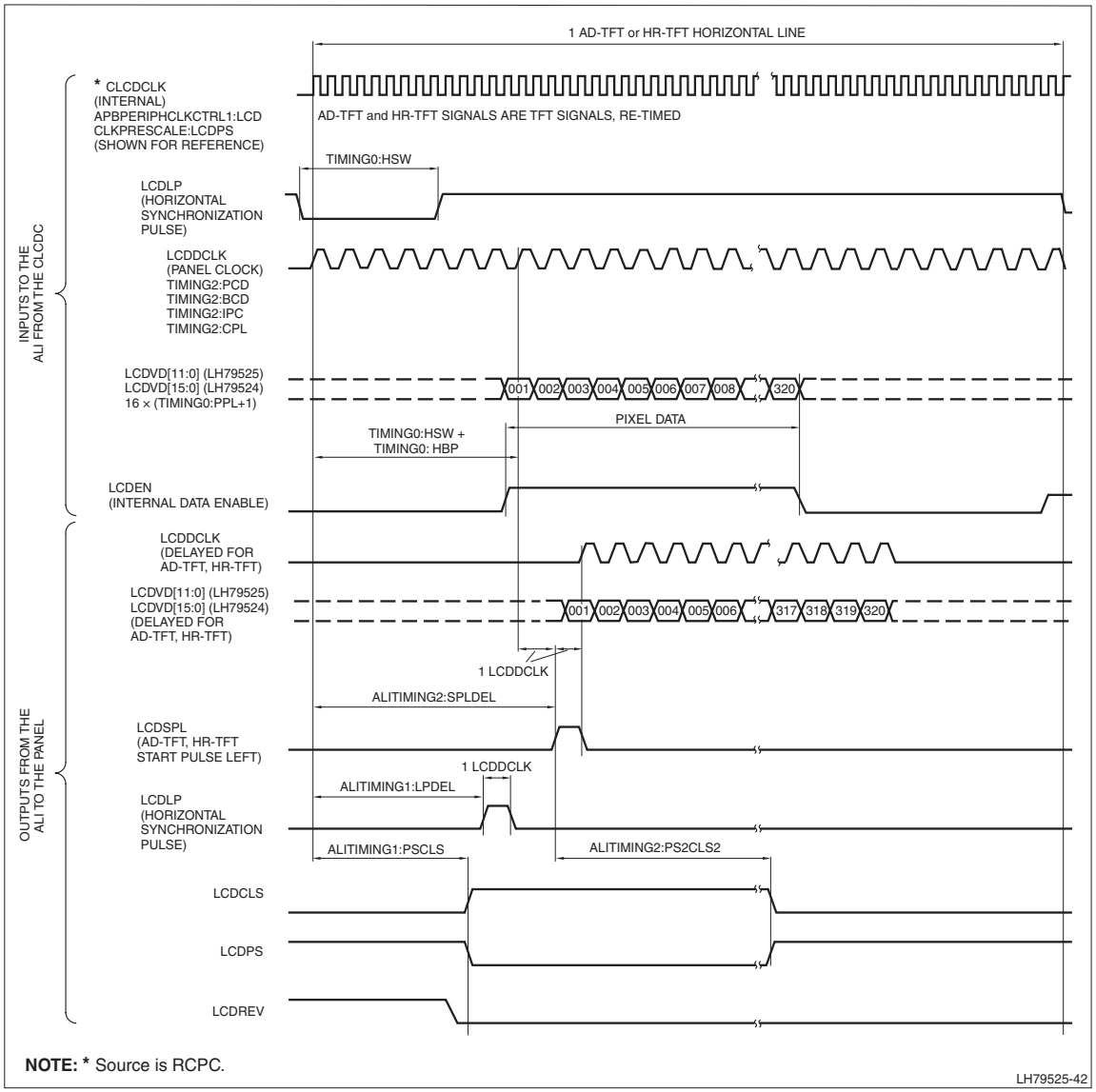


Figure 4-10. AD-TFT, HR-TFT Horizontal Timing Diagram

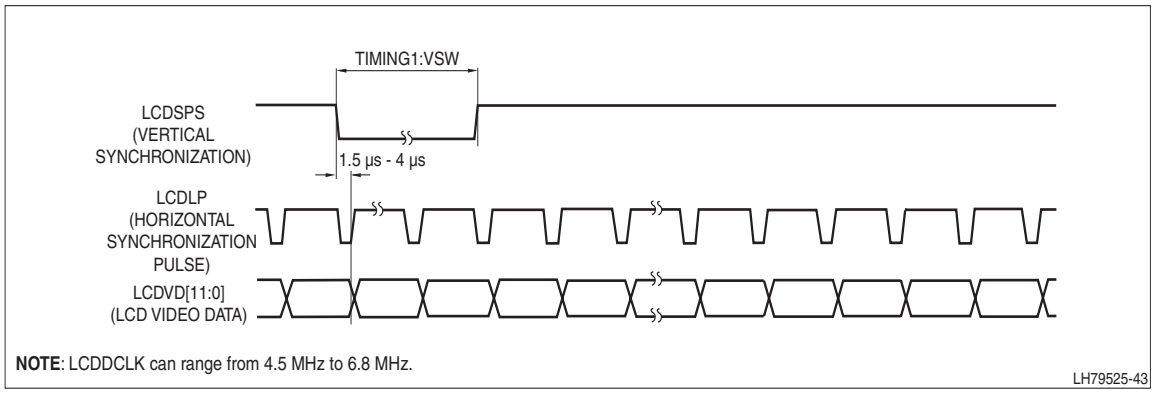


Figure 4-11. AD-TFT, HR-TFT Vertical Timing Diagram

Chapter 5
Direct Memory Access Controller

The DMA Controller in the LH79524/LH79525 is identical in each MCU; all descriptions in this chapter apply to both devices. The DMA Controller provides four concurrent data streams and three modes of transfer:

- Memory to Memory (selectable on Stream 3 only)
- Peripheral to Memory (all streams)
- Memory to Peripheral (all streams)

A built-in data stream arbiter handles all scheduling and conflict resolution. Each stream can alert the operating system of a transfer error via an interrupt. Table 5-1 lists the streams and their priority.

There are seven registers for each stream. Software uses these registers to program:

- DMA enable
- Transfer Size (Byte, Half-word, Word)
- Burst Size (1, 4, 8, or 16)
- Address Increment Enable
- Transfer Direction
- Maximum Count
- Terminal Count

DMA transactions use a 16-word First-In, First Out (FIFO) array, with pack and unpack logic to handle all input/output combinations of byte, half-word, and word transfers. In addition, there are external DMA Request (DREQ) and Acknowledge (DACK) signals synchronized with the External Memory Controller's write and read signals.

Any external peripheral using DREQ and DACK pins must be mapped into the nCS3 memory space for proper operation. See Section 1-6 for memory space mapping.

Table 5-1. DMA Controller Stream Assignments and Request Priority

DMA REQUESTOR	DMA STREAM
SSPRX (highest priority)	Stream 0
SSPTX	Stream 1
UART0RX/External DREQ	Stream 2
UART0TX (lowest priority)/Memory-to-Memory	Stream 3

5.1 Theory Of Operation

The MCU uses a central DMA Controller to service all DMA requirements for DMA-capable devices. The DMA Controller provides DMA support for the DMA-capable peripherals listed in Table 5-1. The DMA Controller has an APB slave port for programming its registers and an AHB port for data transfers. The DMA is controlled by the system clock.

NOTE: The DMA Controller is not used for the display system, Ethernet, or USB. The LCD Controller, Ethernet, and USB have their own DMA port that connects directly to the memory system to retrieve data.

The DMA Controller transfers data between AHB peripherals and memory or between memory and memory. The DMA Controller supports four data streams: Stream 0, Stream 1, Stream 2, and Stream 3. These data streams can be used to service:

- Four peripheral data streams (peripheral-to-memory or memory-to-peripheral)
- Three peripheral data streams and one memory-to-memory data stream.

The four data streams use a fixed-priority arbitration scheme and share one common 16-word-deep FIFO for buffering burst data. Each of the four data streams has its own independent set of DMA Registers and address/transfer count counters. In addition:

- Stream 2 provides a set of external signals for initiating and controlling DMA transfers between external peripherals and memory. The signals DREQ and DACK are brought out to external pins that are multiplexed with other functions. Note that any peripheral using the external DMA feature must be mapped into the nCS3 memory space.
- Stream 3 can conduct memory-to-memory DMA transfers under software control.

The DMA Registers are programmed through an APB Slave interface that has a 32-bit data interface.

A stream can be programmed to transfer from 1 to 65,535 data units. In this context, a data unit represents a group of bits equal in width to the data width of the source peripheral or memory. The source and destination data widths can be programmed independently to be byte, half-word, or word (byte only for DMA to/from UART0). Data is transferred in bursts, with the burst length programmable to 1, 4, 8, or 16 peripheral data units. The stream has source and destination address registers that can be independently programmed to remain fixed or increment after each data access.

The peripheral using the external DMA feature must be connected to nCS3. This is necessary to ensure that nDACK will be asserted.

The DMA process comprises:

1. The external request signal (DREQ) starts a peripheral DMA transfer.
2. The DMA Controller requests use of the AHB.
3. When the AHB arbiter grants the AHB to the DMA Controller, the DMA Controller fills its FIFO with the number of data units specified by the burst length (1, 4, 8, or 16).
4. The DMA Controller continues to request the AHB following the completion of the burst transfer. However, it may lose ownership of the AHB if a higher priority bus master is also requesting the AHB.
5. When the AHB arbiter re-grants the AHB to the DMA Controller, the FIFO empties (writes) its contents to the destination. The destination data width sets the width of this data to a byte, half-word or word. The filling and emptying of the FIFO for a burst transfer is always completed for the current stream being serviced before another stream DMA request is serviced.
6. As DMA requests are received, the DMA Controller arbitrates between them, assigning a requesting source to be serviced based on the priority indicated in Table 5-1. A data packet transfers from the source to the DMA FIFO, then transfers from the FIFO to the destination.

Exceptions to the DMA process are:

- When the DMA is configured to perform a memory-to-memory transfer followed by a peripheral-to-memory transfer, the transfer starts immediately, without the DMA waiting for the external request signal in step 1. The software workaround to this is:
  - Set up a memory-to-memory access.
  - Let the memory-to-memory access complete.
  - Execute up the peripheral-to-memory write, but without the enable bit set.
  - Perform a second write operation, with the enable bit set.
- When Stream 3 is used for memory-to-memory transfers, the transfer starts when software sets an enable bit in the Control Register for that stream. The transfer is conducted in bursts, with the bursts executing back-to-back until the required number of data units are transferred. The DMA Controller retains ownership of the AHB between successive bursts, unless the AHB Arbiter de-grants the DMA Controller for a higher priority bus master.

5.1.1 Use for SSP and UART

The SSP:DCR:RXDMAE bit, SSP:DCR:TXDMAE, UART:DMACTRL:RXDMAEN, and UART:DMACTRL:TXDMAEN bits are not automatically cleared for standard Stream 0 through 3 DMA operations, respectively. These bits should be explicitly cleared by software as soon as possible following DMA completion.

On initiating a DMA operation the DMAC:CTRL:ENABLE bit should be set before any of the above mentioned bits are set.

5.1.2 Changing Mode from Memory to Peripheral

When changing from a Memory-to-Memory (MTM) transfers to a Peripheral-to-Memory (either MTP or PTM transfers), the DMA Controller must be disabled before setting up the Peripheral transfer. Otherwise, the PTM or MTP transfer will start immediately, without a trigger.

For example, allow the MTM transfer to complete, then disable the DMA, configure the PTM transfer with the DMA disabled, then enable the transfer.

5.1.2.1 DMA Priority

When using Memory-to-Memory DMA, it has priority over *all other streams* (Streams 0,1, and 2). Thus, if a MTM DMA is started, no other stream can win arbitration until the entire MTM DMA completes. If a DMA in Stream 0, 1, or 2 is in progress and an MTM DMA is requested, the DMA arbiter will immediately conclude the DMA in progress and proceed to execute the MTM, then upon completion, resume the previous channel. It is advisable to keep MTM DMA sizes reasonably small to avoid choking the DMA channel, which could create system performance issues.

5.1.3 Interrupt, Error, and Status Registers

The DMA Controller provides Interrupt, Error, and Status Registers for controlling the generation of an interrupt, error-handling control, and active-stream monitoring. Each stream has its own interrupt flag, which is set after the last transfer completes. Each of the four interrupt flags can be masked and cleared independently.

Each stream also has its own error flag. An error flag is set when the data stream transfer is aborted due to an Error response from an AHB slave. Each of the four error flags can be separately masked and cleared. The masked interrupt and error flags are all combined into a single interrupt output.

5.1.3.1 Interrupts

The interrupt flags generated by the DMA Controller are combined and supplied to the Interrupt Controller as a combined interrupt. See the Vectored Interrupt Controller chapter for more information on interrupts.

5.1.4 External DMA Handshake Signal Timing

The basic signal timing for external DMA is illustrated in Figure 5-1. Additional timing is available in the Data Sheet.

- DREQ Timing: Once asserted, DREQ must not transition from LOW to HIGH again until after nDACK has been asserted.
- nDACK/DEOT Timing: Figure 5-1 indicates when nDACK and DEOT occur in relation to an external bus access to/from the external peripheral that requested the DMA transfer. This diagram shows the timing with relation to a single read or the last word of a burst read from the requesting peripheral.

nDACK will be extended by wait states (either programmed or forced via the nWAIT pin) in the same manner that the memory cycle is extended.

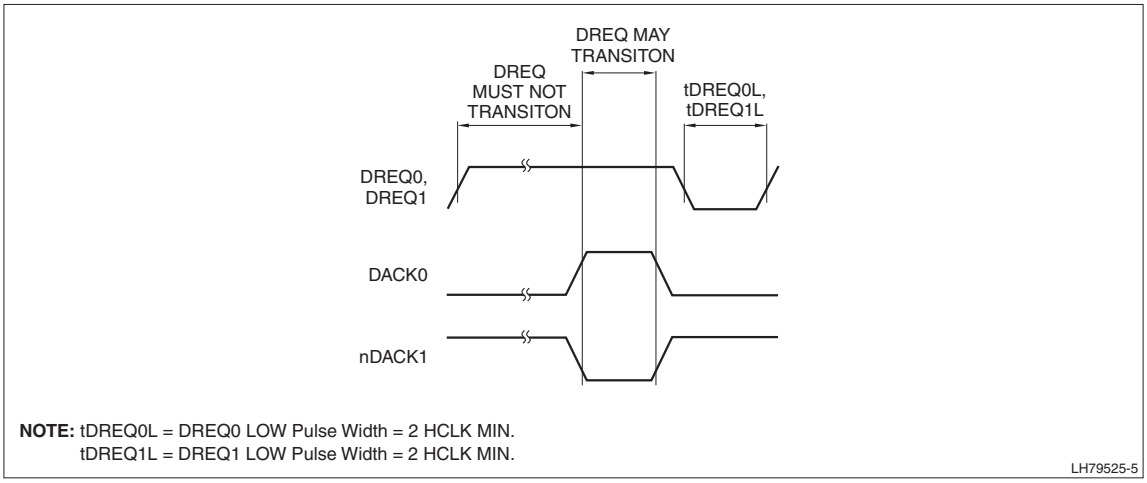


Figure 5-1. Basic DMA Timing

5.2 Register Reference

This section provides the DMA Controller register memory mapping and bit fields.

5.2.1 Memory Map

Each stream has the identical set of 11 registers. The base address for each stream is shown in Table 5-2. The 11 registers are summarized in Table 5-3. The address offset listed is with reference to the particular stream's base address, shown in Table 5-2. For example, the address for the DESTLO register for STREAM2 is:

(STREAM2 Base = 0xFFFE1080) + (DESTLO Offset = 0x008) = 0xFFFE1088.

Table 5-2. DMA Memory Map

DATASTREAM	BASE ADDRESS	DESCRIPTION
STREAM0	0xFFFE1000	Data Stream 0 Register Base Address
STREAM1	0xFFFE1040	Data Stream 1 Register Base Address
STREAM2	0xFFFE1080	Data Stream 2 Register Base Address
STREAM3	0xFFFE10C0	Data Stream 3 Register Base Address
MASK	0xFFFE10F0	DMA Interrupt Mask Register
CLR	0xFFFE10F4	DMA Interrupt Clear
STATUS	0xFFFE10F8	DMA Status Register

Table 5-3. DMA Data Stream Register Summary (One Set of Registers for Each of the Four Data Streams in Table 5-2)

ADDRESS OFFSET FROM STREAM BASE	NAME	DESCRIPTION
0x000	SOURCELO	Source Base Address Register, lower 16 bits
0x004	SOURCEHI	Source Base Address Register, higher 16 bits
0x008	DESTLO	Destination Base Address Register, lower 16 bits
0x00C	DESTHI	Destination Base Address Register, higher 16 bits
0x010	MAX	Maximum Count Register
0x014	CTRL	Control Register
0x018	CURSHI	Current Source Address Register, higher 16 bits
0x01C	CURSLO	Current Source Address Register, lower 16 bits
0x020	CURDHI	Current Destination Address Register, higher 16 bits
0x024	CURDLO	Current Destination Address Register, lower 16 bits
0x028	TCNT	Terminal Count
0x2C - 0x3C	///	Reserved — Do not access

5.2.2 Register Definitions

5.2.2.1 Source Base Registers (SOURCELO and SOURCEHI)

The two 16-bit Source Base Registers contain the 32-bit source base address for the next DMA transfer. When the DMA Controller is enabled, the contents of the Source Base Registers load into the Current Source Address Register.

Table 5-4. SOURCELO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SOURCELO															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x000															

Table 5-5. SOURCELO Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SOURCELO	**Low Order Source Address** This field contains the lower 16-bits of the address for the source of data for the next DMA transfer.

Table 5-6. SOURCEHI Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SOURCEHI															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x004															

Table 5-7. SOURCEHI Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SOURCEHI	**High Order Source Address** This field contains the upper 16-bits of the address for the source of data for the next DMA transfer.

5.2.2.2 Destination Base Registers (DESTLO and DESTHI)

The two 16-bit Destination Base Register contain the 32-bit destination base address for the next DMA transfer. When the DMA Controller is enabled, the contents of the Destination Base Address Registers load into the Current Destination Address Register.

Table 5-8. DESTLO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	DESTLO															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x008															

Table 5-9. DESTLO Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	DESTLO	**Low Order Destination Address** This field contains the lower 16-bits of the address for the destination of data for the next DMA transfer.

Table 5-10. DESTHI Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	DESTHI															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x00C															

Table 5-11. DESTHI Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	DESTHI	**High Order Destination Address** This field contains the upper 16-bits of the address for the destination of data for the next DMA transfer.

5.2.2.3 Maximum Count Register (MAX)

The Maximum Count Register must be programmed with the maximum number of data units of the next DMA transfer. A data unit equals the source-to-DMA data width (byte, half-word or word). When the DMA Controller is enabled, the content of the Maximum Count Register loads into the Terminal Count Register.

If the maximum count is programmed to 1, the DMA Controller performs a single transfer only and sets the terminal count. If the maximum count is programmed to 0, the DMA Controller does not perform any function.

The maximum terminal count is limited by a 16-bit value. ($2^{16} - 1$).

Table 5-12. MAX Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	MAXCOUNT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x010															

Table 5-13. MAX Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	MAXCOUNT	**Maximum Count** This field contains the maximum data unit count.

5.2.2.4 Control Register (CTRL)

The Control Register contains the configuration of the DMA Controller. Constraints on the field values based on the stream type are defined in Table 5-18. Where a value appears in this table, that is the *only* valid value for that stream, and the field must be programmed to this value.

Some caution must be advised when combining values for source size, destination size, burst size, and address incremeting when using different source and destination sizes. For example, programming SOSIZE=00 (byte), DESIZE=10 (word), SOBURST=00 (single) and DEINC=0 (non-incrementing destination) will cause a problem, despite the fact that all are allowed values. Although it is certainly a valid to have different Source/Destination sizes, and non-incrementing is correct for writing to a peripheral with a fixed-address word-size register, the SOBURST set to single will send the SOSIZE (one byte) in *four* separate transactions, each padding the top three bytes with zeros and placing the single byte in the lowest-significant byte position since the destination address remains the same. The correct way to execute this transaction is to have the SOBURST set to 01 (a burst of four), which will correctly fill the four bytes of the single word at the destination.

Table 5-14. CTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		DIR	///	M2M	///	ADMODE	DESIZE		SOBURST		SOSIZE		DEINC	SOINC	ENABLE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RW	RO	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE1000 + 0x14															

Table 5-15. CTRL Fields

BIT	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	DIR	**Peripheral is Source or Destination** 1 = Peripheral is the destination 0 = Peripheral is the source
12	///	**Reserved** Reading returns 0. Write the reset value.
11	M2M	**Stream 3 Memory Transfer** Selects memory-to-memory transfer for Stream 3. Ignored for data streams[2:0]. 1 = Stream 3 is configured for memory-to-memory transfer. The DMA Controller disregards any request from UART0TX and transfers data from source to destination as fast as possible until MaxCnt expires 0 = Stream 3 is not configured for memory-to-memory transfer
10	///	**Reserved** Reading returns 0. Write the reset value.

Table 5-15. CTRL Fields (Cont'd)

BIT	NAME	DESCRIPTION
9	ADMODE	**Current Source/Destination Loading** Determines whether the Current Source Address Register and the Current Destination Address Register load from the Source Base Registers and the Destination Base Registers, respectively, when the DMA Controller is enabled. 1 = Incremental Address Mode for source and destination. Registers are not reloaded from their respective Base Address Registers when the DMA Controller is enabled 0 = Wrapping Address Mode for source and destination. Registers load from their respective Base Address Registers when the DMA Controller is enabled. (default)
8:7	DESIZE	**DMA-to-Destination Data Width** Specifies the DMA-to-destination data width. See Table 5-16.
6:5	SOBURST	**Peripheral Burst Size** Defines the number of peripheral data units in the peripheral burst. Using the peripheral as the destination, the DMA interface automatically reads the correct number of source words to compile a transaction. When stream 3 is configured as a memory-to-memory transfer, SOBURST is the source-side burst length. Table 5-16 shows valid values.
4:3	SOSIZE	**Source-to-DMA Data Width** See Table 5-16. Note that Stream 0 (SSPRX) only supports half-word source data width.
2	DEINC	**Current Destination Register Increment** Enables the Current Destination Register increment after each DMA-to-destination data transfer. 1 = Current Destination Register is incremented 0 = Current Destination Register remains unchanged
1	SOINC	**Current Source Register Increment** Enables a Current Source Register increment after each source-to-DMA data transfer. 1 = Current Source Register increments as data transfers from a source to the DMA. The value increments at the end of the address phase of the AHB transfer 0 = Current Source Register remains unchanged, holding the same value during the entire DMA transfer
0	ENABLE	**DMA Controller Enable/Disable** Enables or disables the DMA Controller. The Source Base, Destination Base, and Maximum Count Registers must be set before the DMA is enabled. The state machine clears this bit when a data transfer finishes. If the software resets this bit during a transfer, that stream interface will be reset. 1 = DMA data transfer is enabled 0 = DMA data transfer is disabled

Table 5-16. DMA Data Width

SOSIZE/DESIZE	AHB DATA WIDTH
00	1 byte
01	1 half-word (2 bytes)
10	1 word (4 bytes)
11	Reserved

Table 5-17. DMA Burst Size

SOBURST	AHB BURST TYPE
00	Single
01	4
10	8
11	16

Table 5-18. Constraints on CTRL Field Values Based on Stream Type

STREAM TYPE	DESIZE	SOBURST	SOSIZE	DEINC	SOINC
SSPRX (Stream 0)	All valid values	00 or 01	00	1	0
SSPTX (Stream 1)	00	00 or 01	All valid values	0	1
UART0RX (Stream 2)	All valid values	— [1]	00	1	0
UART0TX (Stream 3)	00	— [2]	All valid values	0	1
MEM-to-MEM (Stream3)	All valid values	All valid values	All valid values	1	1

Notes:

1. If set to 00 or 01, no restrictions on the UART0 RX FIFO Watermark.
   If set to 10, the UART0 RX FIFO Watermark must be set to 1/4 or larger.
   If set to 11, the UART0 RX FIFO Watermark must be set to 1/2 or larger.
2. If set to 00 or 01, no restrictions on the UART0 TX FIFO Watermark.
   If set to 10, the UART0 TX FIFO Watermark must be set to 3/4 or smaller.
   If set to 11, the UART0 TX FIFO Watermark must be set to 1/2 or smaller.

5.2.2.5 Current Source Registers (CURSHI and CURSLO)

The Current Source Registers are 16-bit Read Only registers that hold the current value of the source address pointer. The value in the registers is used as an AHB address in a source-to-DMA data transfer over the AHB. If the CTRL:SOINC bit is programmed to 1, the value in the Current Source Registers increments as data transfers from a source to the DMA. The value increments at the end of the address phase of the AHB transfer by the HSIZE value. If the CTRL:SOINC bit is 0, the Current Source Register holds the same value during the entire DMA data transfer.

Table 5-19. CURSHI Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CURSHI															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x018															

Table 5-20. CURSHI Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	CURSHI	**Current Source Lower Address** This field contains the higher 16-bits of the address for the source of data for the current DMA transfer.

Table 5-21. CURSLO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CURSLO															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x01C															

Table 5-22. CURSLO Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	CURSLO	**Current Source Lower Address** This field contains the lower 16-bits of the address for the source of data for the current DMA transfer.

5.2.2.6 Current Destination Registers (CURDHI and CURDLO)

The Current Destination Registers are 16-bit Read Only registers that hold the current value of the destination address pointer. The value in the registers is used as an AHB address in a DMA-to-destination data transfer over the AHB. If the DeInc bit in the Control Register is set to 1, the value in the Current Destination Registers increments as data transfers from the DMA to a destination. The value increments at the end of the address phase of the AHB transfer by the HSIZE value. If the DeInc bit is 0, the Current Destination Register holds the same value during the entire DMA data transfer.

Table 5-23. CURDHI Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CURDHI															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x020															

Table 5-24. CURDHI Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	CURDHI	**Current Destination Upper Address** This field contains the upper 16-bits of the address for the destination of data for the current DMA transfer.

Table 5-25. CURDLO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CURDLO															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x024															

Table 5-26. CURDLO Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	CURDLO	**Current Destination Lower Address** This field contains the lower 16-bits of the address for the destination of data for the current DMA transfer.

5.2.2.7 Terminal Count Register (TCNT)

The Terminal Count Register is a 16-bit Read Only register that contains the number of data units remaining in the current DMA transfer. The data unit is equal to the source-to-DMA data width (byte, half-word or word).

The register value is decremented every time data transfers to the DMA FIFO. When the terminal count reaches zero, the FIFO content transfers to the destination and a DMA transfer is finished.

Table 5-27. TCNT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TERM															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	DATASTREAM x BASE + 0x028															

Table 5-28. TCNT Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	TERM	**Terminal Count** This field contains the number of data units remaining in the current DMA transfer. The data unit is the source-to-DMA data width (byte, half-word or word).

5.2.2.8 Interrupt Mask Register (MASK)

The MASK Register allows enabling and disabling (masking) DMA interrupts. Program with a 1 to enable, and a 0 to disable individual interrupts.

Table 5-29. MASK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								MASKE3	MASKE2	MASKE1	MASKE0	MASK3	MASK2	MASK1	MASK0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE1000 + 0x0F0															

Table 5-30. MASK Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	MASKE3	**Data Stream 3 Error Interrupt** 1 = Enables data stream 3 error interrupt 0 = Disables data stream 3 error interrupt
6	MASKE2	**Data Stream 2 Error Interrupt** 1 = Enables data stream 2 error interrupt 0 = Disables data stream 2 error interrupt
5	MASKE1	**Data Stream 1 Error Interrupt** 1 = Enables data stream 1 error interrupt 0 = Disables data stream 1 error interrupt
4	MASKE0	**Data Stream 0 Error Interrupt** 1 = Enables data stream 0 error interrupt 0 = Disables data stream 0 error interrupt
3	MASK3	**Data Stream 3 Interrupt** 1 = Enables data stream 3 interrupt 0 = Disables data stream 3 interrupt
2	MASK2	**Data Stream 2 Interrupt** 1 = Enables data stream 2 interrupt 0 = Disables data stream 2 interrupt
1	MASK1	**Data Stream 1 Interrupt** 1 = Enables data stream 1 interrupt 0 = Disables data stream 1 interrupt
0	MASK0	**Data Stream 0 Interrupt** 1 = Enables data stream 0 interrupt 0 = Disables data stream 0 interrupt

5.2.2.9 Interrupt Clear Register (CLR)

The Interrupt Clear Register clears the status flags. Writing a 1 to a bit clears the interrupt status bit in the STATUS register. This register has an indeterminate value after Reset.

Table 5-31. CLR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								CLEARE3	CLEARE2	CLEARE1	CLEARE0	CLEAR3	CLEAR2	CLEAR1	CLEAR0
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFE1000 + 0x0F4															

Table 5-32. CLR Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	CLEARE3	**Clear ErrorInt3 Flag** 1 = Clears the ERRORINT3 interrupt flag in the Status Register 0 = No effect
6	CLEARE2	**Clear ErrorInt2 Flag** 1 = Clears the ERRORINT2 interrupt flag in the Status Register 0 = No effect
5	CLEARE1	**Clear ErrorInt1 Flag** 1 = Clears the ERRORINT1 interrupt flag in the Status Register 0 = No effect
4	CLEARE0	**Clear ErrorInt0 Flag** 1 = Clears the ERRORINT0 interrupt flag in the Status Register 0 = No effect
3	CLEAR3	**Clear Int3 Flag** 1 = Clears the INT3 interrupt flag in the Status Register 0 = No effect
2	CLEAR2	**Clear Int2 Flag** 1 = Clears the INT2 interrupt flag in the Status Register 0 = No effect
1	CLEAR1	**Clear Int1 Flag** 1 = Clears the INT1 interrupt flag in the Status Register 0 = No effect
0	CLEAR0	**Clear Int0 Flag** 1 = Clears the INT0 interrupt flag in the Status Register 0 = No effect

5.2.2.10 Status Register (STATUS)

The STATUS Register provides status information about the DMA Controller interrupts. The interrupt status bits are cleared by writing to the CLR register.

The INT[3:0] bits are the data stream interrupt flags corresponding to data stream 0 through data stream 3. A data stream sets its corresponding interrupt flag when a data transfer is completed (a complete packet has been transferred to its destination).

The ERRORINT[3:0] bits are the Error interrupts corresponding to data stream 0 through data stream 3. An error interrupt status is set when its corresponding data stream's transfer aborts due to an AHB transfer error. When this occurs, the stream is disabled until software sets the Enable bit again.

The Active flags indicate whether a data stream is transferring data. It is HIGH if a data transfer is in progress. The Active flags have the same polarity as the Enable bits in the Data Stream Control Register.

Table 5-33. STATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				ACTIVE3	ACTIVE2	ACTIVE1	ACTIVE0	ERRORINT3	ERRORINT2	ERRORINT1	ERRORINT0	INT3	INT2	INT1	INT0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE1000 + 0x0F8															

Table 5-34. STATUS Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11	ACTIVE3	**Data Stream 3 Active/Inactive** 1 = Data stream 3 is active 0 = Data stream 3 is not active
10	ACTIVE2	**Data Stream 2 Active/Inactive** 1 = Data stream 2 is active 0 = Data stream 2 is not active
9	ACTIVE1	**Data Stream 1 Active/Inactive** 1 = Data stream 1 is active 0 = Data stream 1 is not active
8	ACTIVE0	**Data Stream 0 Active/Inactive** 1 = Data stream 0 is active 0 = Data stream 0 is not active

Table 5-34. STATUS Fields (Cont'd)

BIT	NAME	DESCRIPTION
7	ERRORINT3	**Data Stream 3 Error Interrupt** Contains the status of the data stream 3 error interrupt. 1 = Error-interrupt asserted 0 = Error-interrupt not asserted
6	ERRORINT2	**Data Stream 2 Error Interrupt** Contains the status of the data stream 2 error interrupt. 1 = Error-interrupt asserted 0 = Error-interrupt not asserted
5	ERRORINT1	**Data Stream 1 Error Interrupt** Contains the status of the data stream 1 error interrupt. 1 = Error-interrupt asserted 0 = Error-interrupt not asserted
4	ERRORINT0	**Data Stream 0 Error Interrupt** Contains the status of the data stream 0 error interrupt. 1 = Error-interrupt asserted 0 = Error-interrupt not asserted
3	INT3	**Data Stream 3 Interrupt Flag** 1 = Interrupt Flag is active 0 = Interrupt Flag is not active
2	INT2	**Data Stream 2 Interrupt Flag** 1 = Interrupt Flag is active 0 = Interrupt Flag is not active
1	INT1	**Data Stream 1 Interrupt Flag** 1 = Interrupt Flag is active 0 = Interrupt Flag is not active
0	INT0	**Data Stream 0 Interrupt Flag** 1 = Interrupt Flag is active 0 = Interrupt Flag is not active

Chapter 6
Ethernet MAC Controller

The on-board Ethernet Media Access Controller (EMAC) is compatible with IEEE 802.3, and has passed the University of New Hampshire (UNH) testing. It supports both 10- and 100-Mbit/s transfer rates, and full and half duplex operation. Other features include:

- Transmit and receive FIFOs
- Media Independent Interface (MII) to the physical layer
- Physical layer management through Management Data I/O (MDIO) interface
- Support for:
  - half-duplex flow control by forcing collisions on incoming frames
  - full-duplex flow control with recognition of incoming pause frames and hardware generation of transmitted pause frames
  - 802.Q VLAN tagging with recognition of incoming VLAN and priority tagged frames
  - multiple buffers per receive and transmit frame
- APB interface
- AHB bus master DMA interface to external system memory
- Interrupt generation for receive and transmit completion
- Statistics counter registers for Remote Monitoring (RMON) and Management Information Base (MIB)
- Automatic pad and CRC generation on transmitted frames
- Automatic discarding of frames received with errors
- Address checking logic supports up to four specific (hardware) 48-bit addresses
- Promiscuous mode support where all valid received frames are copied to memory
- Hash matching of unicast and multicast destination addresses
- Serial network interface operation
- Software programmable MAC address
- Jumbo frames of up to 10,240 bytes supported.

6.1 Theory of Operation

This chapter assumes working knowledge of Ethernet protocol and the IEEE 802.3 specification. The full specification can be obtained at: http://standards.ieee.org/getieee802/

A simplified block diagram of the EMAC appears in Figure 6-1. It is handy to reference the block diagram as the different interfaces are described in the succeeding sections. The EMAC is identical for both the LH79524 and LH79525, and all descriptions in this chapter apply to both devices. Following the Theory of Operation section is a programming example.

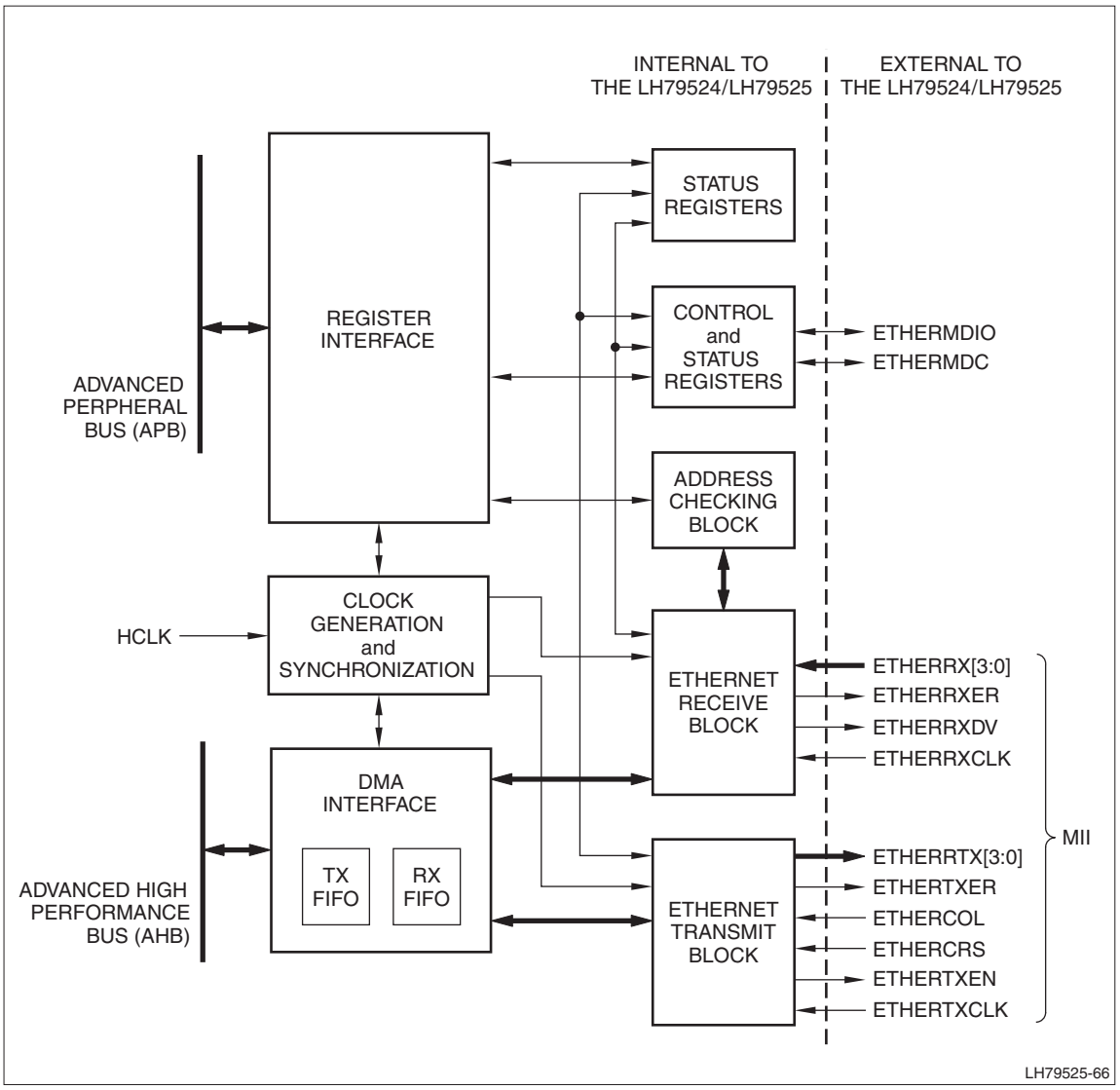


Figure 6-1. EMAC Block Diagram

6.1.1 Operational Overview

The Ethernet Receive Block and Ethernet Transmit Block contain the logic to implement receive and transmit operations at either 10 Mbits/s or 100 Mbits/s. These two blocks implement a Media Independent Interface (MII) for High Speed Ethernet, and Attachment Unit Interface (AUI) for 10 Mbits/s Ethernet (throughout the rest of the document, 'MII' will be used to mean both MII and AUI). The MII allows seamless connection to Physical Layer devices (PHYs).

The Receive and Transmit Blocks can operate independently of one another, thus allowing the EMAC to provide both half-duplex and full-duplex operation. During half-duplex operation, the EMAC can be programmed so that any frame received while the EMAC is transmitting is discarded. Flow control during reception is handled by transmitting null frames to intentionally cause collisions.

The Receive and Transmit Blocks also handle error checking, collision detection, carrier sense, and CRC/FCS generation and decoding. Full CSMA/CD protocol is implemented. Error frames are automatically discarded without being written to the system memory buffers.

Up to four MAC addresses can be programmed into the EMAC. The MAC address(es) must be stored in non-volatile memory off-chip, and read by software upon initiatlization. Software programs the MAC address(es) into the Address Specific register(s), and also must use it to build transmit frames.

Incoming packets are checked for error and validity in the Receive Block. Then, the Address Checking Block compares addresses of received packets to the addresses stored in the Specific Address Registers, and upon match, copies the packet to the receive buffer.

Transmit frame data is assembled by software in buffers, which are then retrieved, checked for validity, and transmitted over the Ethernet connection.

Receive and transmit data is stored in buffers allocated in areas in system memory defined by software. The EMAC maintains a pointer to these buffers and automatically handles pointer indexing. The onboard DMA Interface is an AHB master, and can directly read and write data in system memory, allowing data handling fast enough to support High Speed Ethernet at 100 Mbits/s. Clocking is handled by onboard synchronization circuits, and the divisor is programmable to allow a wide range of HCLK frequencies.

The Statistics and Control Registers are accessed by the core via the APB, through the Register Interface. The Statistics Registers log a wide range of network statistics for use by the LH79524/LH79525 software, RMON/MIB, or other uses. The control registers allow software to define network parameters, enable and disable the Receive and Transmit Blocks, enable and disable interrupts, view status information, and also implement the MDIO interface to manage the PHY.

6.1.1.1 Setup

Prior to use, software must allocate system resources, initialize, and program the EMAC. Receive and Transmit buffers must be allocated in system memory. Software must also load the MAC address into Specific Address Register 1, initialize the Buffer Pointer Queue Registers, set up the network parameters in the Network Control Register and Network Configuration Register, and enable any applicable interrupts in the Interrupt Enable Register. Once these housekeeping details have been completed, the transmitter and receiver can be enabled.

6.1.1.2 Statistics

As network communication transpires, statistics are maintained in the set of Statistics Registers. These registers can be interrogated at any time by software. Upon reading, the count value is reset to zero. If the registers are not read before reaching the maximum count value, they will 'stick' at all ones, indicating an overflow has occurred since the register was last read.

6.1.1.3 Detailed Descriptions

The following sections provide detailed information about the functional blocks within the EMAC, buffers, DMA, and specific operation.

6.1.2 Memory Interface

Ethernet frame data is stored in LH79524/LH79525 system memory. It is transferred to and from the Ethernet MAC through the DMA interface. All transfers are 32-bit words and may be single accesses, or bursts of 2, 3, or 4 words (transfers for the LH79525 are automatically parsed into two 16-bit transfers to accommodate its 16-bit data bus). Four-word bursts are the default data transfer, however single accesses, or bursts fewer than four words may be used to transfer data at the beginning or the end of a buffer. Burst accesses do not cross 16-byte boundaries.

6.1.2.1 FIFO

The receive and transmit FIFO depths are both set to 16 entries.

Data is typically transferred to and from the onboard FIFOs in bursts of four words. For receive, a bus request is asserted when the FIFO contains four words and has space for more. For transmit, a bus request is generated when there is space for four words, or when there is space for two words if the next transfer is to be only one or two words.

6.1.2.2 Receive Buffers

As each valid frame is received, it is stored in a 128-byte receive buffer. The start location for each receive buffer is stored in memory in a list of receive buffer descriptors at a location pointed to by the Receive Buffer Queue Pointer register (RXBQP). The receive buffer start location is a word address which can be offset by up to three bytes depending on the value written to the Network Configuration register (NETCONFIG:RXBUFOS). If the start location of the buffer is offset, the available length of the first buffer of a frame is reduced by the corresponding number of bytes.

Each Receive Buffer Descriptor List entry comprises two words. The first word contains only the address of the receive buffer; the second word contains the receive status. If the length of a receive frame exceeds the buffer length, the status word for the used buffer is written with zeroes except for the Start Of Frame bit and the offset bits, if appropriate. A 1 in bit zero of the address field indicates that the buffer has been used. The receive buffer manager then reads the location of the next receive buffer and fills that with receive frame data. The final buffer descriptor status word contains the Complete Frame status. Table 6-1 provides the details of the Receive Buffer Descriptor List. For each status bit, 1 = TRUE and 0 = FALSE.

Table 6-1. Receive Buffer Descriptor LIst

BIT	DESCRIPTION
	WORD 0
31:2	**Start Address** Address of the beginning of buffer
1	**Wrap** A 1 indicates the last descriptor in receive buffer descriptor list.
0	**Ownership** Needs to be 0 for the Ethernet MAC to write data to the receive buffer. The Ethernet MAC sets this to 1 once it has successfully written a frame to memory. Software must clear this bit before the buffer can be used again.
	WORD 1
31	**Global All Ones Broadcast Address Detected**
30	**Multicast Hash Match**
29	**Unicast Hash Match**
28	**External Address Match**
27	**Reserved** Reading returns 0. Write the reset value.
26	**Specific Address Register 1 Match**
25	**Specific Address Register 2 Match**
24	**Specific Address Register 3 Match**
23	**Specific Address Register 4 Match**
22	**Type ID Match**
21	**VLAN Tag Detected** Type ID 0x8100
20	**Priority Tag Detected** Type ID 0x8100 and null VLAN identifier
19:17	**VLAN Priority** Only valid if bit 21 is set
16	**Concatenation Format Indicator** Only valid if bit 21 is 1
15	**End Of Frame** When 1, the buffer contains the end of a frame. If end of frame is not 1 then the only other valid status bits are 12, 13, and 14.
14	**Start Of Frame** When 1 the buffer contains the start of a frame. If both bits 15 and 14 are 1, the buffer contains an entire frame.
13:12	**Receive Buffer Offset** Indicates the number of bytes by which the data in the first buffer is offset from the word address. Updated with the current values of the network configuration register. If Jumbo Frame Mode is enabled through NETCONFIG:JUMBOFRM, these bits are used as the most-significant bits of Length Of Frame.
11	**Reserved** Reading returns 0. Write the reset value.
10:0	**Length Of Frame** Length includes FCS (if selected). Bits 13:12 are also used if Jumbo Frame Mode is selected.

To receive frames, the buffer descriptors must be initialized by writing an appropriate address to bits [31:2] in the first word of each list entry. Bit zero must be written with 0. Bit one is the wrap bit and indicates the last entry in the list.

The start location of the Receive Buffer Descriptor List must be written to the RXBQP register before programming the Receive Enable bit in the network control register (NETCTL:RXEN) to enable receive. As soon as the receive block starts writing received frame data to the receive FIFO, the receive buffer manager reads the first receive buffer location pointed to by RXBQP.

If the Address Checking Block indicates that the frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

If the current buffer pointer has its Wrap bit set, or it is the 1,024th descriptor, the next receive buffer location is read from the beginning of the Receive Buffer Descriptor List. Otherwise, the next receive buffer location is read from the next consecutive word in memory.

The RXBQP register increments with each successful read, which indexes the 2,048 word locations of a maximum length Receive Buffer Descriptor List. Reading the RXBQP register returns the pointer value, which is the list entry currently being accessed. The value written to the RXBQP register may be any word-aligned address, provided that there are at least 2,048 word locations available between the pointer and the top of memory.

The AMBA 2.0 specification requires that bursts not cross 1KB boundaries. As receive buffer manager Write functions are two-word bursts, the RXBQP register should be programmed with the three least-significant bits as 0.

As each receive buffer is used, the receive buffer manager programs the Used bit of the first descriptor word to 1 to indicate that buffer has been used. If a receive error is detected, the receive buffer currently being written will be recovered. Previous buffers will not be recovered. Software should search through the Used bits in the buffer descriptors to determine how many frames have been received and not rely on the value returned by the RXBQP register, which changes continuously as more buffers are used.

If the statistics registers indicate that CRC errors, excessive length frames, or length field mismatched frames have been encountered, a frame fragment may have been stored in a sequence of frame buffers. Software can detect this by looking for the Start Of Frame bit set in a buffer following a buffer with no End Of Frame bit set. However, in a properly working Ethernet system, there should be no excessive length frames or frames greater than 128 bytes with CRC/FCS errors. Collision fragments will be less than 128 bytes long. Therefore, finding a frame fragment in a receive buffer is rare.

If the Used bit is 1 when the receive buffer manager reads the location of the receive buffer, the buffer has been already used and cannot be used again until software has processed the frame and cleared the Used bit. In this case, the DMA Block will set the buffer not available bit in the Receive Status register (RXSTATUS) and trigger an interrupt. If the Used bit is 1 when the receive buffer manager reads the location of the receive buffer, and a frame is being received, that frame will be discarded and the Receive Resource Error statistics register (RXRERR) will be incremented.

A Receive Overrun condition occurs when either the AHB bus was not granted in time or because the response was 'Not OK'. In a Receive Overrun condition, the Receive Overrun Interrupt is asserted and the buffer currently being written is recovered. The next received frame whose address is recognized reuses the buffer.

If NETCONFIG:DISCARDFCS is 1, the FCS bytes of received frames are not copied to memory. The Length Of Frame field is reduced by four bytes in this case.

6.1.2.3 Transmit Buffer

One or more transmit buffers store frames pending transmission. Transmit buffers can be between 0 and 2,047 bytes long, so it is possible to transmit frames longer than the maximum length specified in IEEE standard 802.3. Zero length buffers are also allowed. The maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit buffer is stored in memory in a Transmit Buffer Descriptor List at a location pointed to by the Transmit Buffer Queue Pointer register (TXBQP). LIke the receive buffers, each list entry consists of two words. The first word contains only the byte address of the transmit buffer; the second word contains the transmit control and status. Frames can be transmitted with or without automatic CRC generation. If CRC is automatically generated, a pad will also be automatically generated to make frames a minimum length of 64 bytes. Table 6-2 defines the transmit buffer descriptor list. For each status bit, 1 = TRUE and 0 = FALSE.

Table 6-2. Transmit Buffer Descriptor List

BIT	FUNCTION
WORD 0	
31:0	**Byte Address Of Buffer**
WORD 1	
31	**Used** Must be zero for the EMAC to send data to the transmit buffer. The EMAC programs this bit to 1 for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again.
30	**Wrap** Marks last descriptor in transmit buffer descriptor list.
29	**Retry Limit Exceeded** Too many retries were made without successful transmission.
28	**Transmit Underrun** This error occurs either when response is not OK or the transmit data could not be fetched in time or when buffers are exhausted in mid-frame.
27	**Buffers Exhausted In Mid-Frame** The buffers ran out of data before the entire frame could be transmitted.
26:17	**Reserved** Reading returns 0. Write the reset value.
16	**No CRC** When programmed to 1, no CRC will be appended to the current frame. This bit only needs to be programmed for the last buffer of a frame.
15	**Last Buffer** Indicates the last buffer in the current frame has been reached.
14:11	**Reserved** Reading returns 0. Write the reset value.
10:0	**Length Of Buffer** The length of the buffer is programmed to this field.

Before transmitting frames, the buffer descriptors must be initialized by writing an appropriate address to bits [31:0] in the first word of each list entry. The second transmit buffer descriptor is initialized with control information that indicates the length of the buffer, whether or not it is to be transmitted with a CRC, and whether the buffer is the last buffer of the frame.

After transmission, the control bits are written back to the second word of the first buffer along with the Used bit and other status information. The Used bit is written as 1 when a frame has been transmitted. Bits 27, 28, and 29 indicate various transmit error conditions. Bit 30 is the Wrap bit which can be set for any buffer within a frame. If no Wrap bit is encountered, after 1,024 descriptors the queue pointers roll over to the start.

The TXBQP register must not be written while transmit is active. If a new value is written, the queue pointer resets itself to point to the beginning of the new queue. If transmit is disabled by writing to NETCTL:TXEN, the TXBQP register resets to point to the beginning of the transmit queue. Note that disabling receive does not have the same effect on the receive queue pointer.

Once the transmit queue is initialized, transmit is activated by writing to the NETCTL:STARTTX bit. Transmit is halted when a buffer descriptor with its Used bit set is read, if a transmit error occurs, or by writing to the NETCTL:TXHALT bit. Transmission is suspended if a pause frame is received while the NETCONFIG:PAUSEEN bit is 1. Rewriting the start bit while transmission is active is allowed. This is implemented with the TXSTATUS:TXGO bit.

The TXGO bit is reset when:

- Transmit is disabled
- A buffer descriptor with its Used bit set is read
- A new value is written to the TXBQ register
- NETCTL:TXHALT is written
- There is a transmit error.

To set TXGO write NETCTL:STARTTX. Transmit halt does not take effect until any ongoing transmit finishes.

If a collision occurs during transmission of a multi-buffer frame, transmission will automatically restart from the first buffer of the frame. If a Used bit is read midway through transmission of a multi-buffer frame, it is treated as a transmit error. Transmission stops, ETHERTXER is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a Used bit being read, transmission will restart from the first buffer descriptor of the frame being transmitted when the transmit start bit is rewritten.

6.1.3 Receive Block

The Receive Block checks for a valid preamble, FCS, alignment, and length; presents received frames to the DMA Block; and stores the frame's destination address for use by the Address Checking Block.

During frame reception, if the frame is found to be too long or the Receive Error (ETHERRXER) pin is asserted, a bad frame indication is sent to the DMA Block. The DMA Block then ceases sending data to memory.

At the end of frame reception, the Receive Block indicates to the DMA Block whether the frame is good or bad. The DMA Block recovers the current receive buffer if the frame was bad. The Receive Block signals the register block to increment the alignment error, the CRC (FCS) error, the short frame, long frame, jabber error, the receive symbol error statistics and the length field mismatch statistics.

The Jumbo Frames enable bit (NETCONFIG:JUMBOFRM) instructs the EMAC to receive jumbo frames of up to 10,240 bytes in size.This operation does not form part of the IEEE 802.3 specification and is disabled by default. When jumbo frames are enabled, frames received with a frame size greater than 10,240 bytes are discarded.

Jumbo Frames of 2047 bytes or less have a four-byte FCS appended to the end of the frame; Jumbo Frames greater than 2047 bytes have a 64-byte FCS appended. The FCS value can be 'discarded' by programming NETCONFIG:DISCARDFCS to 1. This causes the appended bytes to be discarded instead of being copied to memory. However, the appended bytes are still transmitted with the frame and are filled with 0s. When operating with Jumbo Frames, be sure to remove the FCS fields from the frame data.

6.1.4 Transmit Block

The Transmit Block transmits frames in accordance with the Ethernet IEEE 802.3 CSMA/CD protocol. Frame assembly starts by adding a preamble and the start frame delimiter. Data is taken from the transmit FIFO one word at a time. Data is transmitted least-significant nibble first. If NETCONFIG:BITRATE is programmed to 1, the data is serialized and transmitted least-significant bit first instead.

If necessary, padding is added to make the frame length 60 bytes. A 32-bit CRC polynomial is inverted and appended to the end of the frame, making the frame length a minimum of 64 bytes. If the 'No CRC' bit is programmed to 1 in the second word of the final buffer descriptor of a transmit frame, neither pad nor CRC are appended.

In full-duplex mode frames are transmitted immediately. Back-to-back frames are transmitted at least 96 bit times apart to guarantee the inter-frame gap. In half-duplex mode, the transmitter checks the Carrier Sense (ETHERCRS) pin. If asserted, it waits for it to deassert, then starts transmission after the inter-frame gap of 96 bit times. If the Collision pin (ETHERCOL) is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retry transmission after the programmed Back-Off Time has elapsed.

The Back-Off Time is based on an Exclusive OR of the 10 least-significant bits of the data stream from the transmit FIFO and a 10-bit pseudo-random number generator. The number of bits used depends on the number of collisions seen. After the first collision one bit is used, the second two, and so on up to 10. For more than 10, all 10 bits are used. If 16 attempts cause collisions, an error is indicated and no further attempts will be made.

If transmit DMA underruns, bad CRC is automatically appended using the same mechanism as jam insertion and the Transmit Error (ETHERTXER) pin is asserted. For a properly configured system this should never happen.

If the NETCTL:BACKPRESS bit is set in half-duplex mode, the transmit block will transmit 64 bits of data, which can consist of 16 nibbles of 1011, or in bit-rate mode, 64 consecutive 1s, whenever it sees an incoming frame to force a collision. This provides a way of implementing flow control in half-duplex mode.

6.1.4.1 Pause Frame Support

The start of an 802.3 pause frame follows the format in Table 6-3.

Table 6-3. Pause Frame Support

DESTINATION ADDRESS	SOURCE ADDRESS	TYPE (MAC CONTROL FRAME)	PAUSE OPCODE	PAUSE TIME
0x0180C2000001	6 bytes	0x8808	0x0001	2 bytes

If a valid pause frame is received, the Pause Time register (PAUSETIME) is updated with the frame's pause time regardless of its current contents, and regardless of the state of the NETCONFIG:PAUSEEN bit. An interrupt is asserted when a pause frame is received, assuming it is enabled in the Interrupt Mask register. If NETCONFIG:PAUSEEN is 1 and the value of the PAUSETIME register is non-zero, no new frame is transmitted.

A valid pause frame has a destination address that matches either the address stored in Specific Address Register 1 (SPECAD1BOT and SPECAD1TOP) or matches 0x0180C2000001, has a MAC Control Frame Type ID of 0x8808, and has the Pause Opcode of 0x0001.

Pause frames that have FCS or other errors are treated as invalid and discarded. Valid pause frames received increment the Pause Frame Received statistics register (PAUSEFRRX).

The PAUSETIME register decrements every 512 bit times once transmission has stopped. For test purposes, the register decrements every Receive Clock cycle once transmission has stopped if NETCONFIG:RETRY is 1. If the NETCONFIG:PAUSEEN is not 1, decrementing occurs whether transmission has stopped or not. An interrupt is asserted whenever the PAUSETIME register decrements to zero (assuming the interrupt is enabled in the MASK register).

If either NETCTL:TXZEROQ or NETCTL:TXPAUSEFM is programmed to 1, a pause frame will be transmitted only if full duplex is selected in the NETCONFIG register, and transmit is enabled in the NETCTL register. Pause frame transmission occurs immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted. The transmitted pause frame comprises:

- A destination address of 0x0180C2000001
- The MAC Source Address taken from the SPECAD1TOP and SPECAD1BOT registers
- A Type ID of 0x8808 (MAC control frame)
- A Pause Opcode of 0x0001
- A Pause Quantum
- The remaining bytes filled with 0x00 to make the frame length equal to required minimum frame length (60 bytes)
- Valid FCS.

The Pause Quantum for the generated frame depends on the trigger source for the frame:

1. If NETCTL:TXPFRAME is programmed to 1, the pause quantum comes from the TXPAUSEQ register. The TXPAUSEQ register resets to a value of 0xFFFF, giving a maximum pause quantum as default.
2. If NETCTL:TXZEROQ is programmed to 1, the pause quantum will be zero.

After transmission, no interrupts are generated and the only statistics register incremented is the TXPAUSEFM register.

Pause frames can also be transmitted by the MAC using normal frame transmission methods. A pause frame can be sent while the transmitter is paused by resetting the NETCONFIG:PAUSEEN bit.

6.1.5 Address Checking Block

The Address Checking Block examines the destination addresses of received frames and indicates to the DMA Block which frames should be copied to memory. Whether a frame is copied depends on the Network Configuration register, the contents of the Specific Address and Hash registers, and the frame's destination address.

If the NETCONFIG:ENFRM bit is 0, a frame received while the EMAC is transmitting in half-duplex mode will not be copied to memory. If the NETCONFIG:ENFRM bit is 1, frames can be received while transmitting in half-duplex mode.

Special Ethernet frames are also detected by the Address Checking block. Ethernet frames are transmitted one byte at a time, least significant bit first. The first six bytes of an Ethernet frame contains the destination address. The first bit of that destination address is the group/individual bit: 1 for Multicast addresses and 0 for Unicast. The address 0xFFFFFFFFFFFF is the Broadcast address, a special case of Multicast.

The EMAC supports recognition of four specific addresses. Each specific address requires two registers. Specific Address Register Bottom (SPECADxBOT) stores the first four bytes of the address and Specific Address Register Top (SPECADxTOP) contains the last two bytes. The addresses stored can be specific, group, local or universal. SPECAD1 should be used for the MAC address.

The destination address of received frames is compared against the data stored in the specific address registers once they have been programmed. The addresses are deactivated at reset or when their corresponding specific address register bottom is written. They are activated when specific address register top is written, preventing a partial address from becoming active. If a receive frame address matches, the frame is copied to memory.

6.1.5.1 Broadcast Address

The Broadcast address of 0xFFFFFFFFFFFF is only recognized if the NETCONFIG:NOBCAST bit is 0. This causes the store frame and the Broadcast Match Status signal to be sent to the DMA Block.

6.1.5.2 Hash Addressing

The hash address register is 64-bits long and requires two locations in the memory map. The least significant bits are in HASHBOT and the most significant bits in HASHTOP.

The Unicast Hash Enable and the Multicast Hash Enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash registers. The hash function is an Exclusive OR of every sixth bit of the destination address:

HASH_INDEX[5] = XOR (DA[5]:DA[11]:DA[17]:DA[23]:DA[29]:DA[35]:DA[41]:DA[47])

HASH_INDEX[4] = XOR (DA[4]:DA[10]:DA[16]:DA[22]:DA[28]:DA[34]:DA[40]:DA[46])

HASH_INDEX[3] = XOR (DA[3]:DA[09]:DA[15]:DA[21]:DA[27]:DA[33]:DA[39]:DA[45])

HASH_INDEX[2] = XOR (DA[2]:DA[08]:DA[14]:DA[20]:DA[26]:DA[32]:DA[38]:DA[44])

HASH_INDEX[1] = XOR (DA[1]:DA[07]:DA[13]:DA[19]:DA[25]:DA[31]:DA[37]:DA[43])

HASH_INDEX[0] = XOR (DA[0]:DA[06]:DA[12]:DA[18]:DA[24]:DA[30]:DA[36]:DA[42])

DA[0] represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and DA[47] represents the most significant bit of the last byte received.

If the hash index points to a bit that is 1 in the Hash registers, the frame is matched by whether the frame is multicast or unicast. A multicast match results if the Multicast Hash Enable bit is 1. DA[0] is 1 and the hash index points to a bit set in the Hash registers.

A unicast match results if the Unicast Hash Enable bit is 1. DA[0] is 0 and the hash index points to a bit set in the Hash registers.

To receive all multicast frames, program the hash registers with all ones and program the NETCONFIG:MULTIHASHEN bit to 1.

6.1.5.3 Copy All Frames (Promiscuous Mode)

If the NETCONFIG:CPYFRM bit is 1, all non-error frames are copied to memory. For example, frames that are too long, too short, have FCS errors or have the ETHERRXER pin asserted during reception are discarded and all others are received.

6.1.5.4 Type ID Checking

The contents of the IDCHK register are compared against the Length/Type ID of received frames. Bit 22 in the Receive Buffer Descriptor List is 1 if there is a match (see Table 6-1). The reset state of IDCHK is zero, which is unlikely to match the Length/Type ID of any valid Ethernet frame.

6.1.5.5 VLAN Support

An Ethernet encoded 802.1Q VLAN tag contains the parameters in Table 6-4.

Table 6-4. VLAN Support

PARAMETER	DEFINITION
TPID (Tag Protocol Identifier) 16 bits	TCI (Tag Control Information) 16 bits
0x8100	First 3 bits priority, then CFI bit, last 12 bits VID

The VLAN tag is inserted at the 13th byte of the frame, adding four bytes to the frame length. If the VID (VLAN identifier) is null (0x000), this indicates a priority-tagged frame.

The MAC can support frame lengths up to 1,536 bytes, 18 bytes more than the original Ethernet maximum frame length of 1,518 bytes. This is enabled by programming the NETCONFIG:RECBYTE bit to 1.

Bits [21:16] of the Receive Buffer Descriptor List (see Table 6-1) provide information about VLAN tagged frames:

- Bit 21= 1 if the receive frame is VLAN tagged (Type ID = 0x8100)
- Bit 20 = 1 if receive frame is priority tagged (Type ID = 0x8100 and null VID); bit 21 must also = 1
- Bits 19, 18, and 17 = 1 if bit 21 is set
- Bit 16 = 1 if bit 21 is set

6.2 Programming Model

This section provides a programming model and example for the EMAC. Also, refer to Section 6.3, Register Reference, for specific register and bit definitions.

6.2.1 Initialization

Initialization of the EMAC configuration must be done while the transmit and receive circuits are disabled. See the descriptions of the Network Control register and Network Configuration register in Register Reference section.

6.2.1.1 Receive Buffer List

Receive data is written to buffers assigned to system memory. These buffers are described in the Receive Buffer Queue, which is a sequence of Receive Buffer Descriptor entries as defined in Table 6-1.

To create the Receive Buffer Descriptor list:

1. Allocate a number (n) of buffers in system memory of 128 bytes each.
2. Allocate an area 2 × (n) words for the Receive Buffer Descriptor List in system memory and create (m) entries in this list. Mark all entries in this list as owned by EMAC, by programming bit 0 of Word 0 to 0.
3. If fewer than 1,024 buffers are defined, the last descriptor must be marked with the Wrap bit (bit 1 in Word 0 programmed to 1), which will allow the buffer usage to wrap back to the first buffer.
4. Write the system memory address of Receive Buffer Descriptor List to the RXBQP register.
5. The receive circuits can then be enabled by writing to the HASHBOT, HASHTOP, SPECADxBOT, and SPECADxTOP address recognition registers, and then programming NETCTL:RXEN to 1 to enable the receive circuit.

6.2.1.1.1 Address Matching

The HASHBOT, HASHTOP, and the four SPECADxBOT, SPECADxTOP register-pairs must be programmed with the appropriate addresses (refer to Figure 6-2). Each register-pair comprises a bottom register and top register with the bottom register being written first. Address matching is disabled for a particular register pair after the bottom register has been programmed and re-enabled when the top register is programmed. See Section 6.1.5 for details of address matching. Each register pair may be written at any time, regardless of whether the receive circuits are enabled or disabled.

6.2.1.1.2 Receiving Frames

When a frame is received and the receive circuits are enabled, the EMAC checks the address and if it satisfies one of the listed cases, the frame is written to system memory:

- If it matches one of the four Specific Address registers.
- If it matches the Hash Address Function.
- If it is a Broadcast address (0xFFFFFFFFFFFF) and Broadcasts are allowed.
- If the EMAC is configured to promiscuous mode.

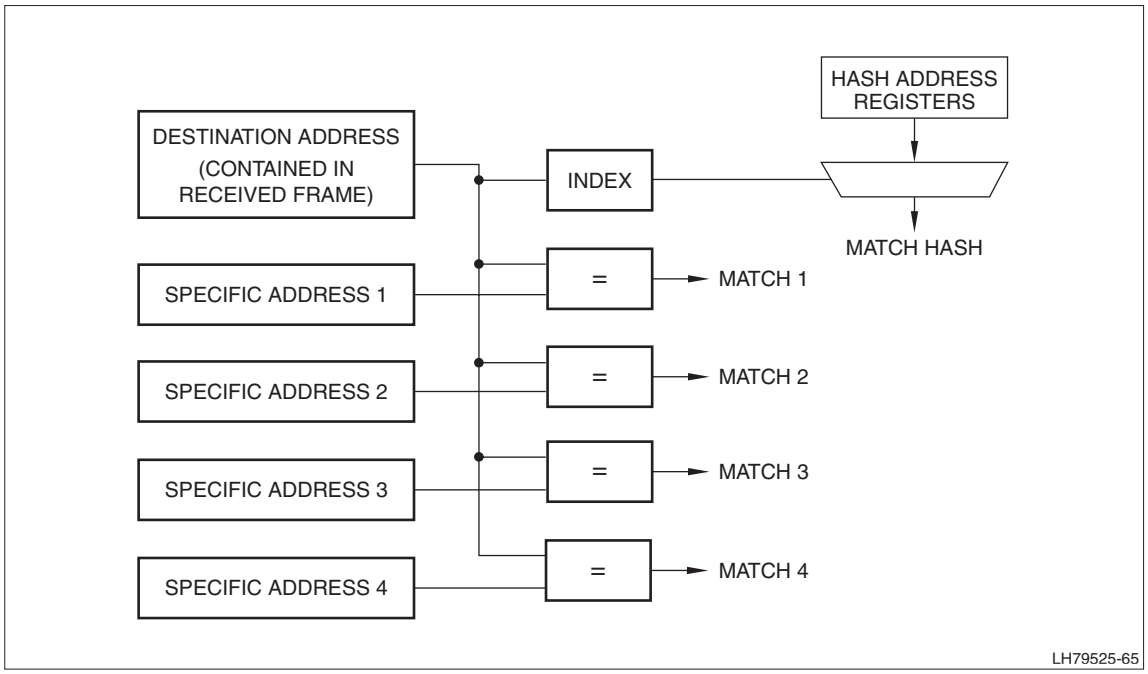


Figure 6-2. Address Matching

The RXBQP register points to the next entry in the Receive Buffer Descriptor List and the EMAC uses this as the address in system memory to which the frame is written. Once the frame has been successfully received and written to system memory, the EMAC then updates the Receive Buffer Descriptor entry (see Table 6-1) with the reason for the address match and marks the area as being owned by software. When complete, the Receive Complete interrupt is set. Software is then responsible for handling the data in the buffer and then releasing the buffer by writing the ownership bit back to 0, then clearing the interrupt.

If the EMAC is unable to write the data at a rate to match the incoming frame, a Receive Overrun interrupt is set. If there is no receive buffer available, i.e. the next buffer is still owned by software, the Receive Buffer Not Available interrupt is set. If the frame is not successfully received, a statistic register is incremented and the frame is discarded without informing software.

6.2.1.2 Transmit Buffer List

Transmit data is read from buffers assigned to system memory. These buffers are described in the Transmit Buffer Queue, which is a sequence of Transmit Buffer Descriptor entries as defined in Table 6-2.

To create this list of buffers:

1. Allocate a number (n) of buffers of between 1 and 2,047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 2 × (n) words for the Transmit Buffer Descriptor List in system memory and create (m) entries in this list. Mark all entries in this list as owned by EMAC, by programming bit 31 of word 1 to 0.
3. If fewer than 1,024 buffers are defined, the last descriptor must be marked with the Wrap bit; program bit 30 in word 1 to 1.
4. Write the address of Transmit Buffer Descriptor List to the TXBQP register.
5. The Transmit circuits can then be enabled by programming NETCTL:TXEN to 1.

6.2.1.3 Transmitting Frames

To set up a frame for transmission:

1. Program NETCTL:TXEN to 0.
2. Allocate an area of system memory for transmit data. This does not have to be contiguous; varying lengths can be used as long as they conclude on byte borders.
3. Set up the transmit buffer list.
4. Program NETCTL:TXEN to 1 and enable interrupts by programming the appropriate bits in the ENABLE register.
5. Write data for transmission into the buffers.
6. Write the address to TXBQP pointer.
7. Write control and length to Word one of the Transmit Buffer Descriptor entry.
8. Program NETCTL:STARTTX to 1 to start transmission.

6.2.1.4 Local Loop Back Mode

To change Loop Back mode (either to, or from Loop Back):

1. Program NETCTL:TXEN and NETCTL:RXEN to 0 to disable transmit and receive circuits.
2. Program NETCTL:LOOPLOCAL to its opposite value to change Loop Back mode.
3. Program NETCTL:TXEN and NETCTL:RXEN to 1 to enable transmit and receive circuits.

NOTE: These writes to the Network Control Register must be programmed in separate steps, and cannot be combined in any way.

6.2.1.5 PHY Maintenance

The PHYMAINT register enables the EMAC to communicate with a PHY using the MDIO interface. It is used during auto-negotiation to ensure that the EMAC and the PHY are configured for the same speed and either half- or full-duplex configuration.

The PHYMAINT register is implemented as a shift register. Writing to the register starts a shift operation that is signalled as complete when the NETSTATUS:PHYIDLE bit is set to 1 (about 2,000 HCLK cycles later, when the NETCONFIG:DIV field is set to 1). An interrupt is generated as this bit is set.

During this time, the MSB of the PHYMAINT register is output on the ETHERMDIO pin and the LSB updated from the ETHERMDIO with each ETHERMDC cycle. This causes the transmission of a PHY management frame on MDIO.

Reading during the shift operation returns the current contents of the shift register. At the end of the management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY.

ETHERMDC should not toggle faster than 2.5 MHz. ETHERMDC is generated by dividing down HCLK. The NETCONFIG:DIV bits set the divisor for HCLK to produce ETHERMDC. The default is 32, which is acceptable for HCLK running up to 80 MHz.

6.2.1.6 Interrupts

There are 14 interrupt conditions that are detected within the EMAC. These are ORed to make a single interrupt, which is presented to the Vectored Interrupt Controller (VIC), if interrupts are enabled. To ascertain which interrupt has been generated, read the Interrupt Status Register (INSTATUS). Note that this register clears when read.

At reset all interrupts are disabled. To enable an interrupt, write to Interrupt Enable Register (ENABLE) with the pertinent interrupt bit programmed to 1. To disable an interrupt, write to Interrupt Disable Register (DISABLE) with the pertinent interrupt bit programmed to 1. To determine whether an interrupt is enabled or disabled, read Interrupt Mask Register (MASK): if the bit is 1, the interrupt is disabled.

6.3 Register Reference

This section provides the EMAC register memory mapping and bit fields.

6.3.1 Memory Map

The base address for the EMAC is 0xFFFC7000.

Table 6-5 Summarizes the EMAC registers. There are three types of registers in the EMAC: control, configuration, and status registers; statistics registers; and matching registers. Address offsets in the table are from the base address. All registers are little endian format.

Table 6-5. EMAC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
CONTROL, CONFIGURATION, AND STATUS REGISTERS		
0x00	NETCTL	Network Control Register
0x04	NETCONFIG	Network Configuration Register
0x08	NETSTATUS	Network Status Register
0x0C	///	Reserved
0x10	///	Reserved
0x14	TXSTATUS	Transmit Status Register
0x18	RXBQP	Receive Buffer Queue Pointer
0x1C	TXBQP	Transmit Buffer Queue Pointer
0x20	RXSTATUS	Receive Status Register
0x24	INSTATUS	Interrupt Status Register
0x28	ENABLE	Interrupt Enable Register
0x2C	DISABLE	Interrupt Disable Register
0x30	MASK	Interrupt Mask Register
0x34	PHYMAINT	PHY Maintenance Register
0x38	PAUSETIME	Pause Time Register
0xBC	TXPAUSEQUAN	Transmit pause quantum
STATISTICS REGISTERS		
0x3C	PAUSEFRRX	Pause Frames Received
0x40	FRMTXOK	Frames Transmitted OK
0x44	SINGLECOL	Single Collision Frames
0x48	MULTFRM	Multiple Collision Frames
0x4C	FRMRXOK	Frames Received OK
0x50	FRCHK	Frame Check Sequence Errors
0x54	ALIGNERR	Alignment Errors
0x58	DEFTXFRM	Deferred Transmission Frames
0x5C	LATECOL	Late collisions
0x60	EXCOL	Excessive collisions
0x64	TXUNDER	Transmit underrun errors

Table 6-5. EMAC Register Summary (Cont'd)

ADDRESS OFFSET	NAME	DESCRIPTION
0x68	SENSERR	Carrier sense errors
0x6C	RXRERR	Receive resource errors
0x70	RXOVERR	Receive overrun errors
0x74	RXSYMERR	Receive symbol errors
0x78	LENERR	Excessive length errors
0x7C	RXJAB	Receive jabbers
0x80	UNDERFRM	Undersize frames
0x84	SQERR	SQE test errors
0x88	RXLEN	Received length field mismatch
0x8C	TXPAUSEFM	Transmitted pause frames
MATCHING REGISTERS		
0x90	HASHBOT	Hash register bottom [31:0]
0x94	HASHTOP	Hash register top [63:32]
0x98	SPECAD1BOT	Specific address 1 bottom
0x9C	SPECAD1TOP	Specific address 1 top
0xA0	SPECAD2BOT	Specific address 2 bottom
0xA4	SPECAD2TOP	Specific address 2 top
0xA8	SPECAD3BOT	Specific address 3 bottom
0xAC	SPECAD3TOP	Specific address 3 top
0xB0	SPECAD4BOT	Specific address 4 bottom
0xB4	SPECAD4TOP	Specific address 4 top
0xB8	IDCHK	Type ID checking
0xC0 – 0xFC	///	Reserved – Do not access

6.3.2 Control, Configuration, And Status Register Definitions

6.3.2.1 Network Control Register (NETCTL)

The NETCTL register allows configuration and testing of the MCU on the network.

Table 6-6. NETCTL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///			TXZEROQ	TXPFRAME	TXHALT	STARTTX	BACKPRESS	WRENSTAT	INCRSTAT	CLRSTAT	MANAGEEN	TXEN	RXEN	LOOPLOCAL	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	WO	WO	WO	WO	RW	RW	WO	WO	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x00															

Table 6-7. NETCTL Fields

BITS	NAME	FUNCTION
31:13	///	**Reserved** Reading returns 0. Write the reset value.
12	TXZEROQ	**Transmit Zero Quantum Pause Frame** Causes a pause frame with zero pause quantum to be transmitted at the next available transmitter idle time. 1 = Transmit zero quantum pause frame 0 = No action
11	TXPFRAME	**Transmit Pause Frame** Transmit a pause frame with the pause quantum from the Transmit Pause Quantum register at the next available transmitter idle time. 1 = Transmit pause frame 0 = No action
10	TXHALT	**Transmit Halt** Halt transmission as soon as any ongoing frame transmission ends. 1 = Halt transmission at end of current frame 0 = No action
9	STARTTX	**Start Transmission** 1 = Start transmission 0 = No action
8	BACKPRESS	**Back Pressure** In half-duplex mode, forces collisions on all received frames. 1 = Force collisions 0 = No action

Table 6-7. NETCTL Fields (Cont'd)

BITS	NAME	FUNCTION
7	WRENSTAT	**Write Enable for Statistics Registers** Makes the statistics registers writable for functional test purposes. 1 = Make statistics registers writable 0 = No action
6	INCRSTAT	**Increment Statistics Registers** Increment all the statistics registers by one for test purposes. 1 = Increment statistics registers 0 = No action
5	CLRSTAT	**Clear Statistics Registers** Clear the statistics registers. Individual statistics registers are cleared each time they are read. 1 = Clear statistics registers 0 = No action
4	MANGEEN	**Management Port Enable** Enables the management port. 1 = Management Port enabled 0 = Force ETHERMDIO to high impedance state and ETHERMDC LOW.
3	TXEN	**Transmit Enable** Enables the EMAC transmitter to send data. When disabled, transmission stops immediately, the transmit FIFO and control registers are cleared, and the TXBPQ register is reset to point to the start of the Transmit Descriptor list. 1 = Enable Ethernet transmitter 0 = Disable Ethernet transmitter
2	RXEN	**Receive Enable** Enables the EMAC to receive data. When disabled, frame reception stops immediately and the receive FIFO is cleared. The RXBPQ register is unaffected. 1 = Enable EMAC receiver 0 = Disable EMAC receiver
1	LOOPLOCAL	**Loop Back Local** Connects ETHERTX to ETHERRX, ETHERTXEN to ETHERRXDV, forces full duplex and drives ETHERRXCLK and ETHERTXCLK with HCLK divided by 4. ETHERRXCLX and ETHERTXCLK may glitch as the EMAC is switched in and out of internal Loop Back. It is important that receive and transmit circuits must already have been disabled before making the switch into and out of internal Loop Back. 1 = Enable Loop Back 0 = Disable Loop Back
0	///	**Reserved** Reading returns 0. Write the reset value.

6.3.2.2 Network Configuration Register (NETCONFIG)

This register allows general network configuration.

Table 6-8. NETCONFIG Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///												IGNORE	ENFRM	DISCARDFCS	LENGTHCHK
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RXBUFOS		PAUSEEN	RETRY	DIV		///	RECBYTE	UNIHASHEN	MULTIHASHEN	NOBCAST	CPYFRM	JUMBOFRM	BITRATE	FULLDUPLEX	///
RESET	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x04															

Table 6-9. NETCONFIG Fields

BITS	NAME	FUNCTION
31:20	///	**Reserved** Reading returns 0. Write the reset value.
19	IGNORE	**Ignore RX FCS** This bit causes frames with FCS/CRC errors to not be rejected and no FCS error statistics to be counted. For normal operation, this bit must be programmed to 0. 1 = Do not reject frames with FCS/CRC errors or count FCS errors 0 = Normal operation
18	ENFRM	**Enable Frames** Enable frames to be received in half-duplex mode while transmitting. 1 = Enable frame receipt in half-duplex mode while transmitting 0 = Disable frame receipt in half-duplex mode while transmitting
17	DISCARDFCS	**Discard Receive FCS** Specifies whether the FCS field of received frames will be copied to memory. 1 = Do not copy FCS field to memory 0 = Copy FCS field to memory
16	LENGTHCHK	**Receive Length Field Checking Enable** Specifies whether frames with measured lengths shorter than their length fields will be discarded. Frames containing a type ID in bytes 13 and 14 (length/type ID = 0600) will not be counted as length errors. 1 = Discard frames with length shorter than length field 0 = Do not discard frames with length shorter than length field
15:14	RXBUFOS	**Receive Buffer Offset** Indicates the number of bytes by which the received data is offset from the start of the first receive buffer.
13	PAUSEEN	**Pause Enable** Specifies if transmission will pause when a valid Pause frame is received. 1 = Pause transmission if a valid Pause frame is received 0 = Do not pause on receipt of a Pause frame

Table 6-9. NETCONFIG Fields (Cont'd)

BITS	NAME	FUNCTION
12	RETRY	**Retry Test** Must be programmed to 0 for normal operation. If programmed to 1, the delay between collisions will always be one slot time. Setting this bit to 1 helps testing the 'Too Many Entries' condition. Also used in the Pause Frame tests to reduce the Pause Counter's decrement time from 512 bit times, to every ETHERRXCLK cycle. 1 = Delay one slot time upon collision 0 = Normal operation
11:10	DIV	**Divisor** Program according to HCLK speed. This determines the HCLK divisor to generate ETHERMDC. For conformance with the IEEE 802.3 specification, MDC must not exceed 2.5 MHz (ETHERMDC is only active during MDIO read and write operations). 00 = Divide HCLK by 8 (for HCLK up to 20 MHz) 01 = Divide HCLK by 16 (for HCLK up to 40 MHz) 10 = Divide HCLK by 32 (for HCLK up to 80 MHz) 11 = Invalid
9	///	**Reserved** Reading returns 0. Write the reset value.
8	RECBYTE	**Receive 1,536 Byte Frames** Specifies if receive frames up to 1,536 bytes in length are allowed instead of rejecting any frame longer 1,518 bytes. 1 = Allow receive frames with up to 1,536 bytes 0 = Reject frames longer than 1,518 bytes
7	UNIHASHEN	**Unicast Hash Enable** Configures EMAC to receive unicast frames when the six-bit hash function of the destination address points to a bit that is set in the hash register. 1 = Receive unicast frames when hash function active 0 = Normal operation
6	MULTIHASHEN	**Multicast Hash Enable** Configures EMAC to receive multicast frames when the six-bit hash function of the destination address points to a bit that is set in the hash register. 1 = Receive multicast frames when hash function active 0 = Normal operation
5	NOBCAST	**No Broadcast** Allows configuration of the EMAC so that frames addressed to a Broadcast address of all 1s will not be received. 1 = Do not receive frames sent to Broadcast address of all 1s 0 = Normal operation
4	CPYFRM	**Copy All Frames** Allows all valid frames to be received. 1 = Allow reception of all valid frames 0 = Normal operation
3	JUMBOFRM	**Jumbo Frames** Enable acceptance of jumbo frames. **Note: When using jumbo frames, discard the last 4 bytes for transfers of 2047 bytes or less, and discard the last 64 bytes for frames larger than 2047 bytes.** 1 = Accept jumbo frames of up to 10,240 bytes 0 = Normal operation
2	BITRATE	**Bit Rate** Allows the interface to be configured for serial operation. Must be programmed before receive and transmit enable in the NETCTL register. If 1, a serial interface is configured with transmit data being driven on pin ETHERTX[0] and received on pin ETHERRX[0] serially. Also the ETHERCRS and ETHERRXDV pins are logically ORed together so either may be used as the data valid signal. 1 = Set EMAC to serial operation 0 = Normal operation
1	FULLDUPLEX	**Full Duplex** Configures the transmit block to ignore the state of collisions and carrier sense, and allows receive while transmitting. 1 = Transmitter ignores the state of collision and carrier sense and allows receive while transmitting 0 = Normal operation
0	///	**Reserved** Reading returns 0. Write the reset value.

6.3.2.3 Network Status Register (NETSTATUS)

The NETSTATUS register is a read-only register that reports status of the PHY and MDIO.

Table 6-10. NETSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													PHYIDLE	MDIOINSTAT	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	—	—
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC7000 + 0x08															

Table 6-11. NETSTATUS Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	PHYIDLE	**PHY Logic Idle** The PHY management logic is idle (i.e. has completed). 1 = PHY Management Logic is idle 0 = PHY Management Logic is active
1	MDIOINSTAT	**ETHERMDIO Input Status** Returns status of the ETHERMDIO pin. Use the PHY Maintenance register for reading managed frames rather than this bit. 1 = ETHERMDIO pin is HIGH 0 = ETHERMDIO pin is LOW
0	///	**Reserved** Reading returns undetermined values. Write 0.

6.3.2.4 Transmit Status Register (TXSTATUS)

This register provides transmit status details. Individual bits may be cleared by writing 1 to them. It is not possible to program a bit to 1 by writing to the register.

Table 6-12. TXSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									TXUNDER	TXCOMPLETE	BUFEX	TXGO	RETRYLIMIT	COLLISION	USEDBIT
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RO	RW	RW	RW
ADDR	0xFFFC7000 + 0x14															

Table 6-13. TXSTATUS Fields

BITS	NAME	FUNCTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	TXUNDER	**Transmit Underrun** This bit shows when transmit DMA was not able to read data from the buffer in system memory. The cause can be that the AHB or ASB bus was not granted in time, a 'Not OK' response was returned, a zero length buffer was read, or because a Used bit was read midway through frame transmission. If this happens, the transmitter forces a bad CRC and the Transmit Error (ETHERTXER) pin HIGH. Read: 1 = DMA Unable to read data from memory 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect
5	TXCOMPLETE	**Transmit Complete** Advises when a frame has been transmitted. Read: 1 = Frame transmission complete 0 = Frame transmission not complete Write: 1 = Reset bit to 0 0 = No effect

Table 6-13. TXSTATUS Fields (Cont'd)

BITS	NAME	FUNCTION
4	BUFEX	**Buffers Exhausted Mid-Frame** If the buffers run out of data during transmission of a frame, transmission stops. When this happens, FCS is bad and the Transmit Error (ETHERTXER) pin is asserted. Read: 1 = Buffer exhausted mid-frame 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect
3	TXGO	**Transmit Go** Indicates that transmit is active. Read: 1 = Transmit is active 0 = Transmit is not active Write: 1 = Reset bit to 0 0 = No effect
2	RETRYLIMIT	**Retry Limit Exceeded** Indicates that the transmission retry limit was exceeded. Read: 1 = Retry limit exceeded 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect
1	COLLISION	**Collision Occurred** Indicates that a collision occurred. Read: 1 = Collision occurred 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect
0	USEDBIT	**Used Bit Read** Indicates that a Transmit Buffer Descriptor is read with its Used bit set. Read: 1 = A transmit buffer description Used bit was set 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect

6.3.2.5 Receive Buffer Queue Pointer (RXBQP)

This register points to the entry in the receive buffer queue (descriptor list) currently being used. It is written with the start location of the receive buffer descriptor list. The lower order bits increment as buffers are used up and wrap to their original values after either 1,024 buffers, or when bit 1 of the entry is set. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used.

Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the 'used' bits.

Receive buffer writes also comprise bursts of two words, and as with transmit buffer reads, it is recommended that bit 2 is always written with zero to prevent a burst from crossing a 1KB boundary, in violation of Section 3.6 of the AMBA specification.

Table 6-14. RXBQP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	RXBQP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RXBQP														///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO
ADDR	0xFFFC7000 + 0x18															

Table 6-15. RXBQP Fields

BITS	NAME	FUNCTION
31:2	RXBQP	**Receive Buffer Queue Pointer** Written with the address of the start of the receive queue, reads as a pointer to the current buffer being used.
1:0	///	**Reserved** Reading returns 0. Write the reset value.

6.3.2.6 Transmit Buffer Queue Pointer (TXBQP)

This register points to the entry in the transmit buffer queue (descriptor list) currently being used. It is written with the start location of the transmit buffer descriptor list. The lower order bits increment as buffers are used up and wrap to their original values after either 1,024 buffers, or when the wrap bit of the entry is set. This register can only be written when transmit is inactive, that is, when TXSTATUS:TXGO is LOW.

As transmit buffer reads consist of a burst of two words, it is recommended that bit 2 is always written with zero. This is to prevent a burst from crossing a 1KB boundary, in violation of Section 3.6 of the AMBA specification.

Table 6-16. TXBQP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	TXBQP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TXBQP														///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	
ADDR	0xFFFC7000 + 0x1C															

Table 6-17. TXBQP Fields

BITS	NAME	FUNCTION
31:2	TXBQP	**Transmit Buffer Queue Pointer** Write the address of the start of the transmit queue. Reads as a pointer to the first buffer of the frame being transmitted or about to be transmitted.
1:0	///	**Reserved** Reading returns 0. Write the reset value.

6.3.2.7 Receive Status Register (RXSTATUS)

Read this register to obtain details of the status of a receive. Once read, individual bits may be cleared by writing 1 to them. It is not possible to set a bit to 1 by writing to the register.

Table 6-18. RXSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													RXCOVERRUN	FRMREC	BUFNOTAVAIL
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
ADDR	0xFFFC7000 + 0x20															

Table 6-19. RXSTATUS Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	RXCOVERRUN	**Receive Overrun** The DMA Block was unable to store the receive frame to memory. Either because the AHB bus was not granted in time or because a 'Not OK' response was returned. The buffer will be recovered if this happens. Read: 1 = DMA unable to store receive frame 0 = Normal operation Write: 1 = Reset bit to 0 0 = No effect
1	FRMREC	**Frame Received** Indicates that one or more frames have been received and placed in memory. Read: 1 = One or more frames have been received and placed in memory 0 = No received frames Write: 1 = Reset bit to 0 0 = No effect
0	BUFNOTAVAIL	**Buffer Not Available** An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA will reread the pointer each time a new frame starts until a valid pointer is found. This bit will be 1 at each attempt that fails even if it has not had a successful pointer read since it has been cleared. Read: 1 = Buffer not available 0 = Buffer available Write: 1 = Reset bit to 0 0 = No effect

6.3.2.8 Interrupt Status Register (INSTATUS)

The EMAC generates a single interrupt. This register indicates the source of this interrupt. For test purposes each bit can be set or reset by directly writing to the interrupt status register regardless of the state of the mask register. Otherwise the corresponding bit in the MASK register must be cleared for a bit to be set. All bits are reset to zero on read. If any bit is set in this register, the Ethernet Interrupt signal will be asserted.

Table 6-20. INSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0															
TYPE	RO															
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PAUSEZERO	PAUSEFRRX	NOTOK	RECOVERRUN	///		TXCOMPLETE	TXBUFEXH	RETRYLIM	ETHTXBUFUR	TXUSDBITRD	RXUSDBITRD	RXCOMP	MNGFRMSENT
RESET	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO		RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x24															

Table 6-21. INSTATUS Fields

BITS	NAME	FUNCTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	PAUSEZERO	**Pause Time Zero** Indicates the PAUSETIME register has decremented to zero. This bit is reset to 0 when read. 1 = PAUSETIME decremented to zero 0 = PAUSETIME not decremented to zero
12	PAUSEFRRX	**Pause Frame Received** Indicates a valid Pause Frame has been received. This bit is reset to 0 when read. 1 = Valid pause frame received 0 = No valid pause frame received
11	NOTOK	**Response Not OK** Indicates that the DMA Block receives response 'Not OK'. This bit is reset to 0 when read. 1 = Response 'Not OK' 0 = Normal operation
10	RECOVERRUN	**Receive Overrun** The DMA Block was unable to store the receive frame to memory. 1 = Receive overrun 0 = No overrun
9	///	**Reserved** Reading returns 0. Write the reset value.
8	///	**Reserved** Reading returns 0. Write the reset value.

Table 6-21. INSTATUS Fields (Cont'd)

BITS	NAME	FUNCTION
7	TXCOMPLETE	**Transmit Complete** Denotes a frame has been successfully transmitted. This bit is reset to 0 when read. 1 = Transmit complete 0 = No complete transmit
6	TXBUFEXH	**Transmit Buffers Exhausted In Mid-frame** The transmit buffers have run out of data before the transmission of the frame completed. This bit is reset to 0 when read. 1 = Transmit buffers exhausted mid-frame 0 = No error
5	RETRYLIM	**Retry Limit Exceeded** The programmed retry limit was exceeded without a successful transmission. This bit is reset to 0 when read. 1 = Retry limit exceeded 0 = No error
4	ETHTXBUFUR	**Ethernet Transmit Buffer Underrun** The transmit DMA did not fetch frame data in time for it to be transmitted or response returned 'Not OK'. Also 1 if a Used bit is read mid-frame or when a new Transmit Queue Pointer is written. This bit is reset to 0 when read. 1 = Transmit buffer underrun 0 = No error
3	TXUSDBITRD	**Transmit Used Bit Read** Indicates that a transmit buffer descriptor has been read with its Used bit set. This bit is reset to 0 when read. 1 = 'Used' bit set 0 = No error
2	RXUSDBITRD	**Receive Used Bit Read** Set when a receive buffer descriptor is read with its Used bit set. This bit is reset to 0 when read. 1 = Used bit set 0 = No error
1	RXCOMP	**Receive Complete** A frame has been stored in memory. This bit is reset to 0 when read. 1 = Receive complete 0 = No receive complete
0	MNGFRMSENT	**Management Frame Sent** The PHY maintenance register has completed its operation. This bit is reset to 0 when read. 1 = Management frame sent 0 = No management frame sent

6.3.2.9 Interrupt Enable Register (ENABLE)

At reset all interrupts are disabled. Writing a 1 to the relevant bit location enables the required interrupt. This register is write only.

Table 6-22. ENABLE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PAUSETMZEROIEN	PAUSEFRRXIEN	NOTOKIEN	RECOVERRUNIEN	///		TXCOMPIEN	TXBUFEXHIEN	RETRYLMTEXIEN	TXBUFUNDERIEN	TXUSEDBITIEN	RXUSEDBITIEN	RXCOMPIEN	MNGDONEIEN
RESET	0	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
TYPE	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFC7000 + 0x28															

Table 6-23. ENABLE Fields

BITS	NAME	FUNCTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	PAUSETMZEROIEN	**Pause Time Zero Interrupt Enable**
12	PAUSEFRRXIEN	**Pause Frame Received Interrupt Enable**
11	NOTOKIEN	**Response Not OK Interrupt Enable**
10	RECOVERRUNIEN	**Receive Overrun Interrupt Enable**
9:8	///	**Reserved** Reading returns 0. Write the reset value.
7	TXCOMPIEN	**Transmit Complete Interrupt Enable**
6	TXBUFEXHIEN	**Transmit Buffers Exhausted In Mid-frame Interrupt Enable**
5	RETRYLMTEXIEN	**Retry Limit Exceeded Interrupt Enable**
4	TXBUFUNDERIEN	**Transmit Buffer Underrun Interrupt Enable**
3	TXUSEDBITIEN	**Transmit Used Bit Read Interrupt Enable**
2	RXUSEDBITIEN	**Receive Used Bit Read Interrupt Enable**
1	RXCOMPIEN	**Receive Complete Interrupt Enable**
0	MNGDONEIEN	**Management Done Interrupt Enable**

6.3.2.10 Interrupt Disable Register (DISABLE)

This register is used to disable individual interrupts. All interrupts are disabled following a reset. Writing a 1 to the relevant bit location disables that particular interrupt. This register is write only.

Table 6-24. DISABLE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PAUSETMZEROIDIS	PAUSEFRRXIDIS	NOTOKIDIS	RECOVERRUNIDIS	///		TXCOMPIDIS	TXBUFEXHIDIS	RETRYLMTEXIDIS	TXBUFUNDERIDIS	TXUSEDBITIDIS	RXUSEDBITIDIS	RXCOMPIDIS	MNGDONEIDIS
RESET	0	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
TYPE	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFC7000 + 0x2C															

Table 6-25. DISABLE Fields

BITS	NAME	FUNCTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	PAUSETMZEROIDIS	**Disable Pause Time Zero Interrupt**
12	PAUSEFRRXIDIS	**Disable Pause Frame Received Interrupt**
11	NOTOKIDIS	**Disable Response 'Not OK' Interrupt**
10	RECOVERRUNIDIS	**Disable Receive Overrun Interrupt**
9:8	///	**Reserved** Reading returns 0. Write the reset value.
7	TXCOMPIDIS	**Disable Transmit Complete Interrupt**
6	TXBUFEXHIDIS	**Disable Transmit Buffers Exhausted In Mid-Frame Interrupt**
5	RETRYLMTEXIDIS	**Disable Retry Limit Exceeded Interrupt**
4	TXBUFUNDERIDIS	**Disable Transmit Buffer Underrun Interrupt**
3	TXUSEDBITIDIS	**Disable Transmit Used Bit Read Interrupt**
2	RXUSEDBITIDIS	**Disable Receive Used Bit Read Interrupt**
1	RXCOMPIDIS	**Disable Receive Complete Interrupt**
0	MNGDONEIDIS	**Disable Management Done Interrupt**

6.3.2.11 Interrupt Mask Register (MASK)

The MASK register is a read-only register that shows the status of the interrupt based on what has been written to the ENABLE and DISABLE registers. As all interrupts are disabled following reset, the interrupt bits in this register are reset to 1.

Table 6-26. MASK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PAUSETMZEROMSK	PAUSEFRRXMSK	NOTOKMSK	RECOVERRUNMSK	///		TXCOMPMSK	TXBUFEXHMSK	RETRYLMTEXMSK	TXBUFUNDERMSK	TXUSEDBITMSK	RXUSEDBITMSK	RXCOMPMSK	MNGDONEMSK
RESET	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC7000 + 0x30															

Table 6-27. MASK Fields

BITS	NAME	FUNCTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	PAUSETMZEROMSK	1 = Pause Time Zero Interrupt masked 0 = Unmasked
12	PAUSEFRRXMSK	1 = Pause Frame Received Interrupt masked 0 = Unmasked
11	NOTOKMSK	1 = Response 'Not OK' Interrupt masked 0 = Unmasked
10	RECOVERRUNMSK	1 = Receive Overrun Interrupt masked 0 = Unmasked
9:8	///	**Reserved** Reading returns 0. Write the reset value.
7	TXCOMPMSK	1 = Transmit Complete Interrupt masked 0 = Unmasked
6	TXBUFEXHMSK	1 = Transmit Buffers Exhausted In Mid-frame Interrupt masked 0 = Unmasked
5	RETRYLMTEXMSK	1 = Retry Limit Exceeded interrupt masked 0 = Unmasked
4	TXBUFUNDERMSK	1 = Transmit Buffer Underrun interrupt masked 0 = Unmasked
3	TXUSEDBITMSK	1 = Transmit Used Bit Read interrupt masked 0 = Unmasked
2	RXUSEDBITMSK	1 = Receive Used Bit Read interrupt masked 0 = Unmasked
1	RXCOMPMSK	1 = Receive Complete interrupt masked 0 = Unmasked
0	MNGDONEMSK	1 = Management Done interrupt masked 0 = Unmasked

6.3.2.12 PHY Maintenance Register (PHYMAINT)

This register enables the EMAC to communicate with a PHY by means of the MDIO interface. It is used during auto negotiation to ensure that the EMAC and the PHY are configured for the same speed and duplex configuration.

The PHY maintenance register is implemented as a shift register. Writing to the register starts a shift operation, which is signalled as complete when bit two is set in the NETSTATUS register, about 2,000 HCLK cycles later. An interrupt is also generated.

During this time, the MSB of the register is output on the ETHERMDIO pin and the LSB updated from the ETHERMDIO pin with each ETHERMDC cycle. This causes transmission of a PHY management frame on ETHERMDIO. See Section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation will return the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits will be updated with data read from the PHY.

Table 6-28. PHYMAINT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	STARTFRM		OPERATION		PHYADDR					REGADDR					///	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PHYRDWR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x34															

Table 6-29. PHYMAINT Fields

BITS	NAME	FUNCTION
31:30	STARTFRM	**Start Of Frame** Must be written 0b01 for a valid frame.
29:28	OPERATION	**Operation** 00 = Invalid 01 = Write 10 = Read 11 = Invalid
27:23	PHYADDR	**Phy Address** The PHY address is written to this field.
22:18	REGADDR	**Register Address** Specifies the address of the register in the PHY to access.
17:16	///	Must be written with 0b10. Reads the value written.
15:0	PHYRDWR	**PHY Read or Write Data** The data to be written to the PHY is contained in these two bytes. For a read, this contains the two bytes read from the PHY.

6.3.2.13 Pause Time Register (PAUSETIME)

The PAUSETIME register contains the current value of PAUSETIME, which is decremented once every 512 bit-times (one slot time).

Table 6-30. PAUSETIME Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PAUSETIME															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC7000 + 0x38															

Table 6-31. PAUSETIME Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	PAUSETIME	**Pause Time** Stores the current PAUSETIME value, which is decremented every 512 bit-times.

6.3.2.14 Transmit Pause Quantum (TXPAUSEQUAN)

The value in this register is used in hardware generation of transmitted pause frames as value for pause quantum.

Table 6-32. TXPAUSEQUAN Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TXPAUSEQUAN															
RESET	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xBC															

Table 6-33. TXPAUSEQUAN Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	TXPAUSEQUAN	**Transmit Pause Quantum** Value for pause quantum.

6.3.3 Statistics Register Definitions

Statistics registers accumulate statistics for software drivers, RMON applications, and other uses. Reading statistics registers resets the counts to zero. If they are not read before reaching maximum count, the count does not wrap, but sticks at all 1s. The receive statistics registers only increment when the NETCTL:RXEN is programmed to 1.

For testing purposes, the statistics registers can be written to. Before writing to the statistics registers, NETCTL:WRENSTAT must first be programmed to 1.

6.3.3.1 Pause Frames Received (PAUSEFRRX)

This register allows software to read the number of good pause frames received. A good frame has a length of 64 to 1,518 (1,522 if NETCONFIG:RECBYTE is 1) and has no FCS, alignment or receive symbol errors.

Table 6-34. PAUSEFRRX Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PAUSEFRRXOk															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x3C															

Table 6-35. PAUSEFRRX Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	PAUSEFRRXOK	**Pause Frames Received OK** A 16-bit register counting the number of good pause frames received.

6.3.3.2 Frames Transmitted OK (FRMTXOK)

This is a 24-bit register counting the number of frames successfully transmitted (no underrun and no excessive retry errors).

Table 6-36. FRMTXOK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///								FRMTXOK							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	FRMTXOK															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x40															

Table 6-37. FRMTXOK Fields

BITS	NAME	FUNCTION
31:24	///	**Reserved** Reading returns 0. Write the reset value.
23:0	FRMTXOK	**Frames Transmitted OK** Shows the number of successfully-transmitted frames.

6.3.3.3 Single Collision Frames (SINGLECOL)

This is a 16-bit register counting the number of frames experiencing a single collision before being successfully transmitted (no underrun errors).

Table 6-38. SINGLECOL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SINGLECOL															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x44															

Table 6-39. SINGLECOL Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SINGLECOL	**Single Collision Frames** Contains the number of single-collision frames.

6.3.3.4 Multiple Collision Frames (MULTFRM)

This register counts the number of frames that experienced between two and 15 collisions before successful transmission.

Table 6-40. MULTFRM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	MULTFRM															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x48															

Table 6-41. MULTFRM Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	MULTFRM	**Multiple Collision Frames** Contains the number of successfully-transmitted frames that experienced between two and 15 collisions.

6.3.3.5 Frames Received OK (FRMRXOK)

This is a 24-bit register containing the number of good frames received. Good frames are defined as having the address recognized and successfully copied to memory. A good frame length is between 64 and 1,518 bytes (1,522 if NETCONFIG:RECBYTE is 1) and has no FCS, alignment, or receive symbol errors.

Table 6-42. FRMRXOK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///								FRMRXOK							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	FRMRXOK															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x4C															

Table 6-43. FRMRXOK Fields

BITS	NAME	FUNCTION
31:24	///	**Reserved** Reading returns 0. Write the reset value.
23:0	FRMRXOK	**Frames Received OK** Number of correctly received frames.

6.3.3.6 Frame Check Sequence Errors (FRCHK)

This register hold the count of frames an integral number of bytes-long, have a bad CRC, and are between 64 and 1,518 bytes in length (1,522 if NETCONFIG:RECBYTE is 1).

Table 6-44. FRCHK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								FRCHK							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x50															

Table 6-45. FRCHK Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	FRCHK	**Frame Check Sequence Errors** Number of Frame Check Sequence errors.

6.3.3.7 Alignment Errors (ALIGNERR)

This register holds the count of frames that are not an integral number of bytes-long and have bad CRC when their length is truncated to the correct number of bytes. The bytes' length is between 64 and 1,518 bytes in length (1,522 if NETCONFIG:RECBYTE is 1).

Table 6-46. ALIGNERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								ALIGNERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x54															

Table 6-47. ALIGNERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	ALIGNERR	**Alignment Errors** Number of alignment errors.

6.3.3.8 Deferred Transmission Frames (DEFTXFRM)

This is a 16-bit register containing the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit underrun.

Table 6-48. DEFTXFRM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	DEFTXFRM															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x58															

Table 6-49. DEFTXFRM Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	DEFTXFRM	**Deferred Transmission Frames** Deferred transmission frames count.

6.3.3.9 Late Collisions (LATECOL)

This is an 8-bit register containing the number of frames that experience a collision after the slot time (512 bit times) has expired. A late collision is counted twice (both as a collision and a late collision).

Table 6-50. LATECOL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								LATECOL							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x5C															

Table 6-51. LATECOL Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	LATECOL	**Late Collisions** Number of late collisions.

6.3.3.10 Excessive Collisions (EXCOL)

This register compiles the number of frames not transmitted due to more than 16 collisions occuring during transmission attempts.

Table 6-52. EXCOL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								EXCCOL							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x60															

Table 6-53. EXCCOL Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	EXCCOL	**Excessive Collisions** Number of frames that were not transmitted because they experienced 16 collisions.

6.3.3.11 Transmit Underrun Errors (TXUNDER)

This register compiles the number of frames not transmitted due to a transmit DMA underrun.

Table 6-54. TXUNDER Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								TXUNDER							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x64															

Table 6-55. TXUNDER Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	TXUNDER	**Transmit Underruns** Number of frames not transmitted due to a transmit DMA underrun. If this register is incremented, no other statistics register is incremented.

6.3.3.12 Carrier Sense Errors (SENSERR)

This register counts the number of Carrier Sense errors.

Table 6-56. SENSERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SENSERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x68															

Table 6-57. SENSERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	SENSERR	**Carrier Sense Errors** Number of frames transmitted where carrier sense was not detected during transmission, or where carrier sense was deasserted after being asserted in a transmit frame without a collision (no underrun). This count is only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behavior of the other statistics registers is unaffected by the detection of a carrier sense error.

6.3.3.13 Receive Resource Errors (RXRERR)

This register counting the number of frames that were address matched but could not be copied to memory because no receive buffer was available.

Table 6-58. RXRERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RXRERR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x6C															

Table 6-59. RXRERR Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	RXRERR	**Receive Resource Errors** The number of frames that were address matched but could not be copied to memory because no receive buffer was available.

6.3.3.14 Receive Overrun Errors (RXOVERR)

Recieve overrun errors are tabulated in this register.

Table 6-60. RXOVERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								RXOVERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x70															

Table 6-61. RXOVERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	RXOVERR	**Receive Overruns** Number of frames that are address recognized but were not copied to memory due to a receive DMA overrun.

6.3.3.15 Receive Symbol Errors (RXSYMERR)

This register counts the number of frames that had ETHERRXER asserted during reception.

Table 6-62. RXSYMERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								RXSYMERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x74															

Table 6-63. RXSYMERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	RXSYMERR	**Receive Symbol Errors** Counts the number of frames that had ETHERRXER asserted during reception.

6.3.3.16 Excessive Length Error Register (LENERR)

This is an 8-bit register containing the number of frames received exceeding 1,518 bytes (1,522 if NETCONFIG:RECBYTE is 1) in length but do not have either a CRC error, an alignment error, nor a receive symbol error.

Table 6-64. LENERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								LENERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x78															

Table 6-65. LENERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	LENERR	**Excessive Length Frames** Number of excessive length frames.

6.3.3.17 Receive Jabbers (RXJAB)

This is an 8-bit register containing the number of frames received exceeding 1,518 bytes (1,522 if NETCONFIG:RECBYTE is 1) in length and have either a CRC error, an alignment error, or a receive symbol error.

Table 6-66. RXJAB Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								RXJAB							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x7C															

Table 6-67. RXJAB Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	RXJAB	**Receive Jabbers** Number of receive jabbers.

6.3.3.18 Undersize Frames (UNDERFRM)

This is an 8-bit register containing the number of frames received that were fewer than 64 bytes in length, but do not have either a CRC error, an alignment error, or a receive symbol error.

Table 6-68. UNDERFRM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								UNDERFRM							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x80															

Table 6-69. UNDERFRM Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	UNDERFRM	**Undersize Frames** Number of undersize frames.

6.3.3.19 SQE Test Errors (SQERR)

This 8-bit register contains the number of frames where the ETHERCOL pin was not asserted within 96 bit times (one inter-frame gap) of the ETHERTXEN pin being deasserted in half-duplex mode.

Table 6-70. SQERR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SQERR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x84															

Table 6-71. SQERR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	SQERR	**SQE Test Errors** Shows the number of frames where the ETHERCOL pin was not asserted within 96 bit times

6.3.3.20 Received Length Field Mismatch (RXLEN)

This register counts the number of received frames that have a measured length shorter than specified in its length field. Checking is enabled via the NETCONFIG:LENGTHCHK bit. Frames containing a type ID in bytes 13 and 14 (length/type ID ≥ 0x0600) will not be counted as length field errors, nor will excessive length frames.

Table 6-72. RXLEN Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								RXLEN							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x88															

Table 6-73. RXLEN Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	RXLEN	**Receive Length Field Mismatch** Contains the number of received frames that have a measured length shorter than that extracted from its length field.

6.3.3.21 Transmitted Pause Frames (TXPAUSEFM)

This register contains the number of Pause Frames transmitted.

Table 6-74. TXPAUSEFM Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TXPAUSEFM															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x8C															

Table 6-75. TXPAUSEFM Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	TXPAUSEFM	**Transmitted Pause Frames** Number of pause frames transmitted.

6.3.4 Matching Registers

The matching registers allow programming specific addresses or Type IDs for matching with incoming frames.

6.3.4.1 Hash Register Bottom (HASHBOT)

This register contains the low-order bits of the Hash Address Register (bits [31:0]). For more information, see Section 6.1.5.2 for more information on hash addressing.

Table 6-76. HASHBOT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	HASHBOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	HASHBOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x90															

Table 6-77. HASHBOT Fields

BITS	NAME	FUNCTION
31:0	HASHBOT	**Hash Register Bottom** Bits [31:0] of the Hash Address Register.

6.3.4.2 Hash Register Top (HASHTOP)

This register contains the high-order bits of the Hash Address Register (bits [63:32]). For more information, see Section 6.1.5.2 for more information on hash addressing.

Table 6-78. HASHTOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	HASHTOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	HASHTOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x94															

Table 6-79. HASHTOP Fields

BITS	NAME	FUNCTION
31:0	HASHTOP	**Hash Register Top** Bits [63:32] of the Hash Address Register.

6.3.4.3 Specific Address 1 Bottom (SPECAD1BOT)

This register contains the least-significant bits (bits [31:0]) of the destination address. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received. SPECAD1 should be programmed with the system MAC address.

Table 6-80. SPECAD1BOT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SPECAD1BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD1BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x98															

Table 6-81. SPECAD1BOT Fields

BITS	NAME	FUNCTION
31:0	SPECAD1BOT	**Least Significant Destination Address Bits** Least significant bits of the destination address.

6.3.4.4 Specific Address 1 Top (SPECAD1TOP)

This register contains the most-significant bits of the destination address (bits [47:32]).

Table 6-82. SPECAD1TOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD1TOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0x9C															

Table 6-83. SPECAD1TOP Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SPECAD1TOP	**Most Significant Destination Address Bits** The most significant bits of the destination address, that is bits 47 to 32.

6.3.4.5 Specific Address 2 Bottom (SPECAD2BOT)

This register contains the least-significant bits of the destination address (bits [31:0]). Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

Table 6-84. SPECAD2BOT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SPECAD2BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD2BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xA0															

Table 6-85. SPECAD2BOT Fields

BITS	NAME	FUNCTION
31:0	SPECAD2BOT	**Least Significant Destination Address Bits** Least significant bits of the destination address.

6.3.4.6 Specific Address 2 Top (SPECAD2TOP)

This register contains the most-significant bits of the destination address (bits [47:32]).

Table 6-86. SPECAD2TOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD2TOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xA4															

Table 6-87. SPECAD2TOP Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SPECAD2TOP	**Most Significant Destination Address Bits** The most significant bits of the destination address.

6.3.4.7 Specific Address 3 Bottom (SPECAD3BOT)

This register contains the least-significant bits of the destination address (bits [31:0]). Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

Table 6-88. SPECAD3BOT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SPECAD3BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD3BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xA8															

Table 6-89. SPECAD3BOT Fields

BITS	NAME	FUNCTION
31:0	SPECAD3BOT	**Least Significant Destination Address Bits** Least significant bits of the destination address.

6.3.4.8 Specific Address 3 Top (SPECAD3TOP)

This register contains the most-significant bits of the destination address (bits [47:32]).

Table 6-90. SPECAD3TOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD3TOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xAC															

Table 6-91. SPECAD3TOP Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SPECAD3TOP	**Most Significant Destination Address Bits** The most significant bits of the destination address.

6.3.4.9 Specific Address 4 Bottom (SPECAD4BOT)

This register contains the least-significant bits of the destination address (bits [31:0]). Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

Table 6-92. SPECAD4BOT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SPECAD4BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD4BOT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xB0															

Table 6-93. SPECAD4BOT Fields

BITS	NAME	FUNCTION
31:0	SPECAD4BOT	**Least Significant Destination Address Bits** Least significant bits of the destination address.

6.3.4.10 Specific Address 4 Top (SPECAD4TOP)

This register contains the most-significant bits of the destination address (bits [47:32]).

Table 6-94. SPECAD4TOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SPECAD4TOP															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xB4															

Table 6-95. SPECAD4TOP Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SPECAD4TOP	**Most Significant Destination Address Bits** The most significant bits of the destination address, that is bits 47 to 32.

6.3.4.11 Type ID Checking (IDCHK)

This register contains the TypeID/Length to compare to received frames.

Table 6-96. IDCHK Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	IDCHK															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC7000 + 0xB8															

Table 6-97. TypeIDCheck Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	IDCHK	**Type ID Checking** TypeID/Length field.

Chapter 7
External Memory Controller

The External Memory Controller (EMC) provides all the interface and control functions for the both static and dynamic memory devices. The EMC features:

- Dynamic memory interface support including SDRAM and JEDEC low-power SDRAM
- Asynchronous static memory device support including RAM, ROM, and Flash, with or without asynchronous page mode
- Low transaction latency
- Read and Write buffers to reduce latency and to improve performance
- 8-bit and 16-bit wide static memory support (32-bit support in the LH79524)
- 16-bit wide synchronous SDRAM memory support (32-bit support in the LH79524)
- Static memory features include:
  - Asynchronous page mode read
  - Programmable wait states
  - Bus turnaround delay
  - Output enable, and write enable delays
  - Extended wait
  - Wait states may be extended indefinitely using the nWAIT pin
- Two Chip Selects for synchronous and up to four Chip Selects for static memory devices
- Up to four memory Byte Lane Selects for static memory devices
- Boot in 8-, 16-, or 32-bit mode.

7.1 Theory of Operation

The combined dynamic and static memory controller controls all static and dynamic memory accesses. Figure 7-1 is a block diagram of the EMC. The EMC contains the following:

- Command sequencer — Rearranges the memory accesses for maximum efficiency.
- Memory Transfer State Machine — Controls the current transfer.
- Static Memory Controller Register Banks — The four static memory controller register banks hold the static memory registers. These registers allow software to control and configure the static memory controller.
- Dynamic Memory Controller Register Banks — The two dynamic memory controller register banks hold the dynamic memory registers. These registers allow software to control and configure the dynamic memory controller.

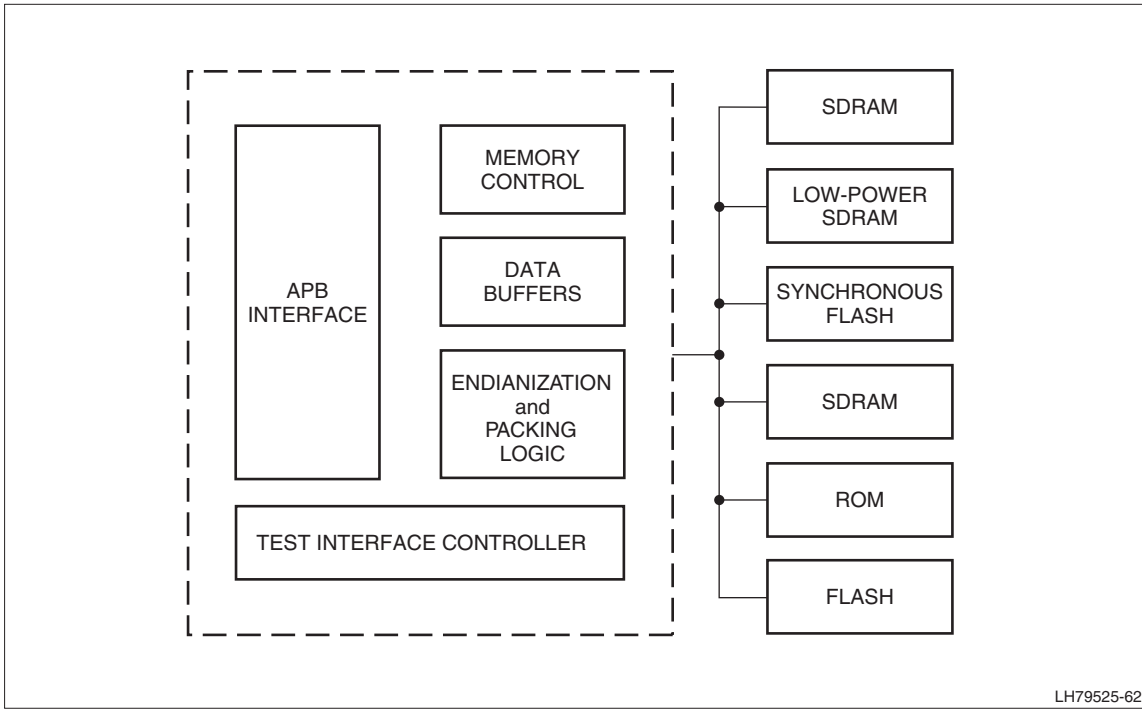


Figure 7-1. External Memory Controller Block Diagram

The EMC supports six banks of external memory. Together these banks occupy 384MB of address space. For an SRAM memory cycle, bits 31 through 26 are decoded to select the proper memory clock phase and the proper Chip Select. Bits 25 through 2 are passed as the external address to the memory bank for 32-bit external memory widths, bits 24 through 1 are passed for 16-bit external memory widths and bits 23 through 0 are passed for 8-bit external memory widths. Regardless of the memory width, the least significant address bit appears on pin 'A0'. For example, when interfacing with 32-bit memory, pin A0 carries the A2 address signal and pin A23 carries the A25 address signal. This is a bit different than many memory interfaces, but offers advantages such as a greater addressing space for 16- and 32- bit memory systems.

During an SDRAM memory cycle, the bank being accessed is selected by the assertion of the appropriate Chip Select signal. This is found by decoding address bit 28. A value of 0 causes Chip Select 0 to be asserted. Address bits 27 through 0 are passed as the external address to the memory bank.

Timing diagrams for both static and dynamic memory transactions are found in the Data Sheet.

7.1.1 External Memory Map

The EMC provides hardware support for booting from external nonvolatile memory. During boot the nonvolatile memory *must* be located at address 0x00000000 in memory. When the system is booted, nCS0, nCS1, or nDCS0 can be remapped to address 0x00000000 by software. The boot device is selected at reset by the state of PC[7:4]. A boot example later in this chapter shows these values in Table 7-2 and Table 7-3.

7.1.1.1 nCS1 Memory Configuration

The memory width and the byte lane state of static memory Chip Select 1 (nCS1) can be configured during a power on reset using the CS10V register, described in Section 3.2.3.

7.2 Static Memory

The static memory interface is externally asynchronous. However, the LH79524/LH79525 generates the external asynchronous signals using the internal system clock to synchronously control the switching. Thus, the timing of static memory signals is easily referenced to the internal system clock frequency.

The diagrams in this chapter as well as the Data Sheet show HCLK waveforms. However, that is simply for reference; HCLK is not transmitted on the external memory interface.

7.2.1 Static Memory Operation

The EMC increases addressing efficiency by automatically shifting the addresses output on the external memory bus, depending on the size of memory devices being addressed. For 8-bit memory systems, address *signal* A0 is connected to address *pin* A0. With a 16-bit system, addressing does not require address signal A0 since accesses begin on half-word boundaries. For 32-bit systems, both address signal A0 and A1 are not needed because memory accesses begin on word boundaries. Byte Lane Enables should be used for addressing individual bytes.

Figure 7-2 shows how the internal address *signals* are switched to the address *pins*. For example, in the configuration shown in Figure 7-2, the memory device width has been set to 32 bits by programming the SCONFIGx:MW field to 0b10. Note that this is the width of the connected memory system, not the device. Thus, if two 16-bit memory devices are connected as a 32-bit wide system, the MW field is programmed to 0b10 for 32 bits. If the 16-bit devices are connected as a 16-bit wide system, program MW to 0b01, and the switches in Figure 7-2 would then move to the '01' position, which connects address signal A1 to pin A0 and so forth. Shifting the address automatically greatly simplifies hardware design and PC board layout. In addition, because all 23 external address lines can be used for 16- and 32-bit systems, the number of addressable locations doubles or quadruples (respectively) compared to memory controllers that do not shift the addresses. The EMC allows the identical number of 8-bit, half-word, and full-word locations to be addressed with the same external address bus.

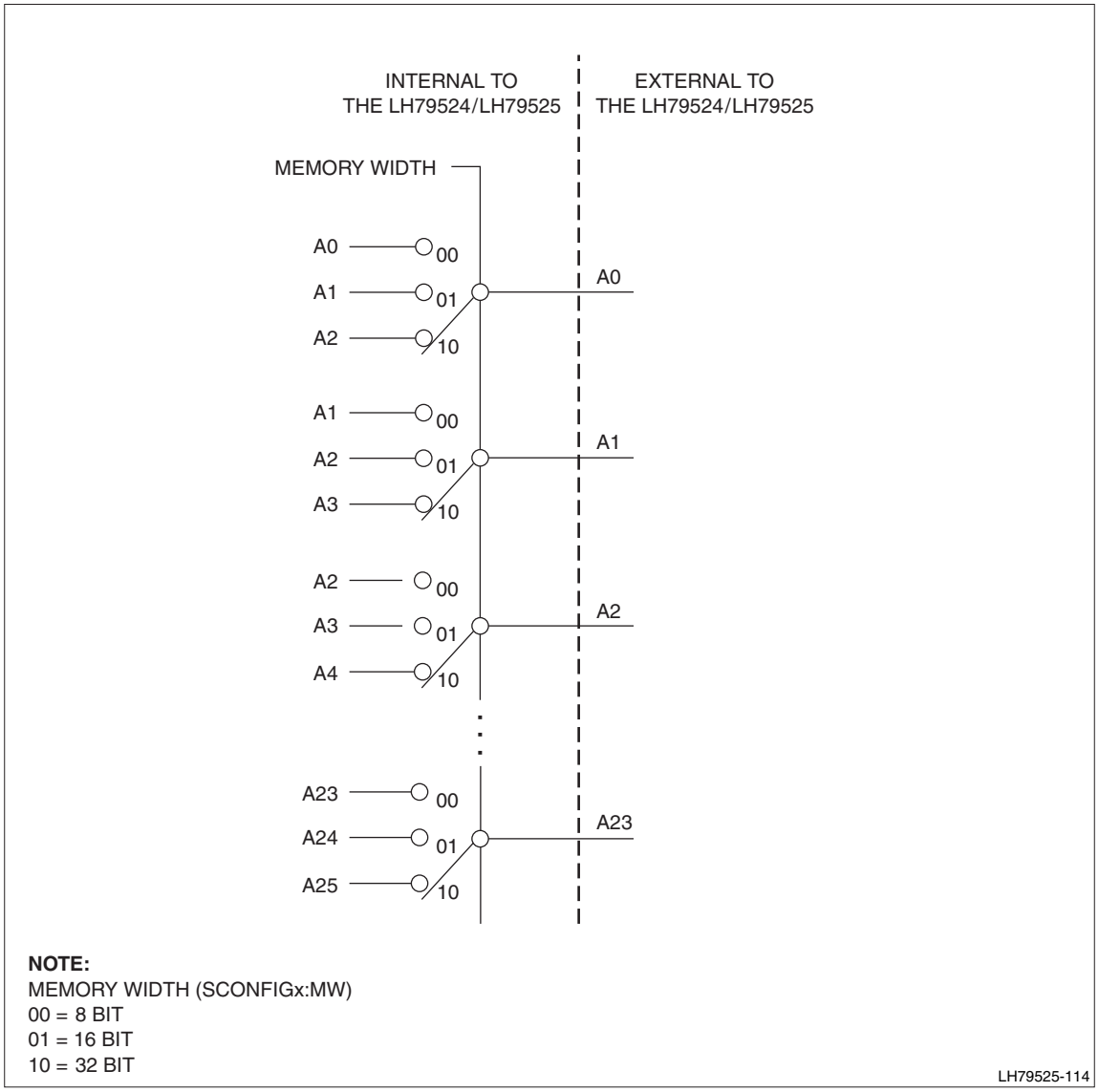


Figure 7-2. Automatic Address Shifting

7.2.2 Hardware Design

Automatic address shifting makes hardware design much simpler. This section provides a description and guide for hardware design and interfacing.

7.2.2.1 Address Connectivity

Rather than connecting different address pins to different memory devices depending on the width, MCU address pin A0 always connects to device pin A0, MCU address pin A1 to device pin A1, continuing through MCU pin A23 connecting to device pin A23.

7.2.2.1.1 Memory Banks of 8-bit or Non Byte-partitioned Memory Devices

For memory banks constructed from 8-bit or non byte-partitioned memory devices, it is important that the Byte Lane State (BLS) bit is cleared to 0 within the respective memory bank control register. This forces all nBLEx lines HIGH during a read access as the byte lane selects are connected to the device write enables.

Figure 7-3 through Figure 7-5 show 8-bit memory being used to configure memory banks that are 8, 16, and 32 bits wide. In each of these configurations, the nBLEx signals are connected to write enable (nWE) inputs of each 8-bit memory. The nWE signal is not used. For write transfers, the relevant nBLEx byte lane signals are asserted LOW, and steer the data to the address bytes. For read transfers, all of the nBLEx lines are deasserted HIGH, which enables the external bus to be defined for at least the width of the accessed memory.

7.2.2.1.2 Memory Banks of 16- or 32-bit Memory Devices

For memory banks constructed from 16- or 32-bit memory devices, it is important that the Byte Lane Select (BLS) bit is set to 1 within the respective memory bank control register. This asserts all nBLEx lines LOW during a read access as during a read all bytes of the device must be selected to avoid un-driven byte lanes on the read data value. In the case of 16- and 32-bit wide memory devices, byte select signals exist and these must be appropriately controlled, as shown in Figure 7-6 and Figure 7-7.

Figure 7-8 shows a connection for a typical memory system with different data width memory devices.

7.2.2.1.3 Address Connectivity, Address Right-justified

Figure 7-3 to Figure 7-8 show memory controller connection to static memory where the address is right justified. Right justification reduces the number of address pins and keeps the number of addresses constant, whether those addresses contain 8, 16, or 32 bits of data.

It's important to understand the difference between address *signals*, which are the actual address signals and the address *pins*, which carry the signals to the external memory bus.

When addressing 8-bit memory, address signals A[23:0] are mapped in a 1:1 correspondence with pins A[23:0], since transactions occur on byte boundaries. With 16-bit memory, address signal A0 is not necessary since all transactions occur on half-word boundaries. Thus, when the memory width is set to 16 bits, the address *pins* contain the right-justified address signals. Signal A0 is not used; signal A1 appears on pin A0; signal A2 appears on pin A1, and so on through signal A24 appearing on pin A23. With 32-bit memory, right-justification omits signals A0 and A1. So, for 32-bit memory, signal A2 appears on pin A0, and signal A25 appears on pin A23.

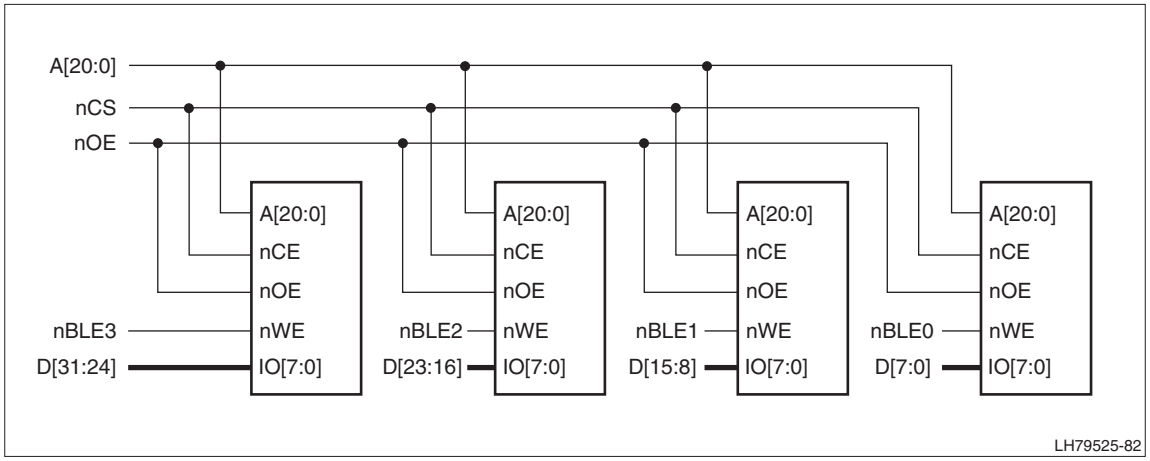


Figure 7-3. 32-bit Memory Bank Constructed From 8-bit Devices

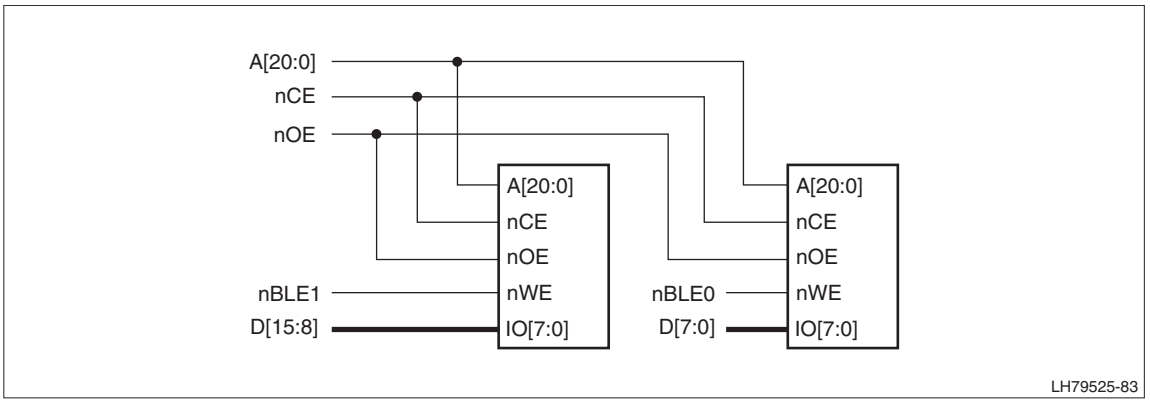


Figure 7-4. 16-bit Memory Bank Constructed From 8-bit Devices

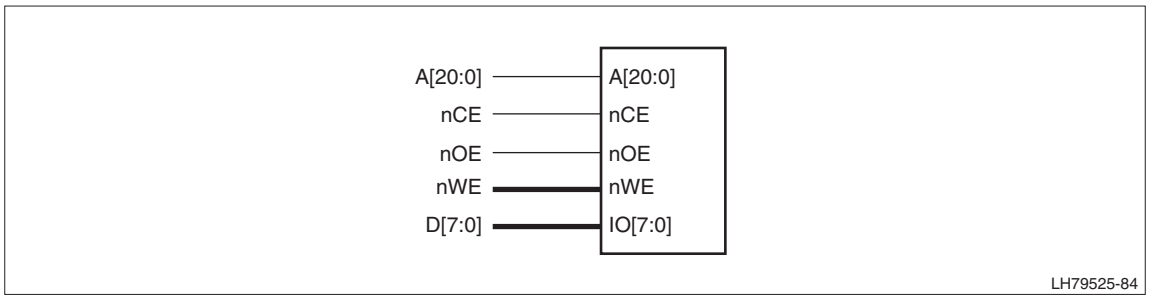


Figure 7-5. 8-bit Memory Bank

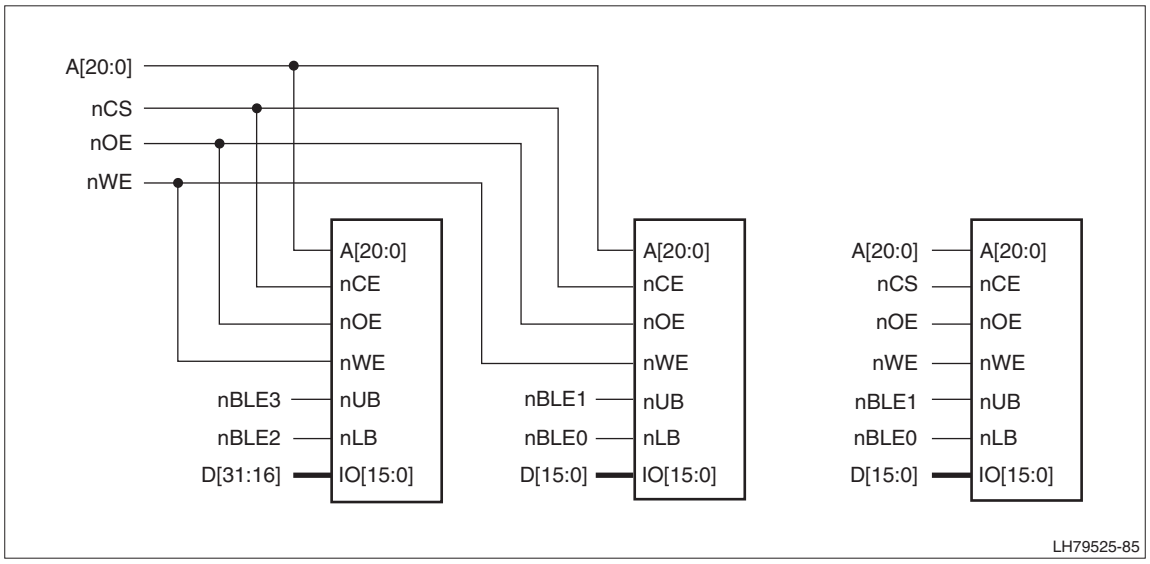


Figure 7-6. 32-bit (left) and 16-bit (right) Memory Banks Constructed From 16-bit Devices

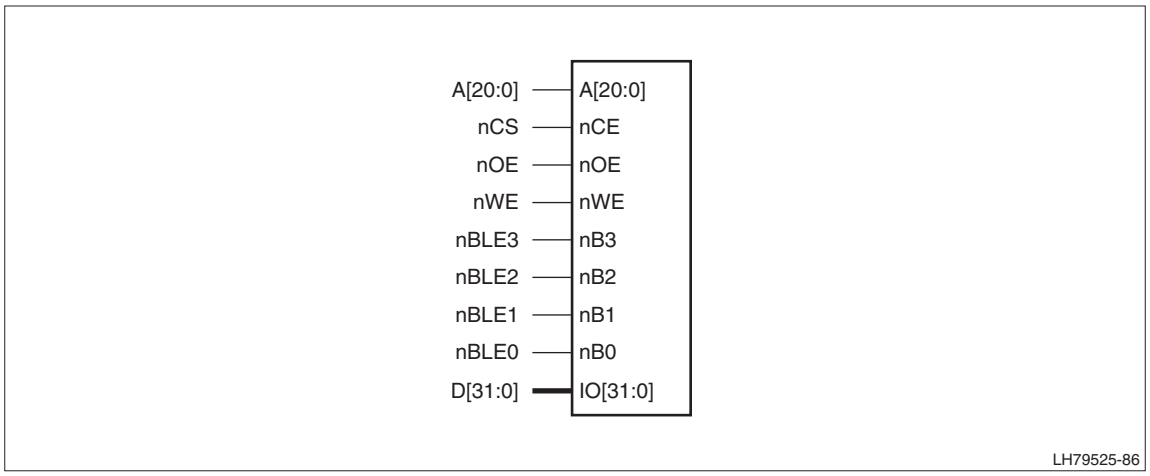


Figure 7-7. 32-bit Memory Bank Constructed From a Single 32-bit Device

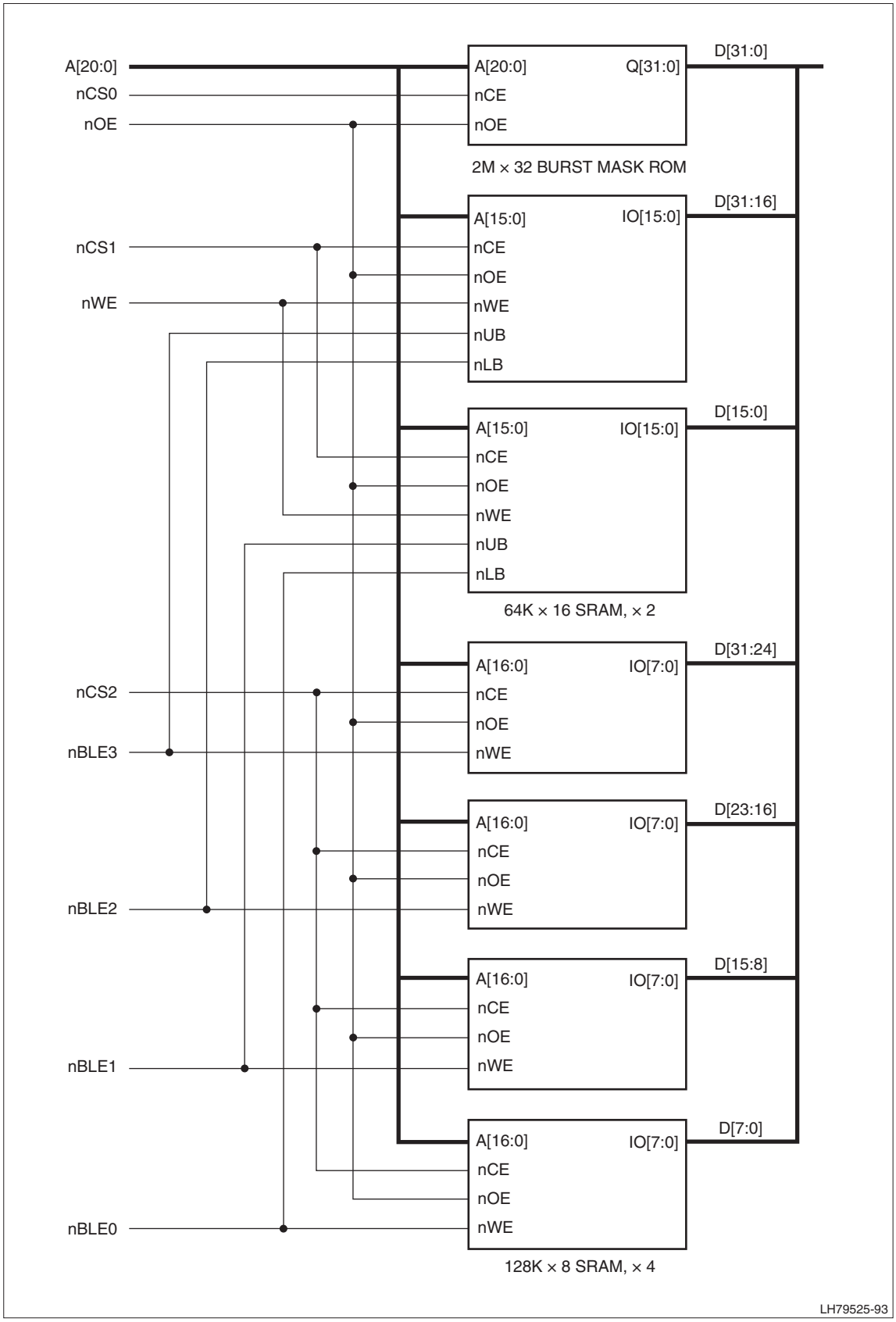


Figure 7-8. Typical Memory Connection Diagram

7.2.3 Software Design

For the bulk of software designs, the automatic address shifting is completely transparent and no software considerations are needed. However, in instances where software must control the signal on a specific address *pin*, the design must account for any address shifting.

7.2.3.1 Simple Shifting Subroutine

If the application requires specific signals on specific address lines, it may be necessary to pre-shift the address before executing a Read or Write. When addressing 16- or 32-bit wide devices, a subroutine based on the flow chart in Figure 7-9 handles the necessary pre-shifting. 'Device Width' can be determined by reading the SCONFIGx:MW field.

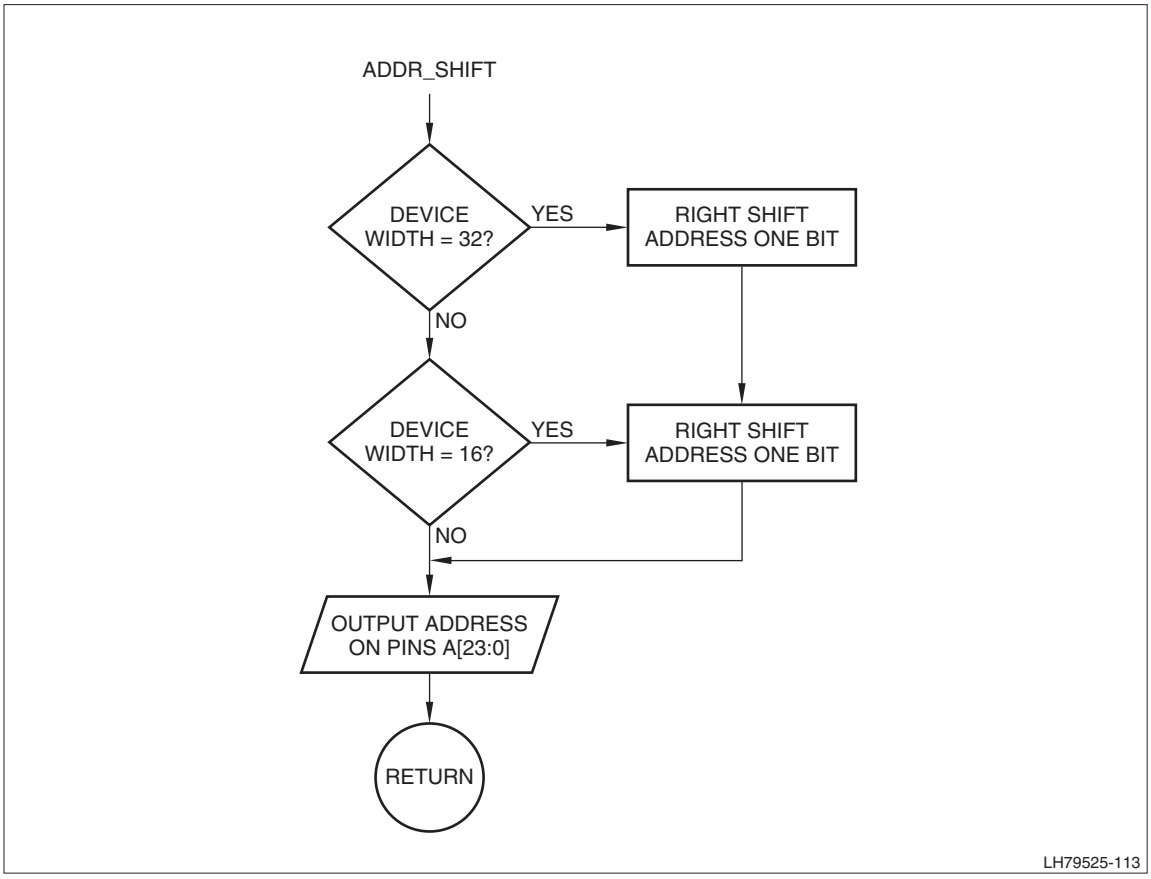


Figure 7-9. Pre-shifting Routine

7.2.4 Static Memory Device Selection

Table 7-1 shows suggested configurations for the static memory controller with different types of memory devices. These fields are found in the SCONFIG register.

Table 7-1. Static Memory Configurations

DEVICE	WRITE PROTECT	PAGE MODE	BUFFER	NOTES
ROM	Enabled	Disabled	Disabled	
Page Mode ROM	Enabled	Enabled	Enabled	
Extended wait ROM	Enabled	Disabled	Disabled	
SRAM	Disabled (or enabled)	Disabled	Disabled	1
Page Mode SRAM	Disabled (or enabled)	Enabled	Enabled	1
Extended Wait SRAM	Disabled (or enabled)	Disabled	Disabled	1
Flash	Disabled (or enabled)	Disabled	Disabled	1, 2
Page Mode Flash	Disabled (or enabled)	Enabled	Enabled	1, 2
Extended Wait Flash	Disabled (or enabled)	Disabled	Disabled	1
Memory mapped peripheral	Disabled (or enabled)	Disabled	Disabled	1

NOTES:
1. SRAM and FLASH memory devices can be write protected if required.
2. Buffering must be disabled when performing FLASH memory commands and during writes.
3. Enabling the buffers means that any access causes the buffer to be used. Depending on the application, this can provide performance improvements. Generally, devices without async-page-mode support work better with the buffer disabled. Depending on the application, this can provide performance improvements.

7.2.4.1 Static Memory Timing Control

Programming the EMC to match the device timing uses the static register bank (See Table 7-10). For Writes, the SWAITWENx and SWAITWRx registers contain programming parameters; for Reads, SWAITOENx and SWAITRDx are the registers. The SWAITWRx and SWAITRDx registers allow programming wait states, and the SWAITWENx and SWAITOENx register allow delaying assertion of the nWE/nBLEx/nOE signals.

The BTC field in the Bank Control Register sets the number of bus turnaround wait states added between external read and write transfers.

For ease in describing the memory timing, letters are used in the following diagrams to represent the values programmed into the above registers: 'A' = SWAITWENx; 'B' = SWAITWRx; 'C' = the 1 HCLK-cycle address delay; 'D' = SWAITOENx; and 'E' = SWAITRD. These diagrams are intended solely to illustrate programming effects. Actual timing diagrams and timing tables appear in the LH79524/LH79525 Data Sheet.

7.2.4.1.1 Read Cycle Wait States

Figure 7-10 shows the Read cycle with zero wait states. As shown in the Figure, SWAITOENx and SWAITRDx (refer to Section 7.5.2.22 and Section 7.5.2.23 for register descriptions) are both programmed to zero, for minimum Read Cycle time. With SWAITOENx programmed to zero, there is no delay of the nOE signal, and it is asserted coincident with the nCSx signal. The zero programmed into the SWAITRDx indicates that the read occurs with zero wait states, on the first rising edge following Address Valid. This time is shown as 'E0' on the diagram. After a small propagation delay, nOE is deasserted (as is nCSx), latching the data into the MCU. The address line is held valid one more HCLK period, denoted by 'C' in the diagram.

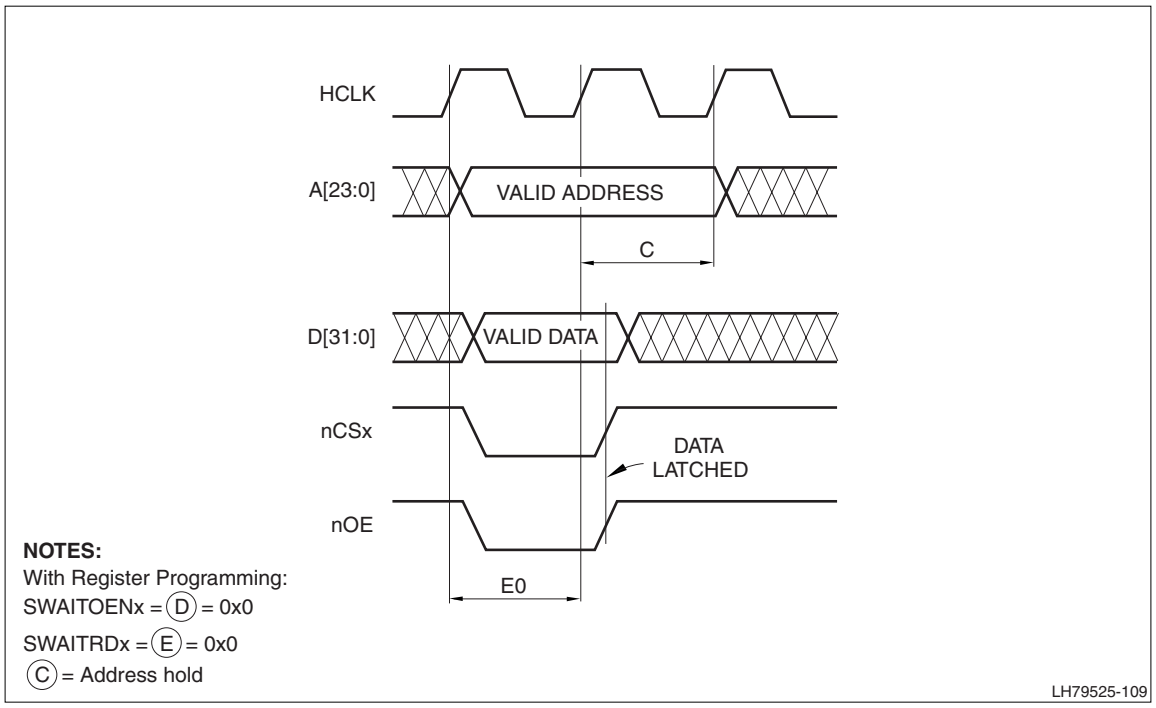


Figure 7-10. Static Read Transaction with Zero Wait States

Wait states are programmed using the SWAITRDx register. Figure 7-11 shows the results of programming SWAITRDx ('E') to 0x3, creating three wait states.

In the Figure, Timing A illustrates programming SWAITOENx to 0x0 and SWAITRDx to 0x3. With no wait states, the date would be read on the rising edge of nOE one HCLK cycle following Address Valid, that time represented by 'E0'. With the wait states, the transaction is extended by times 'E1', 'E2', and then the data is latched at the conclusion of time 'E3'. Thus, programming SWAITRDx to 0x3 causes three wait states, each of duration tHCLK (one HCLK period).

In Timing B, three wait states are also illustrated. However, in this case, the three wait states have been programmed by setting SWAITRDx to 0x2 and SWAITOENx to 0x1. In this case, nOE does not become asserted at the same time as nCSx. Instead, nOE is delayed by the number of HCLK periods enumerated by SWAITOENx, which in this case is one, represented by time 'D1'. Then, instead of the nOE signal deasserting one HCLK period later, it is extended two wait states because of the programmed value in SWAITRDx. The result is the same delay of three HCLK periods, but achieved in a different way to also delay assertion of nOE.

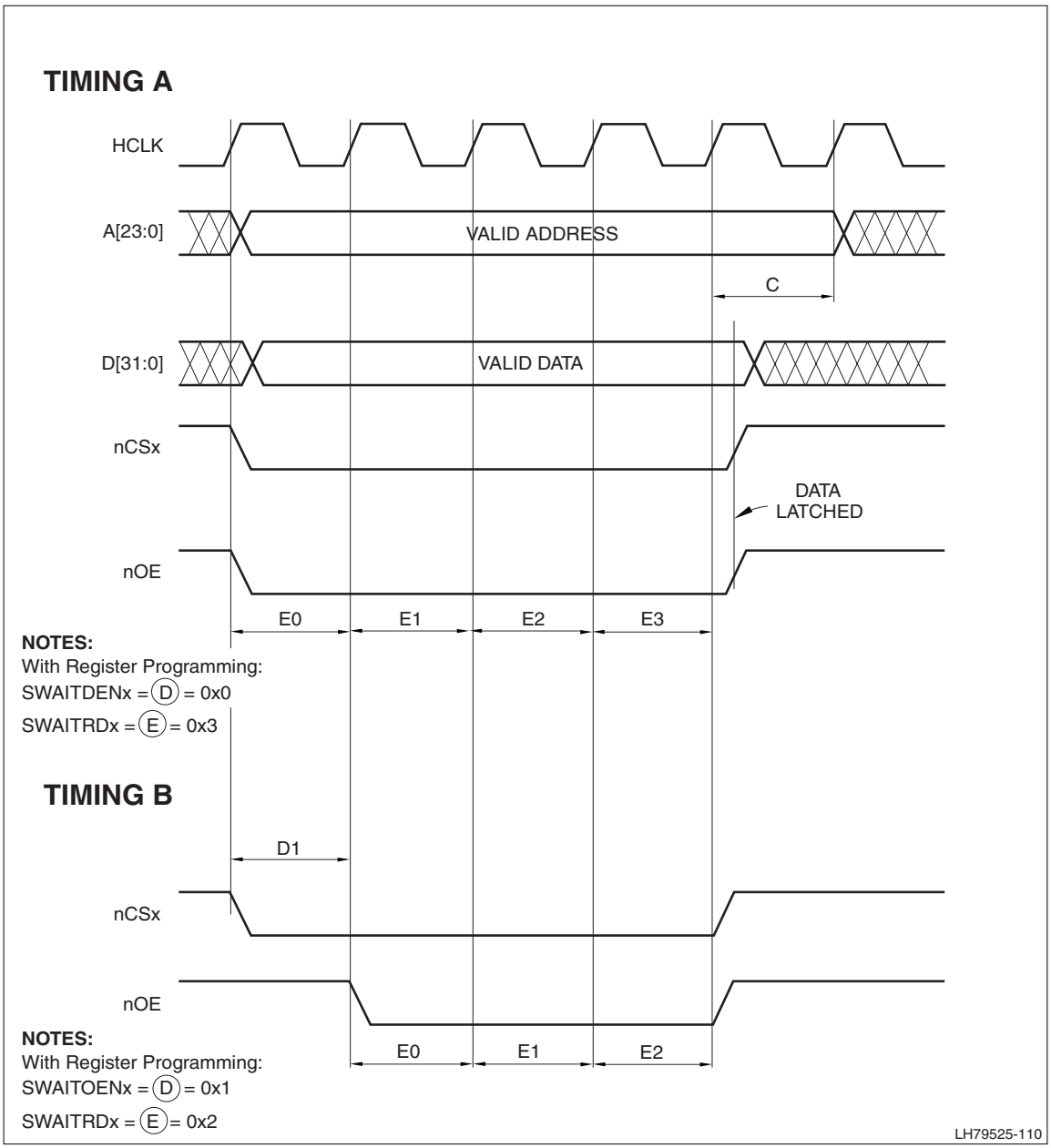


Figure 7-11. Static Read Transaction with Three Wait States

The total Read cycle time is the time that the address is valid (in the figures, until the end of the 'C' time). In general, Read wait states can be derived from the following equation:

tRC (Read cycle time) = tD1 + tD2 + ... tDn + tE0 + tE1 + ... tEn + C, where the length of each term is one HCLK period, and 'n' is the value programmed in the respective register.

The minimum value for the equation is tRC = tE0 + C, and is thus zero wait state timing.

Thus, Read wait states can be programmed using the appropriate mix of nOE extension (programmed in SWAITRDx) with nOE assertion delay (programmed in SWAITOENx).

7.2.4.1.2 Write Cycle Wait States

Write timing starts with assertion of the appropriate memory bank chip selects nCSx and address signals A[23:0]. The write access time is determined by the number of wait states programmed in the SWAITWRx register. Figure 7-12 shows the minimum write cycle time with both SWAITWRx ('A') and SWAITWENx ('B') programmed to zero.

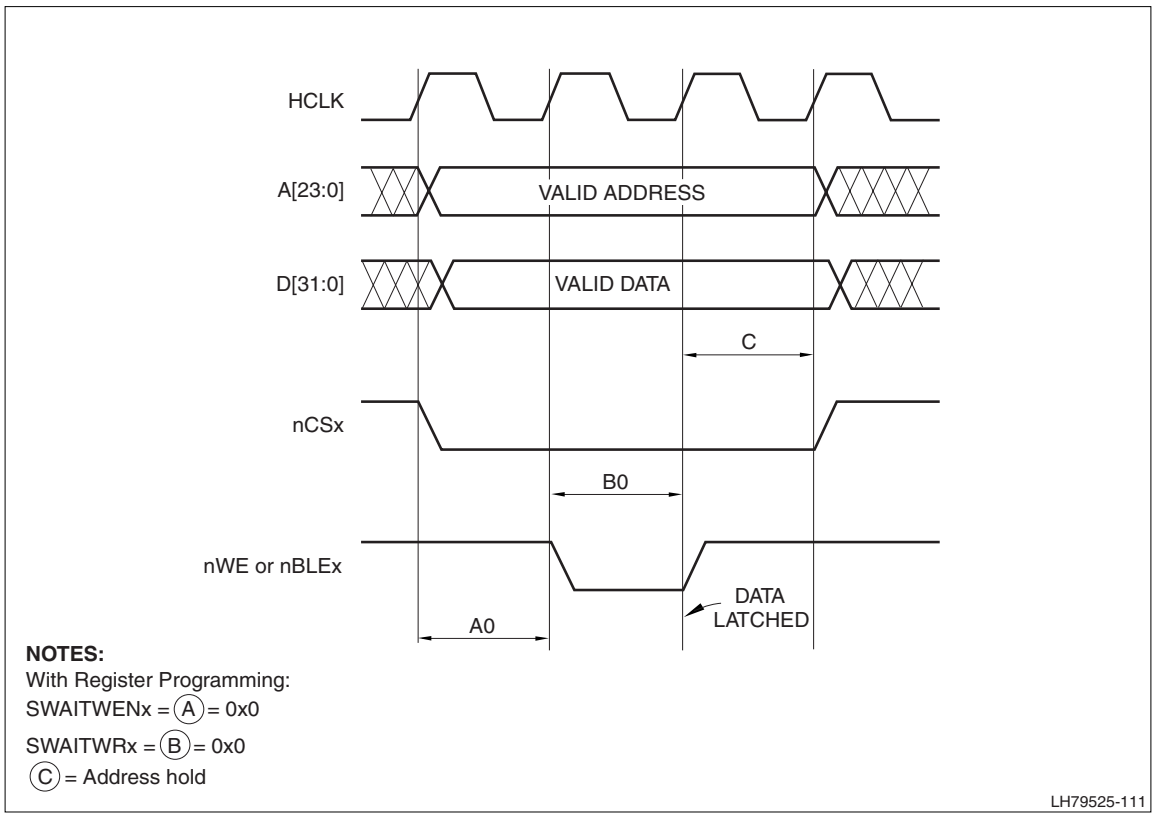


Figure 7-12. Static Write Transaction with Zero Wait States

In the Figure, nCSx is asserted coincident (with a small propagation delay) with the address becoming valid (Valid Address in the Figure). Data becomes valid another small propagation delay later. Unlike Read transactions, nWE (or nBLEx) assertion is always delayed one HCLK cycle, time 'A0' in the Figure. The nBLEx signal has the same timing as nWE for write to 8-bit devices that use the byte lane enables instead of the write enables.

The nWE (or nBLEx) signal remains asserted for one HCLK cycle, denoted 'B0' in Figure 7-12. At the end of 'B0', the nWE (or nBLEx) signal is deasserted and the data is latched into the external memory device. Valid address is held for one additional cycle before deassertion, as is the Chip Select.

Wait states behave slightly differently for Write transactions than for Reads. Instead of the length of the Write cycle (tWC) being the sum of the value programmed into the SWAITWENx and SWAITWRx registers, it has the following relationship:

tWC = tA0 + tB0 + tB1 ... tBn + C, where the length of each term is one HCLK period, and 'n' is the value programmed in the respective register.

The minimum value for the equation is tWC = tA0 +tB0 + C, and is therefore the zero wait state timing.

Note that additional 'A' terms (delaying the assertion of nWE or nBLEx) do not add wait states. This also requires that: SWAITWRx $\geq$ SWAITWENx.

Figure 7-13 shows the results of programming the SWAITWRx and SWAITWENx registers. In Timing A, SWAITWENx ('A') = 0x0, and SWAITWRx ('B') = 0x2. As always, nCSx precedes nWE (nBLEx) by one HCLK period (A0). Then, instead of the nWE (nBLEx) signal deasserting one HCLK period later at the end of B0, it is delayed two wait states, B1 and B2, and the signal deasserts on the rising edge at the end of B2.

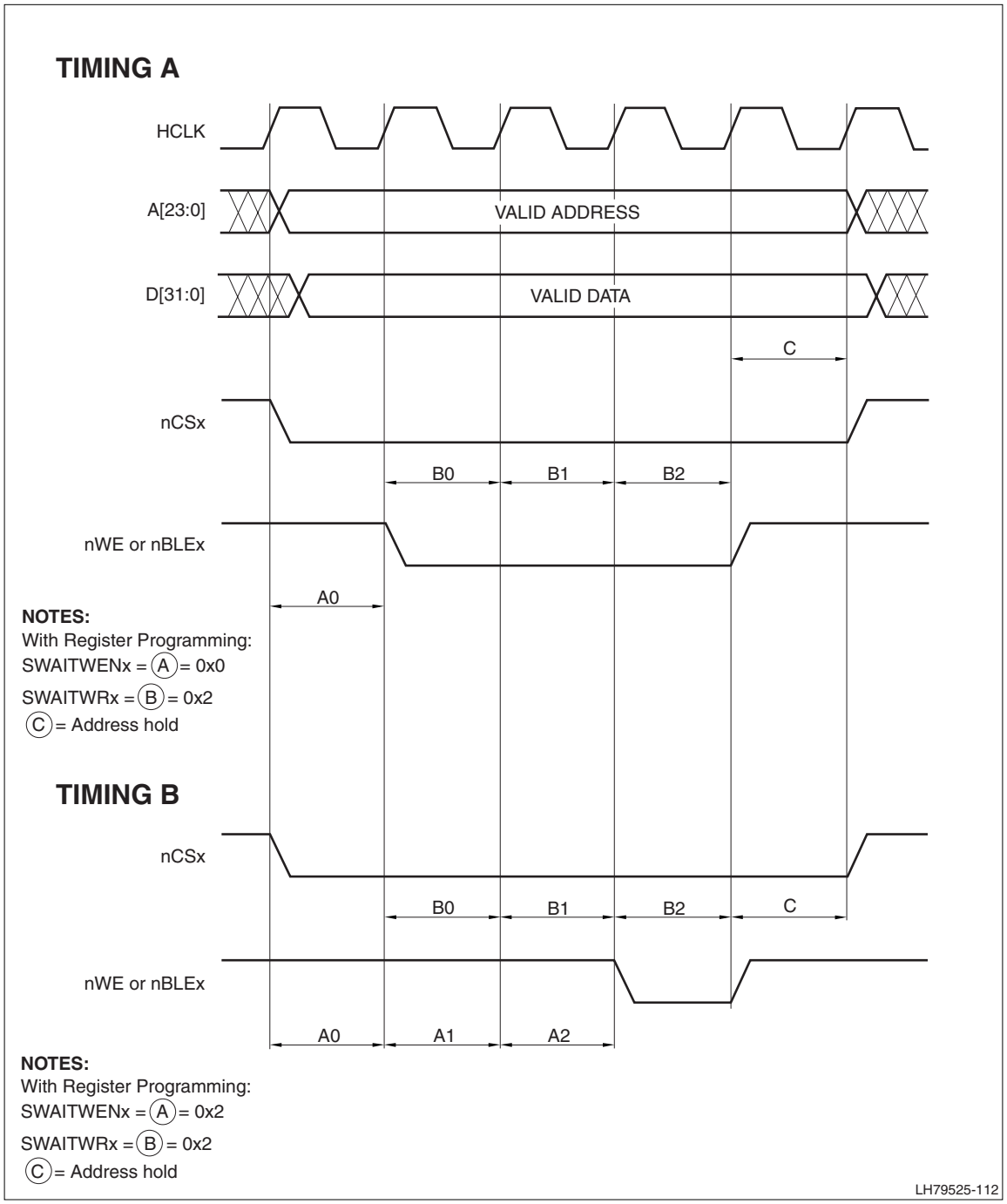


Figure 7-13. Static Write Transaction with Two Wait States

In Timing B, SWAITWENx ('A') = 0x2, and SWAITWRx ('B') = 0x2. As always, nCSx precedes nWE (nBLEx) by one HCLK period (A0). But instead of asserting at the end of A0, assertion is delayed by A1 and A2. However, according to the Write cycle equation, those terms do not delay the total cycle. As shown in the Figure, they are coincident with B0 and B1. The nWE (nBLEx) signal then asserts at the conclusion of A2 and deasserts at the conclusion of B2. Obviously, if SWAITWENx > SWAITWRx, no nWE (nBLEx) would exist. As with Reads, multiple register values produce the same number of wait states. For Write transactions, only the contents of SWAITWRx specify the number of wait states.

7.2.4.2 Bus Turnaround

The EMC can be configured for each memory bank to use external bus turnaround cycles between read and write memory accesses. The STURNx register can be programmed for 1 to 16 bus-turnaround wait states to avoid contention on the external data bus. Bus turnaround cycles are generated between external bus transfers: Read to Read (different memory banks), Read to Write (same memory bank), Read to Write (different memory banks).

7.2.4.3 Byte Lane Control

The EMC generates signals nBLE[3:0] according to:

- Internal memory transfer
- External memory bank data bus width, defined within each Control register
- Decode HADDR[1:0] for write accesses only.

Word transfers (32-bit) are the largest size transfers supported by the memory controller. Any access attempted with a size greater than a word causes an ERROR response to be generated. Each memory Chip Select space can be 8, 16, or 32 bits wide. The type of memory used determines how the nWE and nBLEx signals are connected to provide byte, half-word and word access. For read accesses, the nBLEx signals must be either all HIGH, or all LOW. This is done by programming the Byte Lane State (BLS) bit in the SCONFIG register (SCONFIG:BLS).

7.2.4.4 Write Protection

Each static memory Chip Select space can be configured for write-protection. Usually SRAM is unprotected and ROM devices must be write-protected (to avoid potential bus conflict when performing a write access to ROM), the SCONFIG:P field can be set to write-protect SRAM as well as ROM devices.

7.2.4.5 nWAIT Transfers

The EMC's Static Memory Controller supports an nWAIT input that can be used by an external device to extend the wait time during a memory access. The SMC samples nWAIT at the beginning of each system clock cycle. The system clock cycle in which the nCSx signal is asserted counts as the first wait state. Timing diagrams and extensive descriptions appear the the LH79524/LH79525 Data Sheet.

7.2.4.6 Extended Wait Transfers

The static memory controller supports extremely long transfer times. In normal use the memory transfers are timed using the SWAITRDx and SWAITWRx registers. These registers enable transfers with up to 32 wait states. However, if an extremely slow static memory device has to be accessed, enable the SCONFIG Extended Wait (EW) bit. When this bit is enabled the WAIT register is used to time both the read and write transfers. This register enables transfers to have up to 16,368 wait states.

NOTES:
1. Using extremely long transfer times might mean that SDRAM devices are not refreshed correctly.
2. Very slow transfers can severely degrade system performance as the external memory interface is tied up for long periods of time. This has detrimental efforts on time critical services, such as interrupt latency and low latency devices, for example the CLCDC.

7.3 Interfacing with NAND Flash

The EMC interface for NAND Flash has two modes: one for booting and one for accesses during normal program execution. During boot, the Boot ROM handles the logic for generating control signals on the four address lines used for the Address Latch Enable (ALE), Command Latch Enable (CLE), Flash Write Enable (nFWE), and Flash Read Enable (nFRE) signals. When used for booting, the NAND Flash must be selected with Chip Select nCS0, which has been programmed as GPIO PM0. If not used for booting, the NAND Flash device can be selected with any Chip Select signal. See Chapter 3 and the examples that follow for more information about the Boot Controller.

7.3.1 Booting Example

Booting from NAND Flash is defined by the static signals on Port C[7:4] at the rising edge of nRESETOUT, as shown in Table 7-2 and Table 7-3. Whenever PC6 is HIGH, boot occurs from a NAND Flash (or via I2C or UART, which do not require a Chip Select). Unlike booting from other devices, NAND Flash must be located in the nCS0 Chip Select domain.

If the application uses the on-board Boot ROM to direct a boot from external NAND Flash, four address lines function as NAND Flash control pins. Connection of the MCU to the NAND Flash is illustrated in Figure 7-14.

During boot, the Boot ROM in the LH79524/LH79525 automatically controls the logic to present the proper signals at the proper times on the address lines acting as control signals. Thus, care must be taken to either not use nCS0 for other devices, or ensure that the NAND Flash is not inadvertently accessed by addresses in the nCS0 address space.

In addition, the code in the Boot ROM automatically shifts the addresses, so the control and address signals appear on the same pins for any width NAND Flash. Thus, the connection diagram in Figure 7-14 shows correct implementation for all memory widths.

Table 7-2. Boot Configuration for Silicon Version A.0

PC[7:4]	DEVICE TYPE	DATA BUS WIDTH	CONTROL
0x0	NOR Flash or SRAM	16-bit	nBLEx LOW for Reads
0x1	NOR Flash or SRAM	16-bit	nBLEx HIGH for Reads
0x2	NOR Flash or SRAM	8-bit	nBLEx LOW for Reads
0x3	NOR Flash or SRAM	8-bit	nBLEx HIGH for Reads
0x4	NAND Flash (Small Block)	8-bit	3-byte Address
0x5	NAND Flash (Small Block)	8-bit	4-byte Address
0x6	NAND Flash (Small Block)	8-bit	5-byte Address
0x7	NAND Flash (Small Block)	16-bit	3-byte Address
0x8	NOR Flash or SRAM	32-bit	nBLEx LOW for Reads
0x9	NOR Flash or SRAM	32-bit	nBLEx HIGH for Reads
0xA	RESERVED	///	///
0xB	RESERVED	///	///
0xC	NAND Flash (Small Block)	16-bit	4-byte Address
0xD	NAND Flash (Small Block)	16-bit	5-byte Address
0xE	RESERVED	Reserved	Reserved
0xF	RESERVED	Reserved	Reserved

Table 7-3. Boot Configuration for Silicon Version A.1

PC[7:4]	DEVICE TYPE	DATA BUS WIDTH	CONTROL
0x0	NOR Flash or SRAM	16-bit	nBLEx LOW for Reads
0x1	NOR Flash or SRAM	16-bit	nBLEx HIGH for Reads
0x2	NOR Flash or SRAM	8-bit	nBLEx LOW for Reads
0x3	NOR Flash or SRAM	8-bit	nBLEx HIGH for Reads
0x4	NAND Flash (Small Block)	8-bit	3-byte Address
0x5	NAND Flash (Small Block)	8-bit	4-byte Address
0x6	NAND Flash (Small Block)	8-bit	4/5-byte Address
0x7	NAND Flash (Large Block)	16-bit	3-byte Address
0x8	NOR Flash or SRAM	32-bit	nBLEx LOW for Reads
0x9	NOR Flash or SRAM	32-bit	nBLEx HIGH for Reads
0xA	RESERVED	///	///
0xB	RESERVED	///	///
0xC	NAND Flash (Small Block)	16-bit	4-byte Address
0xD	NAND Flash (Large Block)	16-bit	4/5-byte Address
0xE	I^2C	—	—
0xF	UART	—	—

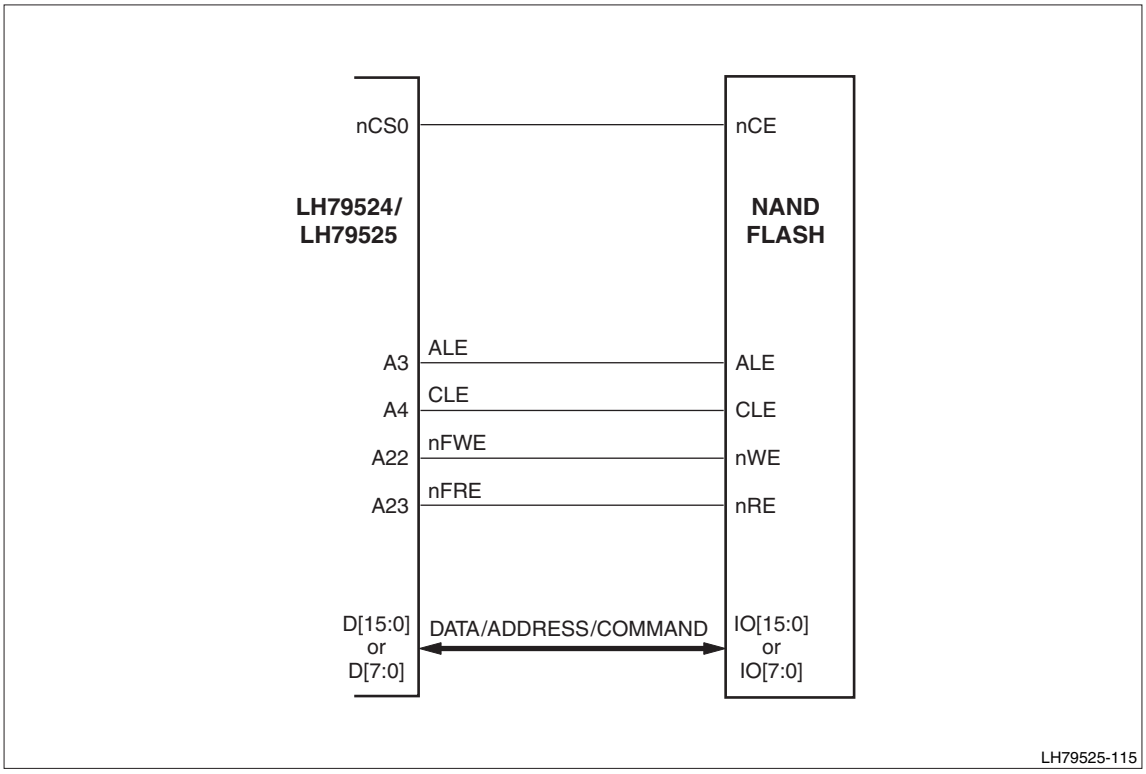


Figure 7-14. Connection to NAND Flash

7.3.2 General NAND Flash Access

At all other times but boot, all address lines function as addresses, and the NAND Flash control signals must be generated by the application software. Unlike booting, where the internal boot code automatically translates the addresses as required by the data width, in general application use, the software must perform this translation prior to writing to the NAND Flash.

7.3.2.1 Transaction Example

This example uses the design in Figure 7-14, with the four address lines connected to ALE, CLE, nWE, and nRE on the NAND Flash. The design can, obviously, use any address lines or GPIO to control the device.

Since NAND Flash devices are available in several widths, it is up to software to ensure that the proper control signals appear on the proper address pins for correct operation (Recall from Section 7.2.2.1.3 that the MCU right justifies addressing).

For example, using pins D[15:0] to communicate with the NAND Flash requires programming SCONFIGx:MW to 0b01 for 16-bit wide external memory transactions. This also causes the automatic address shift to place the A1 address signal on pin A0. Thus, if the design uses a 16-bit NAND Flash, software must assure TRUE conditions on the address signals for the operation being executed, as shown in Table 7-4. A 16-bit addressing example appears in Section 7.3.2.3.

Table 7-4. 16-bit Address Mapping

SIGNAL	OUTPUT PIN
A3	A2 (ALE)
A4	A3 (CLE)
A23	PC6/A22/nFWE
A24	PC7/A23/nFRE

7.3.2.2 16-bit Example Transaction

A quick example illustrates one way for software to handle 16-bit NAND Flash transactions. This example shows a simple data Write to the NAND Flash, but Reads and Commands work similarly. Figure 7-15 shows the timing generated by the following sequence.

With all control signals FALSE, the address of the location to be written in the NAND Flash is placed on the LH79524/LH79525 D[15:0] pins ('A' in the Figure). Software, with the proper signals on D[15:0], then programs a Write to location 0xCXXX10, causing ALE and nFWE to go HIGH ('B'). With the same data on D[15:0], a write to location 0x4XXX10 causes nFWE to go LOW ('C'). Writing to location 0xCXX10 again drives nFWE HIGH, which latches the address in the NAND Flash ('D'). Finally, to clear the interface, write to address 0xCXXX00, forcing ALE to LOW ('E').

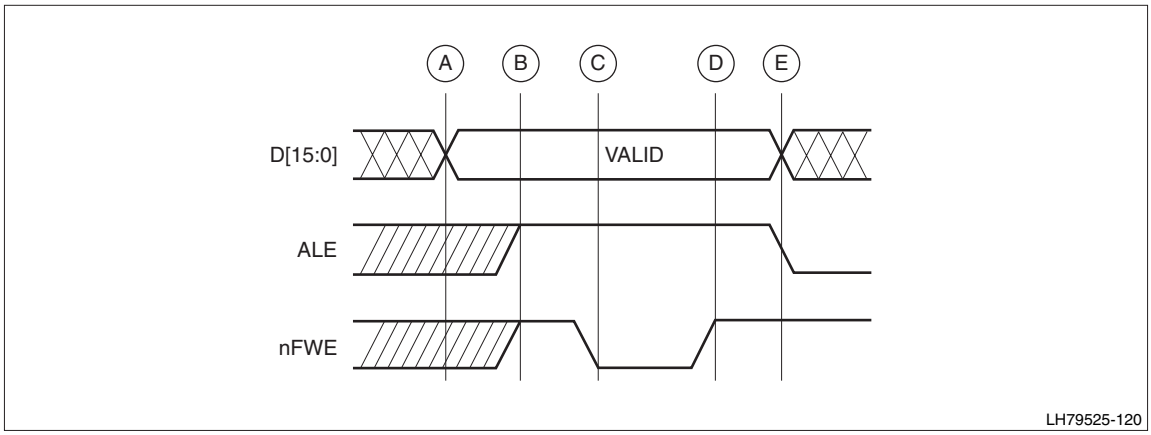


Figure 7-15. NAND Flash Timing Example

7.3.2.3 Address Examples

If nCS0 is used to connect 8-bit NAND Flash, use the following addresses:

- 0x40800000 — to read from NAND Flash
- 0x40800008 — to write address to NAND Flash
- 0x40800010 — to write command to NAND Flash.

If nCS0 is used to connect 16-bit NAND Flash, use the following addresses:

- 0x41000000 — to read from NAND Flash
- 0x41000010 — to write address to NAND Flash
- 0x41000020 — to write command to NAND Flash.

Note in the 16-bit addressing, the lower 24 bits have just been left-shifted one bit to compensate for the right-justified addressing of the EMC.

7.4 Dynamic Memory

7.4.1 Write-protection

Each dynamic memory Chip Select can be configured for write-protection by setting the relevant bit in the write-protect field in the DYNCFGx register (DYNCFGx:P). If a write access is performed to a write-protected memory bank, an ERROR response is generated on the HRESP[1:0] signal. For more information on HRESP see the ARM AMBA specification.

7.4.2 Access Sequencing and Memory Width

Configure the data width of each external Chip Select space by programming the appropriate DYNCFGx register. If the Chip Select data bus width is narrower than the request from the current AMBA bus master, the internal bus transfer takes several external bus transfers to complete. For example, if Chip Select 1 (nDCS1) is configured as 16-bit wide memory and a 32-bit read is initiated, the AHB stalls while the EMC reads two consecutive words from the memory. During these accesses the EMC demultiplexes the data into one 32-bit word and places it on the AHB. Word transfers are the widest transfers supported, and access attempted with a size larger than a word generates an error response.

7.4.3 Bus Address Mapping

These tables provide the mapping of AHB address bus addresses to the external dynamic memory address for various memory configurations and bus widths. The address mapping is selected by programming the Address Mapping (AM) bits in the DYNCFGx register. Note that Auto Precharge is always presented on A10 column address.

Table 7-5. 32-bit Wide Data Bus Address Mapping, SDRAM (RBC)

32-BIT DEVICE 16M SDRAM (1M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	—	10/BA0	—	—	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	—	10/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 16M SDRAM (2M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	—	—	—	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	—	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 64M SDRAM (2M × 32, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	—	—	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 64M SDRAM (4M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	—	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2

Table 7-5. 32-bit Wide Data Bus Address Mapping, SDRAM (RBC) (Cont'd)

32-BIT DEVICE 64M SDRAM (8M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	—	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (4M × 32, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	—	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (8M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	—	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (16M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	13/BA1	12/BA0	—	25	24	23	22	21	20	19	18	17	16	15	14
AHB Address To Column Address	13/BA1	12/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM 8M × 32, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	24	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM 16M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	25	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM 32M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	13/BA1	12/BA0	26	25	24	23	22	21	20	19	18	17	16	15	14
AHB Address To Column Address	13/BA1	12/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 512M SDRAM 32M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	13/BA1	12/BA0	26	25	24	23	22	21	20	19	18	17	16	15	14
AHB Address To Column Address	13/BA1	12/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 512M SDRAM 64M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	13/BA1	14/BA0	27	26	25	24	23	22	21	20	19	18	17	16	15
AHB Address To Column Address	13/BA1	14/BA0	—	12	AP	11	10	9	8	7	6	5	4	3	2

NOTE: The EMC does a constant burst of 8 when configured to access 16 bit memory devices. Similarly it does a burst of 4 when configured to access 32-bit memory devices.

Table 7-6. 32-bit Wide Data Bus Address Mapping, SDRAM (BRC)

32-BIT DEVICE 16M SDRAM (1M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	21/BA1	—	—	—	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	21/BA1	—	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 16M SDRAM (2M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	—	22/BA0	—	—	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	—	22/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2

Table 7-6. 32-bit Wide Data Bus Address Mapping, SDRAM (BRC) (Cont'd)

32-BIT DEVICE 64M SDRAM (2M × 32, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	21/BA1	22/BA0	—	—	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	21/BA1	22/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 64M SDRAM (4M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	22/BA0	—	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23/BA1	22/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 64M SDRAM (8M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	24/BA0	—	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	23/BA1	24/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (4M × 32, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	22/BA0	—	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23/BA1	22/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (8M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	24/BA0	—	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	23/BA1	24/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 128M SDRAM (16M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	24/BA0	—	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	25/BA1	24/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM 8M × 32, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	24/BA0	22	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23/BA1	24/BA0	—	—	AP	—	—	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM (16M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	24/BA0	23	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	25/BA1	24/BA0	—	—	AP	—	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 256M SDRAM (32M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	26/BA0	24	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	25/BA1	26/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 512M SDRAM (32M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	26/BA0	24	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	25/BA1	26/BA0	—	—	AP	11	10	9	8	7	6	5	4	3	2
32-BIT DEVICE 512M SDRAM (64M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	27/BA1	26/BA0	25	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	27/BA1	26/BA0	—	12	AP	11	10	9	8	7	6	5	4	3	2

Table 7-7. 16-bit Wide Data Bus Address Mapping, SDRAM (RBC)

16-BIT WIDE DEVICE 16M SDRAM (1M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	9/BA1	—	—	—	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	9/BA1	—	—	—	AP	—	—	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 16M SDRAM (2M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	—	10/BA0	—	—	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	—	10/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 64M SDRAM (4M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	9/BA1	10/BA0	—	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	9/BA1	10/BA0	—	—	AP	—	—	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 64M SDRAM (8M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	—	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 128M SDRAM (8M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	—	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	10/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 128M SDRAM (16M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	—	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 256M SDRAM (16M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	10/BA0	24	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 256M SDRAM (32M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	25	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 512M SDRAM (32M × 16, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	11/BA1	12/BA0	25	24	23	22	21	20	19	18	17	16	15	14	13
AHB Address To Column Address	11/BA1	12/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 512M SDRAM (64M × 8, RBC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	13/BA1	12/BA0	26	25	24	23	22	21	20	19	18	17	16	15	14
AHB Address To Column Address	13/BA1	12/BA0	—	11	AP	10	9	8	7	6	5	4	3	2	**

NOTE: **The EMC always transfers 32 bits of data in each transaction. For chip selects with a 16-bit wide data bus, the EMC performs two transfers: 1) a column transfer with the lowest bit set to 0, and 2) a column transfer with the lowest bit set to 1.

Table 7-8. 16-bit Wide Data Bus Address Mapping, SDRAM (BRC)

16-BIT WIDE DEVICE 16M SDRAM (1M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	—	20/BA0	—	—	19	18	17	16	15	14	13	12	11	10	9
AHB Address To Column Address	—	20	—	—	AP	–	–	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 16M SDRAM (2M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	21/BA1	—	—	—	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	21/BA1	—	—	—	AP	–	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 64M SDRAM (4M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	21/BA1	22/BA0	—	20	19	18	17	16	15	14	13	12	11	10	9
AHB Address To Column Address	21/BA1	22/BA0	—	—	AP	—	—	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 64M SDRAM (8M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	22/BA0	—	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23	22/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 128M SDRAM (8M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	22/BA0	—	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23/BA1	22/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 128M SDRAM (16M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	24/BA0	—	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	23/BA1	24/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 256M SDRAM (16M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	23/BA1	24/BA0	22	21	20	19	18	17	16	15	14	13	12	11	10
AHB Address To Column Address	23/BA1	24/BA0	—	—	AP	—	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 256M SDRAM (32M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	24/BA0	23	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	25/BA1	24/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 512M SDRAM (32M × 16, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	25/BA1	24/BA0	23	22	21	20	19	18	17	16	15	14	13	12	11
AHB Address To Column Address	25/BA1	24/BA0	—	—	AP	10	9	8	7	6	5	4	3	2	**
16-BIT WIDE DEVICE 512M SDRAM (64M × 8, BRC)															
External Address Pin, A[14:0]	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AHB Address To Row Address	26/BA1	25/BA0	24	23	22	21	20	19	18	17	16	15	14	13	12
AHB Address To Column Address	26/BA1	25/BA0	—	11	AP	10	9	8	7	6	5	4	3	2	**

NOTE: **The EMC always transfers 32 bits of data in each transaction. For chip selects with a 16-bit wide data bus, the EMC performs two transfers: 1) a column transfer with the lowest bit set to 0, and 2) a column transfer with the lowest bit set to 1.

7.4.4 Data Mask Signals

Depending on the external memory system width and the operand size, one or two memory cycles may be required for operand transfer. The Data Mask signals (DQM[3:0]) select the data phase for each cycle, as shown in Table 7-9.

- For 32-bit wide memory systems, only one memory cycle is required for any data transfer width, with the DQM bits configured on write cycles to disable bytes unaffected by the transfer.
- For 16-bit wide memory systems, DQM[1] is used as the memory system upper data mask (UDQM) and DQM[0] is used as the lower data mask (LDQM).
- For 32-bit transfers in 16-bit wide memory systems, two memory data phases are required to complete the memory cycles. Half word (16-bit) and byte-width transfers complete in one data phase.

Table 7-9. Memory System Examples

MEMORY SYSTEM	SIZE	DATA BUS	AHB PHYSICAL ADDRESS	BYTE ENABLES
16M by 16-bit	32MB	D[15:0]	A[24:1]	DQM[1:0]
16M by 32-bit	64MB	D[31:0]	A[25:2]	DQM[3:0]
64M by 16-bit	128MB	D[15:0]	A[26:1]	DQM[1:0]
64M by 32-bit	256MB	D[31:0]	A[27:2]	DQM[3:0]

7.5 Register Reference

The base address for the EMC is 0xFFFF1000.

7.5.1 Memory Map

Table 7-10 Summarizes the EMC registers.

Table 7-10. External Memory Controller Register Summary

ADDRESS OFFSET	REGISTER	DESCRIPTION
0x000	CONTROL	Control Register
0x004	STATUS	Status Register
0x008	///	Reserved; Writing to this register can result in unpredictable behavior. Do not write to this address.
0x00C to 0x01C	///	Reserved
0x020	DYNMCTRL	Dynamic Memory Control Register
0x024	DYNMREF	Dynamic Memory Refresh Timer
0x028	DYNRCON	Dynamic Memory Read Configuration Register
0x02C	///	Reserved
0x030	PRECHARGE	Dynamic Memory Precharge Command Period
0x034	DYNM2PRE	Dynamic Memory Active to Precharge Command Period
0x038	REFEXIT	Dynamic Memory Self-Refresh Exit Time
0x03C	DOACTIVE	Dynamic Memory Last Data Out to Active Time
0x040	DIACTIVE	Dynamic Memory Data-In to Active Command Time
0x044	DWRT	Dynamic Memory Write Recovery Time
0x048	DYNACTCMD	Dynamic Memory Active to Active Command Period
0x04C	DYNAUTO	Dynamic Memory Auto-Refresh Period, and Auto-Refresh to Active Command Period
0x050	DYNREFEXIT	Dynamic Memory Exit Self-Refresh to Active Command Time
0x054	DYNACTIVEAB	Dynamic Memory Active Bank A to Active Bank B Time
0x058	DYNAMICMRD	Dynamic Memory Load Mode Register to Active Command Time
0x05C to 0x07C	///	Reserved
0x080	WAIT	Static Memory Extended Wait
0x084 to 0x0FC	///	Reserved
0x100	DYNCFG0	Dynamic Configuration Register for nDCS0
0x104	DYNRASCAS0	Dynamic Memory RAS and CAS Delay for nDCS0
0x108 to 0x11C	///	Reserved
0x120	DYNCFG1	Dynamic Configuration Register for nDCS1
0x124	DYNRASCAS1	Dynamic Memory RAS and CAS Delay for nDCS1

Table 7-10. External Memory Controller Register Summary

ADDRESS OFFSET	REGISTER	DESCRIPTION
0x128 to 0x1FC	///	Reserved
0x200	SCONFIG0	Static Memory Configuration for nCS0
0x204	SWAITWEN0	Static Memory Write Enable Delay for nCS0
0x208	SWAITOEN0	Static Memory Output Enable Delay for nCS0
0x20C	SWAITRD0	Static Memory Read Delay for nCS0
0x210	SWAITPAGE0	Static Memory Page Mode Read Delay for nCS0
0x214	SWAITWR0	Static Memory Write Delay for nCS0
0x218	STURN0	Static Memory Turn Around Delay for nCS0
0x21C	///	Reserved
0x220	SCONFIG1	Static Memory Configuration for nCS1
0x224	SWAITWEN1	Static Memory Write Enable Delay for nCS1
0x228	SWAITOEN1	Static Memory Output Enable Delay for nCS1
0x22C	SWAITRD1	Static Memory Read Delay for nCS1
0x230	SWAITPAGE1	Static Memory Page Mode Read Delay for nCS1
0x234	SWAITWR1	Static Memory Write Delay for nCS1
0x238	STURN1	Static Memory Turn Around Delay for nCS1
0x23C	///	Reserved
0x240	SCONFIG2	Static Memory Configuration for nCS2
0x244	SWAITWEN2	Static Memory Write Enable Delay for nCS2
0x248	SWAITOEN2	Static Memory Output Enable Delay for nCS2
0x24C	SWAITRD2	Static Memory Read Delay for nCS2
0x250	SWAITPAGE2	Static Memory Page Mode Read Delay for nCS2
0x254	SWAITWR2	Static Memory Write Delay for nCS2
0x258	STURN2	Static Memory Turn Around Delay for nCS2
0x25C	///	Reserved
0x260	SCONFIG3	Static Memory Configuration for nCS3
0x264	SWAITWEN3	Static Memory Write Enable Delay for nCS3
0x268	SWAITOEN3	Static Memory Output Enable Delay for nCS3
0x26C	SWAITRD3	Static Memory Read Delay for nCS3
0x270	SWAITPAGE3	Static Memory Page Mode Read Delay for nCS3
0x274	SWAITWR3	Static Memory Write Delay for nCS3
0x278	STURN3	Static Memory Turn Around Delay for nCS3
0x21C	///	Reserved

7.5.2 Register Definitions

7.5.2.1 Control Register (CONTROL)

The CONTROL Register controls the memory controller operation. The control bits can be altered during normal operation.

Table 7-11. CONTROL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													MODE	///	ENABLE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
ADDR	0xFFFF1000 + 0x000															

Table 7-12. CONTROL Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	MODE	**Mode select** Entering low-power mode reduces memory controller power consumption. Dynamic memory is refreshed as necessary. The memory controller returns to normal functional mode by clearing the low-power mode bit. External memory cannot be accessed in low-power state. If a memory access is performed, an error response is generated. 1 = Low-power Mode 0 = Normal Mode
1	///	**Reserved** Reading returns 0. Write the reset value.
0	ENABLE	**Enable the EMC** Disabling the External Memory Controller reduces power consumption. When the memory controller is disabled the memory is not refreshed. The memory controller is enabled by setting the enable bit. The external memory cannot be accessed in disabled state. If a memory access is performed, an error response is generated. 1 = Enabled 0 = Disabled

7.5.2.2 Status Register (STATUS)

The STATUS Register provides memory controller status information.

Table 7-13. STATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													SA	WRBUF	BUSY
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF1000 + 0x004															

Table 7-14. STATUS Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	SA	**Self-refresh Acknowledge** This bit indicates the operating mode of the EMC. 1 = Self-refresh Mode 0 = Normal Mode
1	WRBUF	**Write Buffer Status** This enables the EMC to enter low-power mode or disabled mode cleanly by determining if the write buffers contain data or not. 1 = Write Buffers contain data 0 = Write Buffers empty
0	BUSY	**Busy** This read-only bit is used to ensure that they memory controller enters the low-power or disabled mode cleanly by determining if the memory controller is busy or not. 1 = EMC is busy performing memory transactions, commands, auto-refresh cycles, or is in self-refresh mode. 0 = EMC is idle

7.5.2.3 Dynamic Memory Control Register (DYNMCTRL)

The Dynamic Memory Control Register is used to control dynamic memory operation. The control bits can be altered during normal operation.

Table 7-15. DYNMCTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		DP	///				INIT		///	MEMCC	///		SR	CS	CE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x020															

Table 7-16. DYNMCTRL Fields

BITS	NAME	FUNCTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	DP	**Low-Power SDRAM Deep-Sleep Mode** 1 = Enter deep power down mode 0 = Normal operation
12:9	///	**Reserved** Reading returns 0. Write the reset value.
8:7	INIT	**SDRAM Initialization** 00 = issue SDRAM NORMAL operation command 01 = issue SDRAM MODE command 10 = issue SDRAM PALL (precharge all) command 11 = issue SDRAM NOP (no operation) command
6	///	**Reserved** Reading returns 0. Write the reset value.
5	MEMCC	**Memory Clock Control** Disabling SDCLK can be performed if there are no SDRAM memory transactions. When enabled, this field can be used in conjunction with the dynamic memory clock control (CS) field. 1 = SDCLK disabled 0 = SDCLK enabled
4:3	///	**Reserved** Reading returns 0. Write the reset value.
2	SR	**Self-Refresh Request** Software can command the EMC into self-refresh by writing a 1 to this bit. Writing 0 to this bit returns the memory controller to normal mode. The Self-refresh Acknowledge bit (SA) in the Status register must be polled to determine the current operating mode of the memory controller. Note that static memory may be accessed normally while dynamic memory is in self-refresh mode. 1 = Enter Self-Refresh Mode 0 = Normal Mode
1	CS	**Dynamic Memory Clock Select** When clock control is 0 the output clock SDCLK is stopped when there are no SDRAM transactions. The clock is also stopped during self-refresh mode. 1 = SDCLK runs continuously 0 = SDCLK stops when all SDRAMs are idle and during self-refresh mode.
0	CE	**Dynamic Memory Clock Enable** Clock enable must be HIGH during SDRAM initialization 1 = All clock enables are driven HIGH continuously 0 = Clock enable signal of idle devices are deasserted to save power

7.5.2.4 Dynamic Refresh Register (DYNMREF)

This register configures dynamic memory operation.

This register should only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable. However, these control bits can, if necessary, be altered during normal operation.

Writing a value of 0x000 disables refreshing. Programming any other value, 'n', results in a delay between refresh cycles of 16 × n. For example, for the refresh period of 16 µs, and an HCLK frequency of 50 MHz, the following value must be programmed into this register:

$(16 \times 10^{-6} \times 50 \times 10^{6}) \div 16 = 50$, or 0x032

Table 7-17. DYNMREF Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///					REFRESH										
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x024															

Table 7-18. DYNMREF Fields

BITS	NAME	FUNCTION
31:11	///	**Reserved** Reading returns 0. Write the reset value.
10:0	REFRESH	**Refresh Timer (×16)** For REFRESH = 0x000: Refresh disabled For REFRESH between 0x001 and 0xFFF: HCLK cycles between SDRAM refresh cycles = 16 × (REFRESH)

7.5.2.5 Dynamic Memory Read Configuration Register (DYNMRCON)

This register allows configuration of the dynamic memory Read strategy. This register should only be modified during initialization. This register provides the Read strategy for all four dynamic memory Chip Select signals. The DYNMRCON resets to 0x00, which is invalid. Therefore, this register must be programmed to 0x01 during initialization if the SDRAM controller is used.

Table 7-19. DYNMRCON Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														RDS	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFF1000 + 0x028															

Table 7-20. DYNMRCON Fields

BITS	NAME	FUNCTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	RDS	**Read Data Strategy** This field selects the Read strategy. 00 = Reserved 01 = Command Delayed Strategy (Clock Out not delayed; command delayed) 10 = Reserved 11 = Reserved

7.5.2.6 Dynamic Precharge Command Period Register (PRECHARGE)

The Dynamic Memory Precharge Command Period Register programs the Precharge Command Period, tRP. This value is normally found in SDRAM data sheets as tRP.

This register must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-21. PRECHARGE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tRP			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x030															

Table 7-22. PRECHARGE Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tRP	**Precharge Command Period** Period = (tRP + 1) External Memory Clock periods

7.5.2.7 Dynamic Memory Active to Precharge Command Period Register (DYNM2PRE)

The Dynamic Memory Active to Precharge Command Period Register enables programming the Active to Precharge Command Period, tRAS. This value is normally found in SDRAM data sheets as t_{RAS}.

This register should only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-23. DYNM2PRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tRAS			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x034															

Table 7-24. DYNM2PRE Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tRAS	**Active to Precharge Command Period** Period = (tRAS + 1) External Memory Clock periods

7.5.2.8 Dynamic Memory Self-Refresh Exit Time Register (REFEXIT)

The Dynamic Memory Self-Refresh Exit Time Register enables programming the Self-refresh Exit Time, tSREX. This value is normally found in SDRAM data sheets as t_{SREX}. This register is used as the self-refresh exit time for all chip selects. Therefore, it must be programmed with the longest exit time period required of all the chip selects.

This register should only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-25. REFEXIT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tSREX			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x038															

Table 7-26. REFEXIT Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tSREX	**Self-Refresh Exit Time** Exit Time = (tSREX + 1) External Memory Clock periods

7.5.2.9 Dynamic Memory Last Data Out to Active Time Register (DOACTIVE)

The Dynamic Memory Last Data Out to Active Time Register enables programming the Last-data-out to Active Command Time, tAPR. This value is normally found in SDRAM data sheets as tAPR.

This register must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-27. DOACTIVE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tAPR			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x03C															

Table 7-28. DOACTIVE Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tAPR	**Last-Data-Out to Active Command Time** Time = (tAPR + 1) External Memory Clock periods

7.5.2.10 Dynamic Memory Data-In to Active Time Register (DIACTIVE)

The Dynamic Memory Data-In to Active Time Register enables programming the Data-in to Active Command time, tDAL. This value is normally found in SDRAM data sheets as tDAL, or tAPW.

This register must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-29. DIACTIVE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tDAL			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x040															

Table 7-30. DIACTIVE Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tDAL	**Data-In to Active Command Time** Time = (tDAL) External Memory Clock periods

7.5.2.11 Dynamic Memory Write Recovery Time Register (DWRT)

The Dynamic Memory Write Recovery Time Register enables programming the Write Recovery Time, tWR. This value is normally found in SDRAM data sheets as tWR, tDPL, tRWL, or tRDL.

This register must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-31. DWRT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	R0	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tWR			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x044															

Table 7-32. DWRT Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tWR	**Write Recovery Time** Write Recovery Time = (tWR + 1) External Memory Clock periods

7.5.2.12 Dynamic Memory Active to Active Command Period Register (DYNACTCMD)

The Dynamic Memory Active to Active Command Period Register enables programming the Active to Active Command Period, tRC. This value is normally found in SDRAM data sheets as tRC.

Note that tRC is programmable only for memory accesses in the same bank. For accesses between banks, tRC is fixed at 432 nS and the value in this register is ignored.

This register must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-33. DYNACTCMD Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											tRC				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x048															

Table 7-34. DYNACTCMD Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	tRC	**Active to Active Command Period** Period = (tRC + 1) External Memory Clock periods

7.5.2.13 Dynamic Memory Auto-Refresh Period, and Auto-Refresh to Active Command Period Register (DYNAUTO)

The Dynamic Memory Auto-Refresh Period, and Auto-Refresh to Active Command Period Register enables programming the Auto-refresh Period, and Auto-refresh to Active Command Period, tRFC. This value is normally found in SDRAM data sheets as tRFC or sometimes as tRC.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-35. DYNAUTO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											tRFC				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x04C															

Table 7-36. DYNAUTO Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	tRFC	**Auto-Refresh Period and Auto-Refresh to Active Command Period** 0x00 to 0x1F = (n + 1) External Memory Clock periods

7.5.2.14 Dynamic Memory Exit Self-Refresh to Active Command Time Register (DYNREFEXIT)

The Dynamic Memory Exit Self-Refresh to Active Command Time Register selects the Exit Self-refresh to Active Command Time, tXSR. This value is normally found in SDRAM data sheets as tXSR.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-37. DYNREFEXIT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											tXSR				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x050															

Table 7-38. DYNREFEXIT Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	tXSR	**Exit Self-Refresh to Active Command Time** Time = (tXSR + 1) External Memory Clock periods

7.5.2.15 Dynamic Memory Active Bank A to Active Bank B Time Register (DYNACTIVEAB)

The Dynamic Memory Active Bank A to Active Bank B Time Register programs the active bank A to active bank B latency, tRRD. This value is normally found in SDRAM data sheets as tRRD.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-39. DYNACTIVEAB Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tRRD			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x054															

Table 7-40. DYNACTIVEAB Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tRRD	**Active Bank A to Active Bank B Latency** Latency = (tRRD + 1) External Memory Clock periods

7.5.2.16 Dynamic Memory Load Mode Register to Active Command Time Register (DYNAMICTMRD)

The Dynamic Memory Load Mode Register to Active Command Time Register specifies the Load Mode Register to Active Command Time, tMRD. This value is normally found in SDRAM data sheets as tMRD, or tRSA.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-41. DYNAMICTMRD Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												tMRD			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x058															

Table 7-42. DYNAMICTMRD Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	tMRD	**Load Mode Register to Active Command Time** Time = (tMRD + 1) External Memory Clock periods

7.5.2.17 Static Memory Extended Wait Register (WAIT)

The Static Memory Extended Wait Register is used to time long static memory read and write transfers (longer than can be supported by the SWAITRD or SWAITWR registers) when the EW bit of the SCONFIG register is enabled. There is only a single WAIT register.

These registers should only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable. However, if necessary, these control bits can be altered during normal operation.

Table 7-43. WAIT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						WAIT									
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x080															

Table 7-44. WAIT Fields

BITS	NAME	FUNCTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9:0	WAIT	**External Wait Time Out** Wait Time Out = (WAIT+1) x 16 HCLK clock periods

7.5.2.18 Dynamic Configuration Register for nDCS0 and nDCS1 (DYNCFGx)

The Dynamic Configuration Register specifies the configuration information for the relevant dynamic memory Chip Select. These registers are normally only modified during system initialization.

Table 7-45. DYNCFGx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///											P	B	///		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///	AM	///	AM						///		MD		///		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x100 for DYNCFG0 0xFFFF1000 + 0x120 for DYNCFG1															

Table 7-46. DYNCFGx Fields

BITS	NAME	FUNCTION
31:21	///	**Reserved** Reading returns 0. Write the reset value.
20	P	**Write Protect** 1 = Write protected 0 = Not Write protected
19	B	**Read and Write Buffer Enable** The Buffer Enable bit must be set to 1 for proper SDRAM interface operation. The buffers must be disabled during SDRAM and SyncFlash initialization. They must also be disabled when performing SyncFlash commands. The buffers must be enabled during normal operation. **NOTE:** The buffers must be disabled during SDRAM and SyncFlash initialization, and when performing SyncFlash commands. The buffers must be enabled during normal operation. 1 = Read and Write Buffers enabled for accesses to this Chip Select 0 = Read and Write Buffers disabled for accesses to this Chip Select
18:15	///	**Reserved** Read undefined, must write as zeros.
14	AM	**Address Mapping** See Table 7-47
13	///	**Reserved** Read undefined, must write as zeros.
12:7	AM	**Address Mapping** See Table 7-47
6:5	///	**Reserved** Read undefined, must write as zeros.
4:3	MD	**Memory Device** 00 = SDRAM 01 = low-power SDRAM 10 = Micron SyncFlash 11 = reserved
2:0	///	**Reserved** Read undefined, must write as zeros.

Table 7-47. Address Mapping

DYNCFGx BITS				DESCRIPTION
[14]	[12]	[11:9]	[8:7]	
16-BIT EXTERNAL BUS HIGH-PERFORMANCE ADDRESS MAPPING (ROW, BANK, COLUMN)				
0	0	000	00	16Mb (2M × 8), 2 banks, row length = 11, column length = 9
0	0	000	01	16Mb (1M × 16), 2 banks, row length = 11, column length = 8
0	0	001	00	64Mb (8M × 8), 4 banks, row length = 12, column length = 9
0	0	001	01	64Mb (4M × 16), 4 banks, row length = 12, column length = 8
0	0	010	00	128Mb (16M × 8), 4 banks, row length = 12, column length = 10
0	0	010	01	128Mb (8M × 16), 4 banks, row length = 12, column length = 9
0	0	011	00	256Mb (32M × 8), 4 banks, row length = 13, column length = 10
0	0	011	01	256Mb (16M × 16), 4 banks, row length = 13, column length = 9
0	0	100	00	512Mb (64M × 8), 4 banks, row length = 13, column length = 11
0	0	100	01	512Mb (32M × 16), 4 banks, row length = 13, column length = 10
16-BIT EXTERNAL BUS LOW POWER SDRAM ADDRESS MAPPING (BANK, ROW, COLUMN)				
0	1	000	00	16Mb (2M × 8), 2 banks, row length = 11, column length = 9
0	1	000	01	16Mb (1M × 16), 2 banks, row length = 11, column length = 8
0	1	001	00	64Mb (8M × 8), 4 banks, row length = 12, column length = 9
0	1	001	01	64Mb (4M × 16), 4 banks, row length = 12, column length = 8
0	1	010	00	128Mb (16M × 8), 4 banks, row length = 12, column length = 10
0	1	010	01	128Mb (8M × 16), 4 banks, row length = 12, column length = 9
0	1	011	00	256Mb (32M × 8), 4 banks, row length = 13, column length = 10
0	1	011	01	256Mb (16M × 16), 4 banks, row length = 13, column length = 9
0	1	100	00	512Mb (64M × 8), 4 banks, row length = 13, column length = 11
0	1	100	01	512Mb (32M × 16), 4 banks, row length = 13, column length = 10
32-BIT EXTERNAL BUS HIGH PERFORMANCE ADDRESS MAPPING (ROW, BANK, COLUMN)				
1	0	000	00	16Mb (2M × 8), 2 banks, row length = 11, column length = 9
1	0	000	01	16Mb (1M × 16), 2 banks, row length = 11, column length = 8
1	0	001	00	64Mb (8M × 8), 4 banks, row length = 12, column length = 9
1	0	001	01	64Mb (4M × 16), 4 banks, row length = 12, column length = 8
1	0	001	11	64Mb (2M × 32), 4 banks, row length = 11, column length = 8
1	0	010	00	128Mb (16M × 8), 4 banks, row length = 12, column length = 10
1	0	010	01	128Mb (8M × 16), 4 banks, row length = 12, column length = 9
1	0	010	10	128Mb (4M × 32), 4 banks, row length = 12, column length = 8
1	0	011	00	256Mb (32M × 8), 4 banks, row length = 13, column length = 10
1	0	011	01	256Mb (16M × 16), 4 banks, row length = 13, column length = 9
1	0	011	10	256Mb (8M × 32), 4 banks, row length = 13, column length = 8
1	0	100	00	512Mb (64M × 8), 4 banks, row length = 13, column length = 11
1	0	100	01	512Mb (32M × 16), 4 banks, row length = 13, column length = 10

Table 7-47. Address Mapping (Cont'd)

DYNCFGx BITS				DESCRIPTION
[14]	[12]	[11:9]	[8:7]	
32-BIT EXTERNAL BUS LOW POWER SDRAM ADDRESS MAPPING (BANK, ROW, COLUMN)				
1	1	000	00	16Mb (2M × 8), 2 banks, row length = 11, column length = 9
1	1	000	01	16Mb (1M × 16), 2 banks, row length = 11, column length = 8
1	1	001	00	64Mb (8M × 8), 4 banks, row length = 12, column length = 9
1	1	001	01	64Mb (4M × 16), 4 banks, row length = 12, column length = 8
1	1	001	10	64Mb (2M × 32), 4 banks, row length = 11, column length = 8
1	1	010	00	128Mb (16M × 8), 4 banks, row length = 12, column length = 10
1	1	010	01	128Mb (8M × 16), 4 banks, row length = 12, column length = 9
1	1	010	10	128Mb (4M × 32), 4 banks, row length = 12, column length = 8
1	1	011	00	256Mb (32M × 8), 4 banks, row length = 13, column length = 10
1	1	011	01	256Mb (16M × 16), 4 banks, row length = 13, column length = 9
1	1	011	10	256Mb (8M × 32), 4 banks, row length = 13, column length = 8
1	1	100	00	512Mb (64M × 8), 4 banks, row length = 13, column length = 11
1	1	100	01	512Mb (32M × 16), 4 banks, row length = 13, column length = 10

NOTES:

1. A Chip Select can be connected to a single memory device, in this case the Chip Select data bus width is the same as the device width. Alternatively the Chip Select can be connected to a number of external devices. In this case the Chip Select data bus width is the sum of the memory device data bus widths. For example, for a Chip Select connected to:
   - two x 16-bit wide memory devices, choose a 32-bit wide address mapping.
   - a 32-bit wide memory device, choose a 32-bit wide address mapping.
   - a 16-bit wide memory device, choose a 16-bit wide address mapping.
   - four x 8-bit wide memory devices, choose a 32-bit wide address mapping.
   - two x 8-bit wide memory devices, choose a 16-bit wide address mapping.
2. The SDRAM column and row width and number of banks are computed automatically from the address mapping.
3. The SDRAM section of the memory controller does a constant burst of 8 when configured to access 16 bit memory devices. Similarly, it does burst of 4 when configured to access 32-bit memory devices.

7.5.2.19 Dynamic Memory RAS and CAS Delay Register for nDCS0 and nDCS1 (DYNRASCASx)

The Dynamic Memory RAS and CAS Delay Register selects the RAS and CAS latencies for the relevant dynamic memory. Note that the same value must be programmed into the device's Mode register.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-48. DYNRASCASx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						CAS		///						RAS	
RESET	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
TYPE	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x104 for DYNRASCAS0 0xFFFF1000 + 0x124 for DYNRASCAS1															

Table 7-49. DYNRASCASx Fields

BITS	NAME	FUNCTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9:8	CAS*	**CAS Latency** 00 = reserved 01 = one clock cycle 10 = two clock cycles 11 = three clock cycles
7:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	RAS*	**RAS Latency** 00 = reserved 01 = one clock cycle 10 = two clock cycles 11 = three clock cycles

NOTE: *The RAS to CAS latency (RAS) and CAS latency (CAS) are both defined in HCLK clock cycles.

7.5.2.20 Static Memory Configuration Register (SCONFIGx)

The Static Memory Configuration Registers are used to configure the static memory configuration.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-50. SCONFIGx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///											P	B	///		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///							EW	BLS[1]	PC	///		PM	///	MW[2]	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x200 for SCONFIG0 0xFFFF1000 + 0x220 for SCONFIG1 0xFFFF1000 + 0x240 for SCONFIG2 0xFFFF1000 + 0x260 for SCONFIG3															

NOTES:
1. The reset value for nCS1 changes depending on what byte lane boot state is selected. If nBLE0 isn't driven externally at reset, reset value will be 0. Other chip selects always have a reset value of 0.
2. The reset value for nCS1 changes accordingly to what boot mode width is selected. If nBLE2 and nBLE1 aren't driven externally at reset, system will boot in 16-bit mode, with MW resetting to 0b01. Other chip selects always have a reset value of 0b00.

Table 7-51. SCONFIGx Fields

BITS	NAME	FUNCTION
31:21	///	**Reserved** Reading returns 0. Write the reset value.
20	P	**Write Protect** 1 = Write protected 0 = Not Write protected
19	B	**Buffer Enable** 1 = Read and write buffer enabled 0 = Read and write buffer disabled
18:9	///	**Reserved** Reading returns 0. Write the reset value.

Table 7-51. SCONFIGx Fields (Cont'd)

BITS	NAME	FUNCTION
8	EW	**Extended Wait** Extended wait uses the WAIT register to time both the read and write transfers rather than the SWAITRD and SWAITWR registers. This enables much longer transactions. 1 = Extended wait enabled 0 = Extended wait disabled
7	BLS	**Byte Lane State** This bit allows connection of different storage widths. For byte-wide static memories, the nBLE[3:0] signal is usually connected to nWE. In this case for reads all the nBLE[3:0] bits must be HIGH. This means that the BLS bit must be 0. 16-bit wide static memory devices usually have the nBLE[3:0] signals connected to the nUB and nLB (upper byte and lower byte) signals in the static memory. In this case a write to a particular byte must assert LOW the appropriate nUB or nLB signal. For reads, all the nUB and nLB signals must be asserted LOW so that the bus is driven. In this case the BLS bit must be HIGH. **IMPORTANT**: When accessing NAND Flash, this bit must be programmed to 1 by the Boot ROM code for proper operation. Writes 1 = The active bits in nBLE[3:0] are LOW 0 = The active bits in nBLE[3:0] are HIGH Reads 1 = The active bits in nBLE[3:0] are LOW 0 = All the bits in nBLE[3:0] are HIGH
6	PC	**Chip Select Polarity** 1 = active HIGH Chip Select 0 = active LOW Chip Select
5:4	///	**Reserved** Read undefined, must write as zeros.
3	PM	**Page Mode** In page mode the memory controller can burst up to four external accesses. Therefore devices with asynchronous page mode burst of four or higher are supported. Asynchronous page mode burst two devices are not supported and must be accessed normally. 1 = asynchronous page mode enabled 0 = disabled
2	///	**Reserved** Read undefined, must write as zeros.
1:0	MW	**Memory Width** Set this to the memory width of the bank, not the devices. For example, if two 8-bit devices are configured as a 16-bit memory bank, choose 01. 00 = 8-bit 01 = 16-bit 10 = 32-bit 11 = reserved

7.5.2.21 Static Memory Write Enable Delay Registers (SWAITWENx)

The Static Memory Write Enable Delay Registers allow programming a delay between Address Valid and the assertion of nWE (nBLEx). See Section 7.2.4.1.2 for a complete description of programming these registers.

Wait states behave slightly differently for Write transactions than for Reads. Instead of the length of the Write cycle (tWC) being the sum of the valued programmed into the SWAITWENx and SWAITWRx registers, it has the following relationship (with 'A' being SWAITWENx, 'B' being SWAITWRx, and 'C' being the address hold time):

tWC = tA0 + tB0 + tB1 ... tBn + C, where the length of each term is one HCLK period, and 'n' is the value programmed in the respective register.

The minimum value for the equation is tWC = tA0 + tB0 + C, and is therefore the zero wait state timing.

Note that additional 'A' terms (delaying the assertion of nWE or nBLEx) do not add wait states. This also requires that: SWAITWRx ≥ SWAITWENx.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-52. SWAITWENx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												WAITWEN			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x204 for SWAITWEN0 0xFFFF1000 + 0x224 for SWAITWEN1 0xFFFF1000 + 0x244 for SWAITWEN2 0xFFFF1000 + 0x264 for SWAITWEN3															

Table 7-53. SWAITWENx Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	WAITWEN	**Wait Write Enable** Program the delay from assertion of nCSx to the assertion of nWE (nBLEx). See Section 7.2.4.1.2 for timing examples.

7.5.2.22 Static Memory Output Enable Delay Registers (SWAITOENx)

The Static Memory Output Enable Delay Registers enable programming the delay from the Valid Address to nOE assertion. See Section 7.2.4.1.1 for a complete description of programming these registers.

The total Read cycle time is the total time that the address is valid. During this time, several parameters are programmable. In the following equations, 'D' represents the SWAITOENx register, 'E' represents the SWAITRDx register, and 'C' represents the address hold time. In general, Read wait states can be derived from the following equation:

tRC (Read cycle time) = tD1 + tD2 + ... tDn + tE0 + tE1 + ... tEn + C, where the length of each term is one HCLK period, and 'n' is the value programmed in the respective register.

The minimum value for the equation is tRC = tE0 + C, and is therefore the zero wait state timing.

Thus, Read wait states can be programmed using the appropriate mix of nOE extension (programmed in SWAITRDx) with nOE assertion delay (programmed in SWAITOENx).

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-54. SWAITOENx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												WAITOEN			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x208 for SWAITOEN0 0xFFFF1000 + 0x228 for SWAITOEN1 0xFFFF1000 + 0x248 for SWAITOEN2 0xFFFF1000 + 0x268 for SWAITOEN3															

Table 7-55. SWAITOENx Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	WAITOEN	**Wait Output Enable** Delay from Valid Address to assertion of nOE. See Section 7.2.4.1.1 for timing examples.

7.5.2.23 Static Memory Read Delay Registers (SWAITRDx)

The Static Memory Read Delay Registers enable programming Read cycle wait states. A complete description of programming this register appears in Section 7.2.4.1.1.

The total Read cycle time is the total time that the address is valid. During this time, several parameters are programmable. In the following equations, 'D' represents the SWAITOENx register, 'E' represents the SWAITRDx register, and 'C' represents the address hold time. In general, Read wait states can be derived from the following equation:

tRC (Read cycle time) = tD1 + tD2 + ... tDn + tE0 + tE1 + ... tEn + C, where the length of each term is one HCLK period, and 'n' is the value programmed in the respective register.

The minimum value for the equation is tRC = tE0 + C, and is therefore the zero wait state timing.

Thus, Read wait states can be programmed using the appropriate mix of nOE extension (programmed in SWAITRDx) with nOE assertion delay (programmed in SWAITOENx).

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-56. SWAITRDx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											WAITRD				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x20C for SWAITRD0 0xFFFF1000 + 0x22C for SWAITRD1 0xFFFF1000 + 0x24C for SWAITRD2 0xFFFF1000 + 0x26C for SWAITRD3															

Table 7-57. SWAITRDx Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	WAITRD	**Read Access Delay** Non-page mode Read or Asynchronous Page Mode Read, delay first read only: Delay = (n + 1) HCLK cycles

7.5.2.24 Static Memory Page Mode Read Delay Registers (SWAITPAGEx)

The Static Memory Page Mode Read Delay Registers enable programming the delay for Asynchronous Page Mode sequential accesses.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-58. SWAITPAGEx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											WAITPAGE				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x210 for SWAITPAGE0 0xFFFF1000 + 0x230 for SWAITPAGE1 0xFFFF1000 + 0x250 for SWAITPAGE2 0xFFFF1000 + 0x270 for SWAITPAGE3															

Table 7-59. SWAITPAGEx Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	WAITPAGE	**Asynchronous Page Mode Delay** Number of Wait States for Asynchronous Page Mode Read accesses after the first Read: Asynchronous Page Mode Delay = (WAITPAGE + 1) HCLK cycles

7.5.2.25 Static Memory Write Delay Registers (SWAITWRx)

The Static Memory Write Delay Registers enable programming the number of Write wait states. See Section 7.2.4.1.2 for a complete description of programming these registers.

Wait states behave slightly differently for Write transactions than for Reads. Instead of the length of the Write cycle (tWC) being the sum of the valued programmed into the SWAITWENx and SWAITWRx registers, it has the following relationship (with 'A' being SWAITWENx, 'B' being SWAITWRx, and 'C' being the address hold time):

tWC = tA0 + tB0 + tB1 ... tBn + C, where the length of each term is one HCLK period, and 'n' isthe value programmed in the respective register.

The minimum value for the equation is tWC = tA0 + tB0 + C, and is therefore the zero wait state timing.

Note that additional 'A' terms (delaying the assertion of nWE or nBLEx) do not add wait states. This also requires that: SWAITWRx ≥ SWAITWENx

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

Table 7-60. SWAITWRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											WAITWR				
RESET	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x214 for SWAITWR0 0xFFFF1000 + 0x234 for SWAITWR1 0xFFFF1000 + 0x254 for SWAITWR2 0xFFFF1000 + 0x274 for SWAITWR3															

Table 7-61. SWAITWRx Fields

BITS	NAME	FUNCTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4:0	WAITWR	**Write Wait States** Program with the number of required Write wait states. See Section 7.2.4.1.2 for timing examples.

7.5.2.26 Static Memory Turn Around Delay Registers (STURNx)

The Static Memory Turn Around Delay Registers enable programming the number of bus turnaround cycles.

These registers must only be modified during system initialization, or when there are no current or outstanding transactions. Software can ensure that there are no current or outstanding transactions by waiting until the memory controller is idle, then entering Low-Power Mode (CONTROL:MODE = 1), or Disable Mode (CONTROL:ENABLE = 0). When in these two modes, external memory access is not allowed, ensuring that changing parameters will not corrupt external data. Low-Power Mode automatically refreshes SDRAM; Disable Mode requires commanding the SDRAM to Self Refresh (DYNMCTRL:SR = 1) prior to entering Disable.

To prevent bus contention on the external memory data bus, the BTC field controls the number of bus turnaround cycles added between static memory read and write accesses. The BTC field also controls the number of turnaround cycles between static memory and dynamic memory accesses.

Table 7-62. STURNx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												BTC			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF1000 + 0x218 for STURN0 0xFFFF1000 + 0x238 for STURN1 0xFFFF1000 + 0x258 for STURN2 0xFFFF1000 + 0x278 for STURN3															

Table 7-63. STURNx Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	BTC	**Bus Turnaround Cycles** SRAM Wait State time for Write accesses after the first Read: Bus Turnaround Cycles = (BTC + 1) HCLK cycles

Chapter 8
General Purpose Input/Output

The LH79524 and LH79525 have a robust set of General Purpose Input/Output (GPIO) pins that can be used for any required input or output function. The LH79524 has 14 ports, providing 108 pins of GPIO. The LH79525 has 11 ports, with 86 individual pins. All descriptions, unless noted, apply to both the LH79524 and LH79525.

8.1 Theory of Operation

The GPIO pins are multiplexed with other functions. Some of the GPIO ports have Schmitt triggers, and some have pull-up or pull-down resistors built in (See Table 8-2). Table 8-1 summarizes the configuration of each port. Note that Port J is input only, and Port M is output only. At reset, all ports except Ports J and M are configured for input. Ports J and M are not configured as GPIO at reset.

Table 8-1. GPIO Ports

PORT	LH79524 GPIO	LH79525 GPIO
A	8 Input/Output Pins	8 Input/Output Pins
B	8 Input/Output Pins	8 Input/Output Pins
C	8 Input/Output Pins	8 Input/Output Pins
D	8 Input/Output Pins	8 Input/Output Pins
E	8 Input/Output Pins	8 Input/Output Pins
F	8 Input/Output Pins	8 Input/Output Pins
G	8 Input/Output Pins	8 Input/Output Pins
H	8 Input/Output Pins	8 Input/Output Pins
I	8 Input/Output Pins	8 Input/Output Pins
J	8 Input Pins	8 Input Pins
K	8 Input/Output Pins	Not Available
L	8 Input/Output Pins	Not Available
M	8 Output Pins	6 Output Pins
N	4 Input/Output Pins	Not Available

8.1.1 Port Configuration

Port pins can be configured individually as inputs or outputs (except, of course, Ports J and M). The direction of each pin is programmed using the Data Direction Registers (P1DDRx and P2DDRx). Data for output pins is written to the port Data Register (P1DRx and P2DRx). Input data is read from the same registers.

8.1.1.1 Multiplexing

All of the GPIO pins are multiplexed with one or more other functions. Selecting certain on-chip peripherals removes the GPIO function from that pin. Configuration of these multiplexed pins is described in the I/O Configuration chapter.

Table 8-2 and Table 8-3 show the multiplexed functions of each GPIO pin. In the case of Port J and Port M, the reset function is not GPIO, hence those functions are listed in the 'Multiplexed Function' column.

Table 8-2. LH79524 GPIO Multiplexing

CABGA NO.	AT RESET	MULTIPLEXED FUNCTION(S)
N1	PA0	INT2/UARTRX2/UARTIRRX2
M2	PA1	INT3/UARTTX2/UARTIRTX2
L3	PA2	CTCAP0A/CTCMP0A
M1	PA3	CTCAP0B/CTCMP0B
L2	PA4	CTCAP1A/CTCMP1A
L1	PA5	CTCAP1B/CTCMP1B
K3	PA6	CTCAP2A/CTCMP2A/SDA
K2	PA7	CTCAP2B/CTCMP2B/SCL
R2	PB0	nDACK/nUARTCTS0
R1	PB1	DREQ/nUARTRTS0
P2	PB2	SSPFRM/I2SWS
N3	PB3	SSPCLK/I2SCLK
M4	PB4	SSPRX/I2SRXD/UARTRX1/UARTIRRX1
P1	PB5	SSPTX/I2STXD/UARTTX1/UARTIRTX1
N2	PB6	INT0/UARTRX0/UARTIRRX0
M3	PB7	INT1/UARTTX0/UARTIRTX0
N7	PC0	A16
R6	PC1	A17
T5	PC2	A18
P6	PC3	A19
R5	PC4	A20
T4	PC5	A21
P5	PC6	A22/nFWE
R4	PC7	A23/nFRE
P15	PD0	D8
P14	PD1	D9
N13	PD2	D10
T15	PD3	D11
N12	PD4	D12
T14	PD5	D13
P12	PD6	D14
T13	PD7	D15

Table 8-2. LH79524 GPIO Multiplexing (Cont'd)

CABGA NO.	AT RESET	MULTIPLEXED FUNCTION(S)
B12	PE0	LCDLP/LCDHRLP
D11	PE1	LCDDCLK
B13	PE2	LCDPS
C13	PE3	LCDCLS
D12	PE4	LCDDSPLEN/LCDREV
B16	PE5	LCDVDDEN
B15	PE6	LCDVEEN/LCDMOD
D14	PE7	nWAIT/nDEOT
A8	PF0	LCDVD6
A9	PF1	LCDVD7
B9	PF2	LCDVD8
C9	PF3	LCDVD9
B10	PF4	LCDVD10
A11	PF5	LCDVD11
B11	PF6	LCDEN/LCDSPL
A12	PF7	LCDFP/LCDSPS
A5	PG0	ETHERTXEN
B6	PG1	ETHERTXCLK
A6	PG2	LCDVD0
C7	PG3	LCDVD1
B7	PG4	LCDVD2
A7	PG5	LCDVD3
C8	PG6	LCDVD4
B8	PG7	LCDVD5
C4	PH0	ETHERRX3
A3	PH1	ETHERRXDV
B4	PH2	ETHERRXCLK
C5	PH3	ETHERTXER
D6	PH4	ETHERTX0
A4	PH5	ETHERTX1
B5	PH6	ETHERTX2
C6	PH7	ETHERTX3
D3	PI0	ETHERMDC
B1	PI1	ETHERMDIO
B2	PI2	ETHERCOL
D4	PI3	ETHERCRS
C3	PI4	ETHERRXER
A1	PI5	ETHERRX0
A2	PI6	ETHERRX1
B3	PI7	ETHERRX2
R16	PK0	D16

Table 8-2. LH79524 GPIO Multiplexing (Cont'd)

CABGA NO.	AT RESET	MULTIPLEXED FUNCTION(S)
M12	PK1	D17
T16	PK2	D18
R15	PK3	D19
P13	PK4	D20
R14	PK5	D21
R13	PK6	D22
N11	PK7	D23
C1	PL0	LCDVD14
C2	PL1	LCDVD15
A10	PL2	LCDVD12
C10	PL3	LCDVD13
C12	PL4	D28
A14	PL5	D29
B14	PL6	D30
C14	PL7	D31
C11	PN0	D26
A13	PN1	D27
R12	PN2	D24
P11	PN3	D25
H2	AN3/LR/Y-	PJ0
H3	AN4/WIPER	PJ1
G1	AN9	PJ2
G2	AN2/LL/Y+	PJ3
G3	AN8	PJ4
F1	AN5	PJ5/INT5
E1	AN7	PJ6/INT6
F3	AN6	PJ7/INT7
L16	nCS0	PM0
L15	nCS1	PM1
M16	nCS2	PM2
L14	nCS3	PM3
J15	nBLE0	PM4
J14	nBLE1	PM5
K16	nBLE2	PM6
K15	nBLE3	PM7

Table 8-3. LH79525 GPIO Multiplexing

PIN	AT RESET	MULTIPLEXED FUNCTION
36	PA0	INT2/UARTRX2/UARTIRRX2
35	PA1	INT3/UARTTX2/UARTIRTX2
34	PA2	CTCAP0A/CTCMP0A
32	PA3	CTCAP0B/CTCMP0B
31	PA4	CTCAP1A/CTCMP1A
30	PA5	CTCAP1B/CTCMP1B
29	PA6	CTCAP2A/CTCMP2A/SDA
28	PA7	CTCAP2B/CTCMP2B/SCL
44	PB0	nDACK/nUARTCTS0
43	PB1	DREQ/nUARTRTS0
42	PB2	SSPFRM/I2SWS
41	PB3	SSPCLK/I2SCLK
40	PB4	SSPRX/I2SRXD/UARTRX1/UARTIRRX1
39	PB5	SSPTX/I2STXD/UARTTX1/UARTIRTX1
38	PB6	INT0/UARTRX0/UARTIRRX0
37	PB7	INT1/UARTTX0/UARTIRTX0
60	PC0	A16
59	PC1	A17
58	PC2	A18
56	PC3	A19
55	PC4	A20
54	PC5	A21
53	PC6	A22/nFWE
52	PC7	A23/nFRE
90	PD0	D8
89	PD1	D9
88	PD2	D10
87	PD3	D11
85	PD4	D12
84	PD5	D13
83	PD6	D14
82	PD7	D15
141	PE0	LCDLP/LCDHRLP
139	PE1	LCDDCLK
138	PE2	LCDPS
137	PE3	LCDCLS
136	PE4	LCDDSPLEN/LCDREV
134	PE5	LCDVDDEN
133	PE6	LCDVEEN/LCDMOD
120	PE7	nWAIT/nDEOT
153	PF0	LCDVD6
151	PF1	LCDVD7
149	PF2	LCDVD8
147	PF3	LCDVD9

Table 8-3. LH79525 GPIO Multiplexing (Cont'd)

PIN	AT RESET	MULTIPLEXED FUNCTION
146	PF4	LCDVD10
145	PF5	LCDVD11
143	PF6	LCDEN/LCDSPL
142	PF7	LCDFP/LCDSPS
162	PG0	ETHERTXEN
161	PG1	ETHERTXCLK
159	PG2	LCDVD0
158	PG3	LCDVD1
157	PG4	LCDVD2
156	PG5	LCDVD3
155	PG6	LCDVD4
154	PG7	LCDVD5
171	PH0	ETHERRX3
170	PH1	ETHERRXDV
169	PH2	ETHERRXCLK
167	PH3	ETHERTXER
166	PH4	ETHERTX0
165	PH5	ETHERTX1
164	PH6	ETHERTX2
163	PH7	ETHERTX3
4	PI0	ETHERMDC
2	PI1	ETHERMDIO
1	PI2	ETHERCOL
176	PI3	ETHERCRS
175	PI4	ETHERRXER
174	PI5	ETHERRX0
173	PI6	ETHERRX1
172	PI7	ETHERRX2
20	AN3/LR/Y-	PJ0
19	AN4/WIPER	PJ1
18	AN9	PJ2
17	AN2/LL/Y+	PJ3
16	AN8	PJ4
15	AN5	PJ5/INT5
13	AN7	PJ6/INT6
12	AN6	PJ7/INT7
104	nCS0	PM0
103	nCS1	PM1
102	nCS2	PM2
100	nCS3	PM3
110	nBLE0	PM4
109	nBLE1	PM5

8.2 Register Reference

This section describes the location and programming of the GPIO registers. Registers are denoted with an 'x' that is replaced with the port letter of the register. For example, the Port A Data Direction Register is P1DDRA.

8.2.1 Memory Map

The base addresses for the GPIO registers are:

Ports A and B: 0xFFFDF000

Ports C and D: 0xFFFDE000

Ports E and F: 0xFFFDD000

Ports G and H: 0xFFFDC000

Ports I and J: 0xFFFDB000

Ports K and L: 0xFFFDA000

Port M and N: 0xFFFD9000

As Port J is an input-only port, the location at base 0xFFFDB000 with an offset of 0x0C (which would be the space for the Port J Data Direction Register) is reserved. However, output-only Port M does have a Data Direction Register at location 0xFFFD9000 with an offset of 0x08, and must be programmed before use.

Table 8-4. GPIO Port Memory Map

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	P1DRx	Port A/C/E/G/I/K/M Data Registers
0x04	P2DRx	Port B/D/F/H/J/L/N Data Registers
0x08	P1DDRx	Port A/C/E/G/I/K/M Data Direction Registers
0x0C	P2DDRx	Port B/D/F/H/L/N Data Direction Registers

8.2.2 Register Descriptions

8.2.2.1 Port A/C/E/G/I/K/M Data Registers (P1DRx)

Values written to P1DRx are output on the PA/PC/PE/PG/PI/PK/PM pins if the corresponding P1DDRx Data Direction bits are set for output. When the corresponding Data Direction Register bit for a pin is set for input, the value read is the state of the GPIO pin. Reading this register returns either:

- The last bit value written if the bit is configured as an output.
- The current value on the corresponding port pin if configured as an input.

Port K is only available on the LH79524. Port M is an output only port. This register will not input values from the Port M pins.

Table 8-5. P1DRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								PORT_DATA							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Port A: 0xFFFDF000 + 0x00 Port C: 0xFFFDE000 + 0x00 Port E: 0xFFFDD000 + 0x00 Port G: 0xFFFDC000 + 0x00 Port I: 0xFFFDB000 + 0x00 Port K: 0xFFFDA000 + 0x00 (LH79524 Only) Port M: 0xFFFD9000 + 0x00 (Bits 7 and 6 LH79524 Only)															

Table 8-6. P1DRx Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7:0	PORT_DATA	**Port Input/Output Data** Contains the bit-by-bit Port input or output data, depending on how the corresponding bit in the P1DDRx Register is programmed. Note that bits 7 and 6 of Port M exist on LH79524 only.

8.2.2.2 Port B/D/F/H/J/L/N Data Register (P2DRx)

Values written to P1DRx are output on the PB/PD/PF/PH/PJ/PL/PN pins if the corresponding P1DDRx Data Direction bits are set for output. When the corresponding Data Direction Register bit for a pin is set for input, the value read is the state of the GPIO pin. Reading this register returns either:

- The last bit value written if the bit is configured as an output.
- The current value on the corresponding port pin if configured as an input.

Port L is only in the LH79525. Port J is an input only port. This register will not output values onto the Port J pins.

Table 8-7. P2DRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								PORT_DATA							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Port B: 0xFFFDF000 + 0x04 Port D: 0xFFFDE000 + 0x04 Port F: 0xFFFDD000 + 0x04 Port H: 0xFFFDC000 + 0x04 Port J: 0xFFFDB000 + 0x04 Port L: 0xFFFDA000 + 0x04 (LH79524 Only) Port N: 0xFFFD9000 + 0x04 (4 bits, LH79524 Only)															

Table 8-8. P2DRx Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7:0	PORT_DATA	**Port Input/Output Data** Contains the bit-by-bit Port input or output data, depending on how the corresponding bit in the P1DDRx Register is programmed. Note that Port N consists of 4 bits and exists on LH79524 only.

8.2.2.3 Port A/C/E/G/I/K Data Direction Register (P1DDRx)

P1DDRx is the Data Direction Register. Bits in P1DDR Register program the corresponding Port pin to be either an input or an output. Port K, and Port M bits 7 and 6 are only in the LH79524.

Table 8-9. P1DDRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								PORT_DIRECTION							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Port A: 0xFFFDF000 + 0x08 Port C: 0xFFFDE000 + 0x08 Port E: 0xFFFDD000 + 0x08 Port G: 0xFFFDC000 + 0x08 Port I: 0xFFFDB000 + 0x08 Port K: 0xFFFDA000 + 0x08 (LH79524 Only) Port M: 0xFFFD9000 + 0x08 (Bits 7 and 6 LH79524 Only)															

Table 8-10. P1DDRx Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Writing to these bits has no effect. Reading returns 0.
7:0	PORT_DIRECTION	**Port Data Direction** Configures port pins as input or output. 1 = Output 0 = Input (Port M is output only and must programmed to 1 for use)

8.2.2.4 Port B/D/F/H/L/N Data Direction Register

P2DDRx is the Data Direction Register. Bits in P2DDRx Register program the corresponding Port pin to be either an input or an output. Port J has no P2DDRx.

Table 8-11. P2DDRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								PORT_DIRECTION							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Port B: 0xFFFDF000 + 0x0C Port D: 0xFFFDE000 + 0x0C Port F: 0xFFFDD000 + 0x0C Port H: 0xFFFDC000 + 0x0C Port L: 0xFFFDA000 + 0x0C (LH79524 Only) Port N: 0xFFFD9000 + 0x0C (LH79524 Only)															

Table 8-12. P2DDRx Register Definitions

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Writing to these bits has no effect. Reading returns 0.
7:0	PORT_DIRECTION	Configures port pins as input or output. 1 = Output 0 = Input

Chapter 9
I²C Module

The I²C Module implements the Inter-IC bus (I²C), and provides:

- Two-wire synchronous serial interface
- Operation in both the standard mode, for data rates up to 100 Mbits/s, and the fast mode, with data rates up to 400 Mbits/s
- Communication with devices in the fast mode as well as the standard mode if both are attached to the bus.

The I²C Module block diagram appears in Figure 9-1.

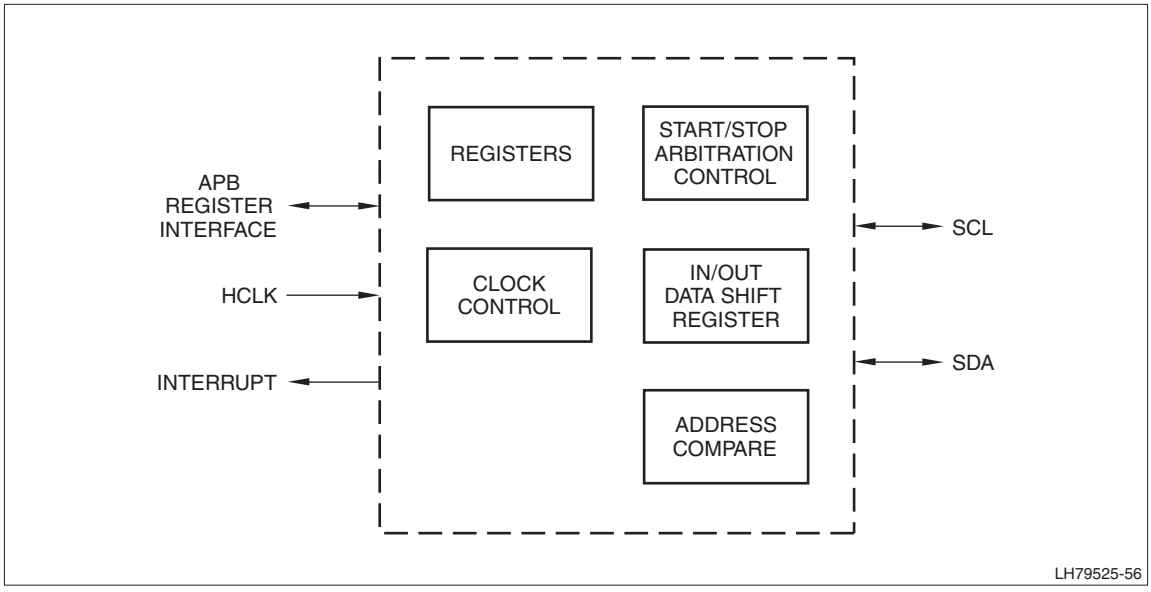


Figure 9-1. I²C Module Block Diagram

9.1 Theory of Operation

The LH79524/LH79525 implements a two-wire I^2C Module capable of operating in either Master or Slave mode. The block conforms to the I^2C 2.1 Bus Specification for data rates up to 400 kbps. The two wires (pins) in the interface are SCL (serial clock) and SDA (serial data).

The I^2C Module buffers a single byte of serial data on receive and transmit. Registers provide control over operating mode, serial clock frequency, and slave-mode address. A status register contains status bits that remain set until cleared by software.

The slaves each have a unique address that is determined by the system designer. When the master wants to communicate with a slave, the master transmits a start condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave will then send an acknowledge (ACK) pulse after the address and R/W bit is received to notify the master that the slave has received the request. If the master (master-transmitter) is writing to the slave, (slave-receiver), the receiver will receive a byte of data. This transaction will continue until the master terminates the transmission with a stop condition. If the master (master-receiver), is reading from the slave (slave-transmitter), the receiver will transmit a byte of data to the master, and the master will then acknowledge the transaction with the ACK pulse. This transaction will continue until the master terminates the transmission by not acknowledging the transaction after the last byte is received, and then the master will issue a stop condition. This is shown in Figure 9-2.

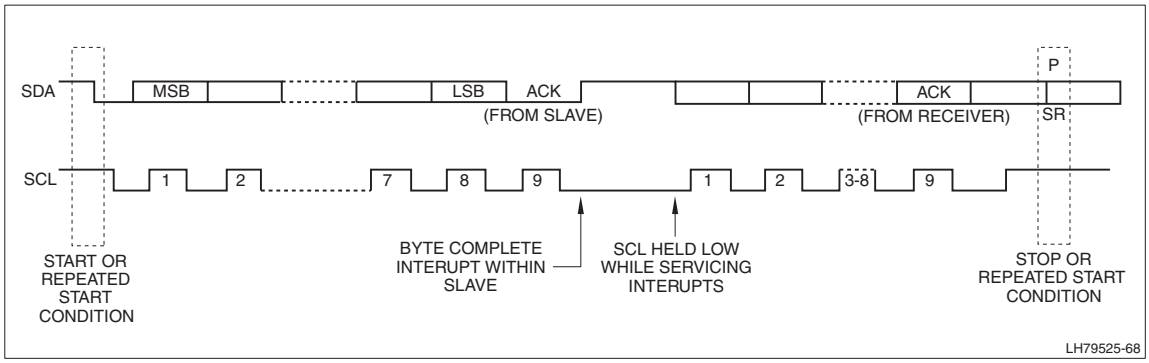


Figure 9-2. I^2C Bus Protocol

9.1.1 Setting I²C Clock Timing

When the I²C Module is in Master mode, the serial clock (SCL) is generated from HCLK, using two registers, ICHCNT and ICLCNT for timing parameters. When the I²C Module is in Slave mode, SCL is provided by the Master.

The equation for calculating the proper number of HCLKs required for setting the proper SCL clock HIGH and LOW period is:

$$HCNT = ROUND\_UP\ (MIN\_SCL\_HIGH_{time} \times HCLK\ frequency) - k)$$

$$LCNT = ROUND\_UP\ (MIN\_SCL\_LOW_{time} \times HCLK\ frequency) - k)$$

'ROUND_UP' means to round all fractions up to the next highest integer. The LOW count is calculated in the same way.

Permissible values for the parameters in the equations are found inTable 9-1. Example timing results appear in Table 9-2.

Table 9-1. I²C Clock Parameters

VALUE	400 kbit/s	100 kbits/s
k (constant)	4	3
MIN_SCL_HIGH	0.6 μs	4.0 μs
MIN_SCL_LOW	1.3 μs	4.7 μs

Table 9-2. Sample I²C HIGH Period Counts

I²C DATA RATE (kbit/s)	HCLK (MHz)	SCL HIGH REQUIRED MIN (μs)	HCNT	SCL HIGH TIME (μs)
100	6.6	4	24	4.09
100	9.9	4	37	4.14
100	51.6	4	204	4.01
400	10.0	0.6	2	0.60
400	15.3	0.6	6	0.654
400	20.0	0.6	8	0.66
400	51.6	0.6	27	0.600

The HCNT register does not translate directly into the SCL's high pulse width time. The design takes into consideration the rise time of the external pullup before activating the HCNT counter. By doing this, it ensures that the MCU can guarantee the high pulse width. The feedback sensor within the I²C block awaits a logic transition from 0 to 1 internally (0 to Pullup externally) on the I/O prior to enabling the high pulse width counter.

9.1.2 Interrupt Handling

In Slave mode, the I^2C Module handles address comparison, shifts data into or out of the ICDATA register, and generates ACK pulses at the appropriate times. In short, the interface hardware handles the bit-level operation of the protocol.

Interrupts are generated on both receive and transmit data, in a way similar to typical serial data interrupts. On transmit, when the data in ICDATA has been sent and the I^2C Module is ready to accept another byte of transmit data, a transmit interrupt is generated. On receive, when new data is received over the interface and placed into the ICDATA register, a receive interrupt is generated.

For the purposes of interrupt generation, no distinction is made between address bytes and data bytes. However, in Slave mode, the I^2C Module ignores transfers not addressed to it. In 7-bit addressing mode, addresses that do not match that of the I^2C Module do not generate interrupts. Nor do any following data transactions.

In 10-bit addressing mode, the address is transferred in two bytes. If the first transfer (containing the most significant address bits) matches the most-significant bits of the Slave address, an interrupt will be generated on both halves of the address. If the second part of the address does not match, the ICSTAT RXABORT bit is set, informing the interrupt handler that the address was not a complete match.

In transmit mode, the ICCON must be updated on a byte-by-byte basis, because the ICCON START bit must be set to 1 to initiate a byte transfer. This register also contains bits that set the operating mode.

In Master mode, bus synchronization is handled in hardware. Addressing, which was handled in hardware in Slave mode, is handled in software in Master mode. For example, the R/W bit of a 7-bit address must be set in software; it is not overwritten by the state of the R/W bit in the ICCON register.

Status bits in the ICSTAT register reports the precise state of the I^2C Module. For example, there are status bits reporting whether the current transfer is a Slave address or if a transmit abort or receive data overrun has occurred.

9.1.3 Slave Mode

In slave-receiver mode, the I^2C Module interrupts the processor whenever an address or data byte has been received. The sequence is that the byte is received and acknowledged by the I^2C Module, then the processor is interrupted. The ICDATA register will contain a data byte, 7-bit Slave address, or one of the two 10-bit Slave address bytes. Status bits in the ICSTAT register allow the processor to determine the type of transfer.

Whenever data is received, the ICSTAT:FULL bit is set. If the bit remains set, one more transfer can take place over the interface before the I^2C Module holds SCL LOW, preventing the master from sending any more bytes and creating an overflow condition. Reading the ICDATA register clears the bit.

In slave-transmitter mode, the I^2C Module interrupts the processor when an address is received, when the I^2C Module is ready to receive another data byte, and on repeat START conditions. When the I^2C Module is ready to send another byte, it sets the ICSTAT FULL bit. This will be cleared when the ICDATA register is written by the processor. In slave-transmitter mode, the ICCON register must be written for each byte, setting the ICCON START bit to initiate the transfer.

Interrupts set the ICSTAT INT bit. Before the interrupt routine is exited, this bit must be cleared by reading the ICSTAT register.

In address and repeat START transactions, reading the ICDATA and ICSTAT registers is all that is required of the interrupt handler, since address comparisons are performed in hardware.

9.1.4 Master Mode

Master mode is similar to Slave mode from an interrupt-handling point of view, but the I^2C Module now transmits rather than receives addresses, and bus arbitration must be performed as well.

Master-mode transactions start by testing ICSTAT IDLE bit to verify that the I^2C Module is idle, then initiating an address transaction. Address bytes and START bytes are generated in software and written to the ICDATA register as if they were data.

The handling of individual data interrupts is much the same as in Slave mode.

9.1.5 Resetting a Locked Slave

If transactions on I^2C have been interrupted, there is the possibility that a slave's state machine could be "dead-locked" into a particular state other than idle. Exiting this state requires issuing a Master Abort command, which is not possible with the I^2C module. However, the interface can be bit-banged using the GPIO function to signal a Master Abort (i.e., STOP condition, SCL High, and SDA Rising Edge). This should resolve any (one or many) peripheral slaves that have entered a dead-lock state. Once this is done, switch back to the I^2C function and continue to operate as normal.

9.2 Register Reference

This section provides the I^2C Module register memory mapping and bit fields.

9.2.1 Memory Map

The base address for the I^2C Module is 0xFFFC5000. Table 9-3 summarizes the I^2C Module registers.

Table 9-3. I^2C Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	ICCON	Configuration Register
0x04	ICSAR	Slave Address Register
0x08	ICUSAR	Upper Slave Address Register
0x0C	ICDATA	Data Register
0x10	ICHCNT	Clock High Period Count Register
0x14	ICLCNT	Clock Low Period Count Register
0x18	///	Reserved; Do Not Access
0x1C	ICSTAT	Status Register

9.2.2 Register Definitions

9.2.2.1 I²C Configuration Register (ICCON)

The ICCON register allows controlling the operating mode of the I²C Module, operating parameters, and contains the flags used to start a transfer and to set the data direction. Not all bits are active in every operating mode. Check the individual bit description to determine if the bit is active during a particular operating mode.

Table 9-4. ICCON Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SABT	RWC	STOP	START	SPEED	I2CEN	MODE	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x00															

Table 9-5. ICCON Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	SABT	**Stop After Bus Transaction** This bit is used when in Master mode and the ACK signal is not driven LOW by the Slave with which the I²C Module is communicating. Under those conditions, this bit is used to define whether or not a STOP Condition is generated after each bus transaction or not. This bit is also used to generated repeated Start (Sr) conditions, which is essential for 10-bit mode communications. The Sr condition is generated by programming the SABT and STOP bits to 1 for the current transaction. In this case, Sr is generated after the ACK is driven LOW by the slave. 1 = No STOP Condition is generated 0 = An I²C STOP Condition is generated after each bus transaction
6	RWC	**Read/Write Control** Only active in Master mode, it specifies direction for data transfers. 1 = Read (Master receiver) 0 = Write (Master transmitter)
5	STOP	**Stop Transfer** This bit is only active when in Master mode. It instructs the I²C Module to terminate the data transfer after the completion of the current transaction. Once the transaction is terminated, this bit is automatically reset to 0. This bit is also used to generate the Sr condition (see description in the SABT bit). 1 = Terminate data transfer following current transaction completion 0 = Do not terminate data transfer if additional data is pending

Table 9-5. ICCON Fields (Cont'd)

BITS	NAME	DESCRIPTION
4	START	**Start Transfer** This bit is active in both Master and Slave modes. In Master mode, this bit causes the I^2C Module to commence a transaction on the I^2C bus. In Slave mode this bit causes the I^2C Module to transmit a byte of data to the Master. This bit is automatically reset to 0 at the completion of the transaction. 1 = (Master) Commence transaction; (Slave) transmit one data byte 0 = No action
3	SPEED	**Fast/Standard Speed** Use this bit to set the transaction speed of the I^2C Module. 1 = Fast interface speed (400 kbit/s) 0 = Standard interface speed (100 kbit/s)
2	I2CEN	**I^2C Enable** This bit turns the I^2C Module on and off. 1 = I^2C Module is enabled 0 = I^2C Module is disabled (SCL and SDA are not driven)
1:0	MODE	**I^2C Module Mode** The I^2C Module Mode field sets the operating mode of the I^2C Module. 11 = I^2C Master Mode 10-Bit addressing 10 = I^2C Master Mode 7-Bit addressing 01 = I^2C Slave Mode 10-Bit addressing 00 = I^2C Slave Mode 7-Bit addressing

9.2.2.2 I^2C Slave Address Register (ICSAR)

Software programs this register with the unit address used by the I^2C Module when in Slave mode. In 7-bit addressing mode, the entire address is contained in this register. In 7-bit mode, bit 0 is not used.

In 10-bit addressing mode, this register holds the lower 8 address bits; the upper 2 address bits and the R/W bit are in the ICUSAR register. This register is not used in Master mode.

Table 9-6. ICSAR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SLAD7							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x04															

Table 9-7. ICSAR Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.

Table 9-7. ICSAR Fields

BITS	NAME	DESCRIPTION
7:0	SLAD7	**Least-Significant 7 bits of Slave Address** SLAD7[7:1] holds the lower 7 bits of the I^2C Module's Slave address. In 10-bit addressing mode, SLAD7[7:0] holds the lower 8 bits of the slave address. In 7-bit mode, SLAD7[7:1] contains the address, and SLAD7[0] is not used.

9.2.2.3 I²C Upper Slave Address Register (ICUSAR)

Software programs the ICUSAR register with the upper 2 address bits in 10-bit addressing mode, plus the read/write data direction (SRW) bit. This register is not used in Master mode.

Table 9-8. ICUSAR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								UPPERAD					SLAD89		///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x08															

Table 9-9. ICUSAR Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:3	UPPERAD	**Upper Address** This field is only used for 10-bit mode. When using 10-bit mode, this field must be programmed to 0b11110. When using 7-bit mode, this field must be programmed to 0b00000, which is the reset value.
2:1	SLAD89	**Slave Address Bits [9:8]** This field contains the upper 2 bits of the 10-bit slave address. This field is not used in 7-bit addressing mode.
0	///	**Reserved** Reading returns 0. Write the reset value.

9.2.2.4 I²C Data Register (ICDATA)

The ICDATA register holds the received data or the data to be transmitted.

Table 9-10. ICDATA Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								DAT							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x0C															

Table 9-11. ICDATA Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	DAT	**I²C Data** This field contains the transmitted or received I²C data.

9.2.2.5 I²C Clock High Time Register (ICHCNT)

The ICHCNT register allows programming the length of the serial clock HIGH time.

Table 9-12. ICHCNT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								HCNT							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x10															

Table 9-13. ICHCNT Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	HCNT	**High Count** This field allows programming the SCL HIGH time, in HCLK periods. The value is a hexadecimal number. The HIGH time is ICHCNT + 3 HCLK periods in 100 kbit/s mode, and ICHCNT + 4 HCLK cycles in 400 kbit/s mode. The ICHCNT Register must be programmed before any I²C bus transaction can take place to insure proper timing.

9.2.2.6 I²C Clock Low Time Register (ICLCNT)

The ICLCNT register allows programming the length of the serial clock LOW time.

Table 9-14. ICLCNT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								LCNT							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC5000 + 0x14															

Table 9-15. ICLCNT Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	LCNT	**Low Count** This register allows programming the SCL LOW time, in HCLK periods. The value is a hexadecimal number. The LOW time is ICLCNT + 3 HCLK periods in 100 kbit/s mode, and ICLCNT + 4 HCLK cycles in 400 kbit/s mode. The ICLCNT Register must be set before any I²C bus transaction takes place to insure proper timing. Note that the value in this register must be at least 3 for proper I²C operation.

9.2.2.7 I²C Status Register (ICSTAT)

The ICSTAT register provides status regarding the state of the module.

Table 9-16. ICSTAT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SLAVEAD	RXABORT	TXABORT	IDLE	10BITADDR	///	FULL	INTR
RESET	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC5000 + 0x1C															

Table 9-17. ICSTAT Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	SLAVEAD	**Slave Address** 1 = Last byte received on the I²C bus was a Slave address byte 0 = Last byte received was not a Slave address byte
6	RXABORT	**Receive Abort** This bit indicates a Receive fault. 1 = (Slave Mode) The I²C Module is in 10-bit Slave mode and the upper address bits matched but the lower address bits did not. 1 = (Master Mode) The upper and lower address bits match but a restart was issued by the Master in Master-receive mode, and the repeated upper address does not match. 0 = No Receive Abort
5	TXABORT	**Transmit Abort** This bit indicates a Transmit fault. This bit remains 1 until the ICSTAT Register is read by software. 1 = I²C Module is operating in the Master-transmitter mode and a Slave device does not respond with an ACK signal after receiving a byte of data, or arbitration was lost. 0 = No Transmit Abort
4	IDLE	**Idle** Indicates that no messages are currently being processed. 1 = I²C Module is not processing any messages 0 = I²C Module is actively processing messages
3	10BITADDR	**10-bit Address** This status bit denotes that a 10-bit address was detected. This bit remains 1 until automatically cleared when the ICSTAT Register is read by software. 1 = A 10-bit address is detected 0 = No 10-bit address detected
2	///	**Reserved** Reading returns 0. Write the reset value.

Table 9-17. ICSTAT Fields (Cont'd)

BITS	NAME	DESCRIPTION
1	FULL	**Full Flag** Indicates that a byte of address or data has been received on the I^2C bus and written into the ICDATA register. This bit remains 1 until automatically cleared when the ICDATA Register is read by software or until the TXABORT bit is set to 1. 1 = A byte of address or data has been received on the I^2C bus and written into the ICDATA register 0 = No address or data byte received
0	INTR	**Interrupt** This bit indicates the source of the interrupt condition. This bit remains 1 until reset by software. The source of the interrupt can be: • A data byte has been received on the I^2C bus and written to ICDATA • A Stop condition has been detected • The FULL flag is 1 • In Master mode, when the TXABORT flag is 1 • In Slave mode, when the RXABORT flag is 1 1 = Interrupt condition is active 0 = No interrupt

Chapter 10
I²S Converter

The Synchronous Serial Port (SSP) to I²S converter is an interface that converts a synchronous serial communication stream in TI DSP-compatible mode into an I²S compliant synchronous serial stream. The I²S converter operates on serial data in both master and slave mode.

- The I²S converter provides:
  - Programmable Word Select (WS) delay
  - Left/right channel information:
    - Current WS value at the pin
    - WS value associated with next entry written to TX FIFO
    - WS value associated with next entry read from RX FIFO
  - Ability to invert WS state
  - Ability to invert the bit clock
- Supports frame size of 16 bits only. Any other frame size will result in a frame size error. Each frame transmits starting with the most-significant bit.
- Master and Slave modes supported
- A single combined interrupt is generated as an OR function of the individual interrupt requests.
- Additional interrupts:
  - Transmit FIFO underrun
  - SSP Protocol error
  - External CODEC Protocol error
- Passes SSP data unaltered when module is not enabled.
- Loopback Test Mode support.

Figure 10-1 shows a block diagram of the I²S Converter.

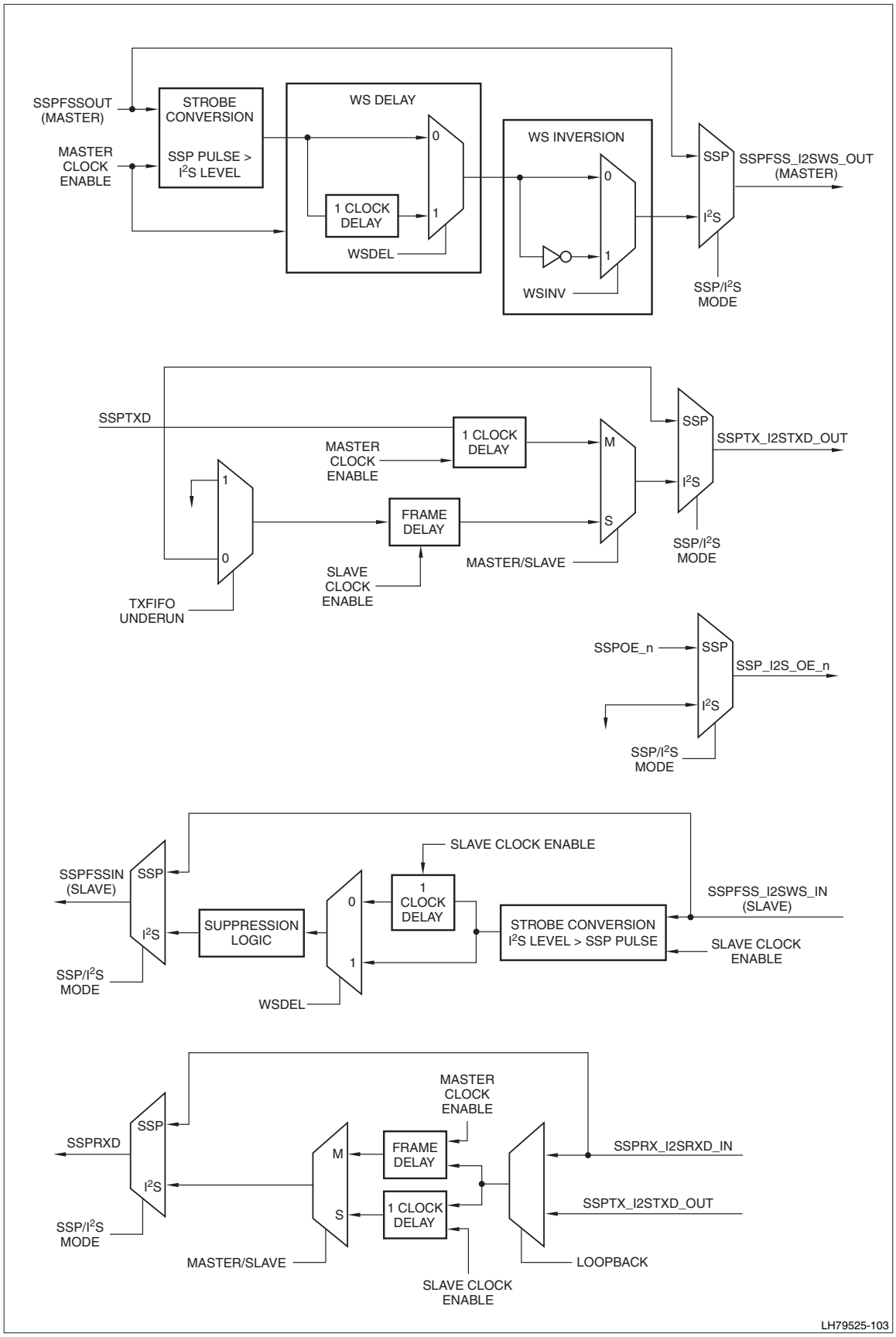


Figure 10-1. I²S Converter Block Diagram

10.1 Theory of Operation

10.1.1 Conversion

The SSP-to-I^{2}S converter converts a data stream from the TIDSP format to the I^{2}S format on transmit or from I^{2}S format to TI DSP format on receive. The TI DSP format is supported by the SSP block and is fully described in the SSP Chapter. The I^{2}S format is an audio standard made popular by Phillips Semiconductor. Figure 10-2 shows the Texas Instruments DSP format for continuous transfers.

The I^{2}S frame format is described in the I^{2}S Bus specification published by Phillips Semiconductors. An example I^{2}S transaction is shown in Figure 10-3.

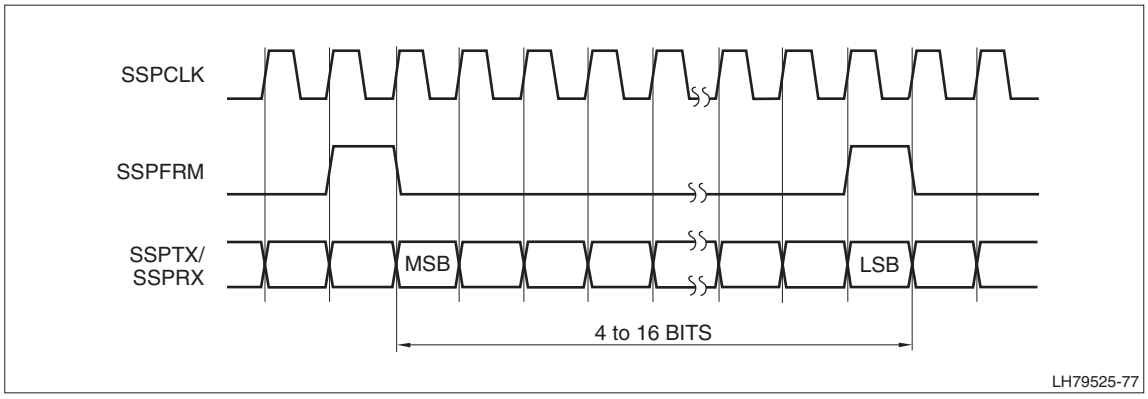


Figure 10-2. TI SSP Frame Format

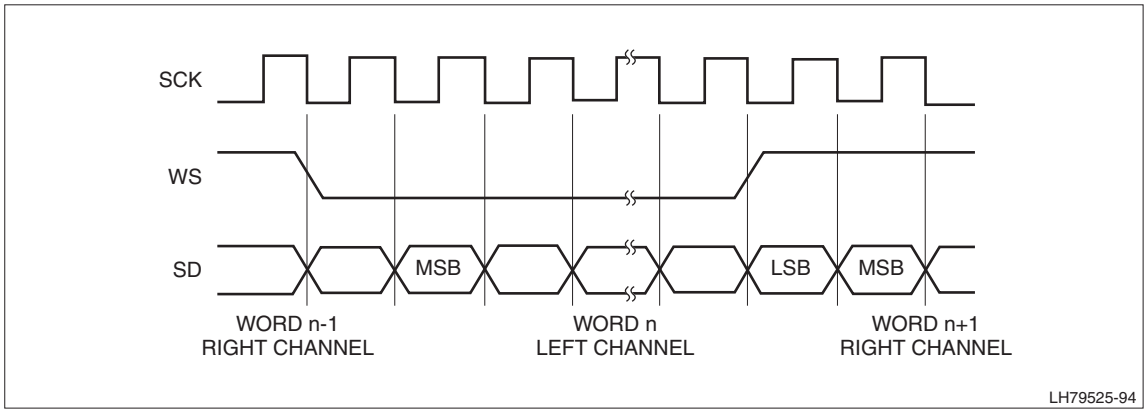


Figure 10-3. I^{2}S Format

The I^2S converter operates in both Master and Slave Modes. In Master Mode, the clock and word select inputs (SSPCLKOUT and SSPFSSOUT) are supplied by the SSP block. In slave mode, the clock and word select inputs are supplied by the external CODEC via the PB3/SSPCLK/I2SCLK and PB2/SSPFRM/I2SWS pins.

The I^2S Converter requires the SSP to be enabled and operating in continuous mode. It is also required that the SSP operate in full duplex mode (SOD = 0) while the I^2S Converter is enabled. The SOD allows the SSP to operate in receive-only mode which may result in a condition where the channel indicated by the TXFIFO (left or right) may differ from the channel indicated by the RXFIFO. The I^2S Converter requires the FIFOs to be in sync with respect to which channel each is expecting. If the FIFOs become out of sync, then the behavior of the I^2S may be unpredictable. For this reason, I^2S is not compatible with receive-only mode of the SSP.

The SSPOE_n signal from the SSP is altered by the I^2S Converter to enable the transmit data pad whenever the I^2S is enabled. If the I^2S Converter is disabled, then the output enable passes unchanged through the I^2S Converter.

10.1.2 Driving/Latching Edges

The SSP and any device connected to it is expected to drive data and the frame pulse on the rising edge of the clock and latch both on the falling edge of the clock. The I^2S Specification states that the I^2S Converter may drive data and the word strobe on either edge of the clock but both are always latched on the rising edge of the clock.

Therefore, the data and frame pulse received by the I^2S from the SSP are sampled on the falling edge of the clock and moved to the rising edge of the clock. The data and word strobe received by the I^2S at the pins of the chip are sampled on the rising edge of the clock. If the I^2S Converter is disabled, the SSP data and frame pulse are passed through without being altered by the I^2S.

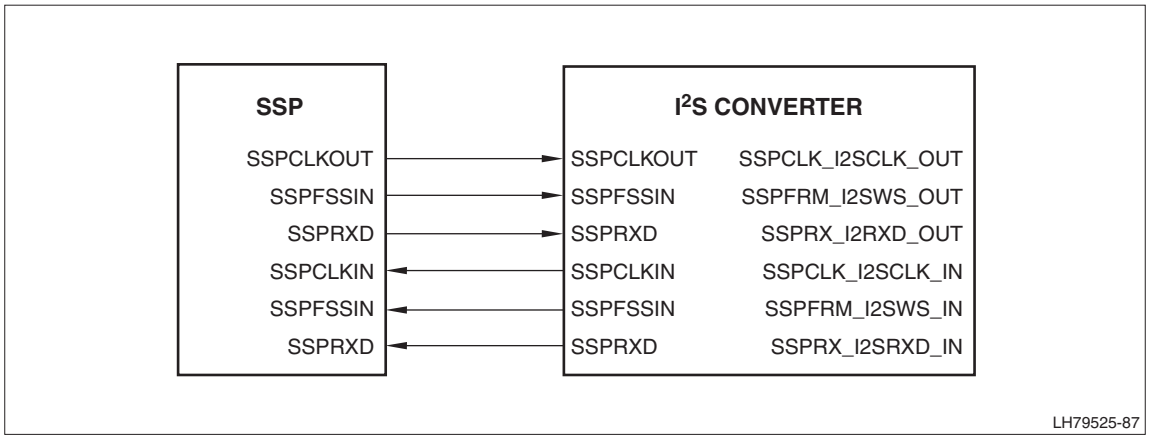


Figure 10-4. Driving/Latching Diagram

10.1.3 Transmission

10.1.3.1 Master Mode Transmission

During Master Mode transmission, the I^2S converter supplies the clock, frame output, and data (on the PB5/SSPTX/I2STXD/UARTTX0/UARTIRTX0 pin) to the external CODEC. The Master Mode clock is the SSP master mode clock, SSPCLKOUT, inverted as indicated by the CTRL:CLKINV bit.

The SSP frame output pulse is converted to a level to generate PB2/SSPFRM/I2SWS. If WSDEL is set to 1, then PB2/SSPFRM/I2SWS is delayed by one clock so that it transitions with the MSB of the transmitted data.

The data to be transmitted is received by the I^2S converter from the SSP. It is delayed by one clock and then transmitted on the PB5/SSPTX/I2STXD/UARTTX0/UARTIRTX0 pin.

Note that when in master mode, the I^2S Converter requires that the SSP operate in TI continuous mode. If the transmit FIFO is starved, the SSP will operate in single-word transfer mode in which is stops generating SSPCLKOUT and SSPFSSOUT as soon as the transfer of the last word in the FIFO has completed. The I^2S Converter needs these signals to perform its conversion.

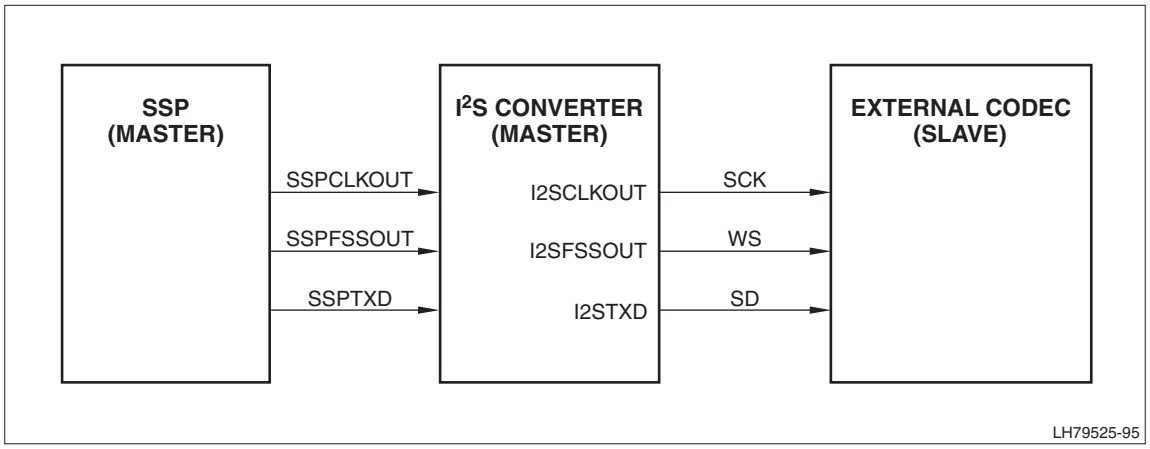


Figure 10-5. I^2S Master Mode Transmission Block Diagram

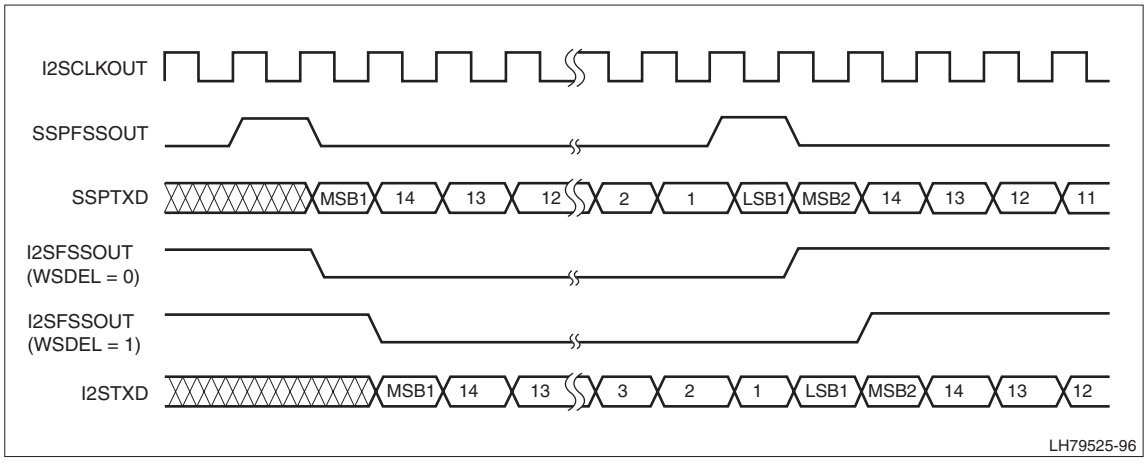


Figure 10-6. I^2S Master Mode Transmission Timing Diagram

10.1.3.2 Slave Mode Transmission

During Slave Mode transmission, the I^2S converter receives its clock (PB3/SSPCLK/I2SCLK) and frame input (PB2/SSPFRM/I2SWS) from the external CODEC. In response to the external CODEC signals, the I^2S transmits data on the PB5/SSPTX/I2STXD/UARTTX0/ UARTIRTX0 pin.

The slave mode clock received by the SSP is the I^2S slave mode clock input, PB3/SSPCLK/I2SCLK, inverted as indicated by the CTRL:CLKINV bit.

The frame input received by the I^2S converter is converted to a pulse and sent to the SSP on SSPFSSIN. This conversion is accomplished by generating a pulse to the SSP for every edge detected on PB2/SSPFRM/I2SWS. If WSDEL is set to 0, then the pulse is delayed by 1 clock. In this case, the data cannot be received from the SSP and transmitted to the external CODEC in time, as depicted by Figure 10-7. For this reason, the data received from the SSP in slave mode is delayed by the I^2S converter before being transmitted on the PB5/SSPTX/I2STXD/UARTTX0/UARTIRTX0 pin. This results in a one-frame lag in the transmission of data in slave mode, regardless of the value of CTRL:WSDEL. Until the delay pipe is filled, the External Codec will receive a 0 on the PB5/SSPTX/I2STXD/UARTTX0/UARTIRTX0 pin.

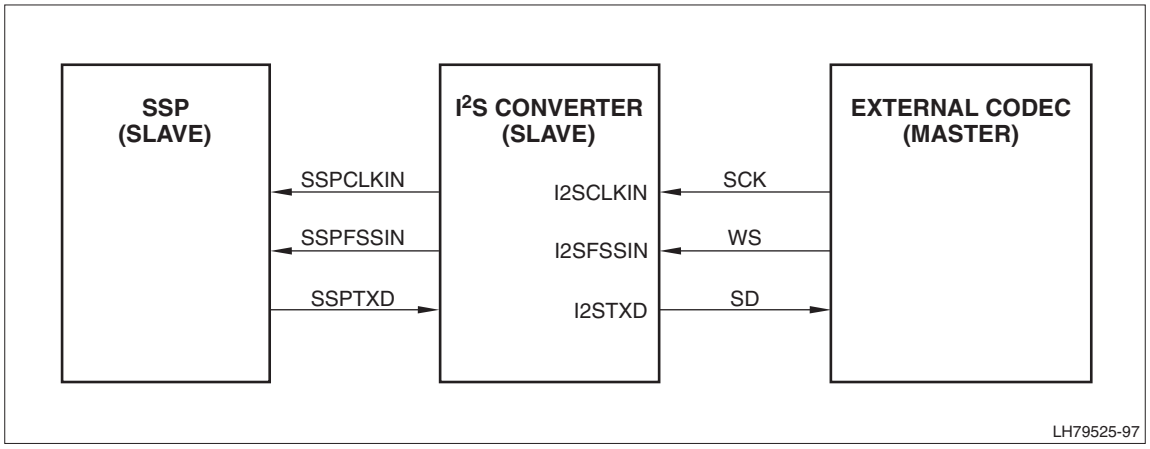


Figure 10-7. I^2S Slave Mode Transmission Block Diagram

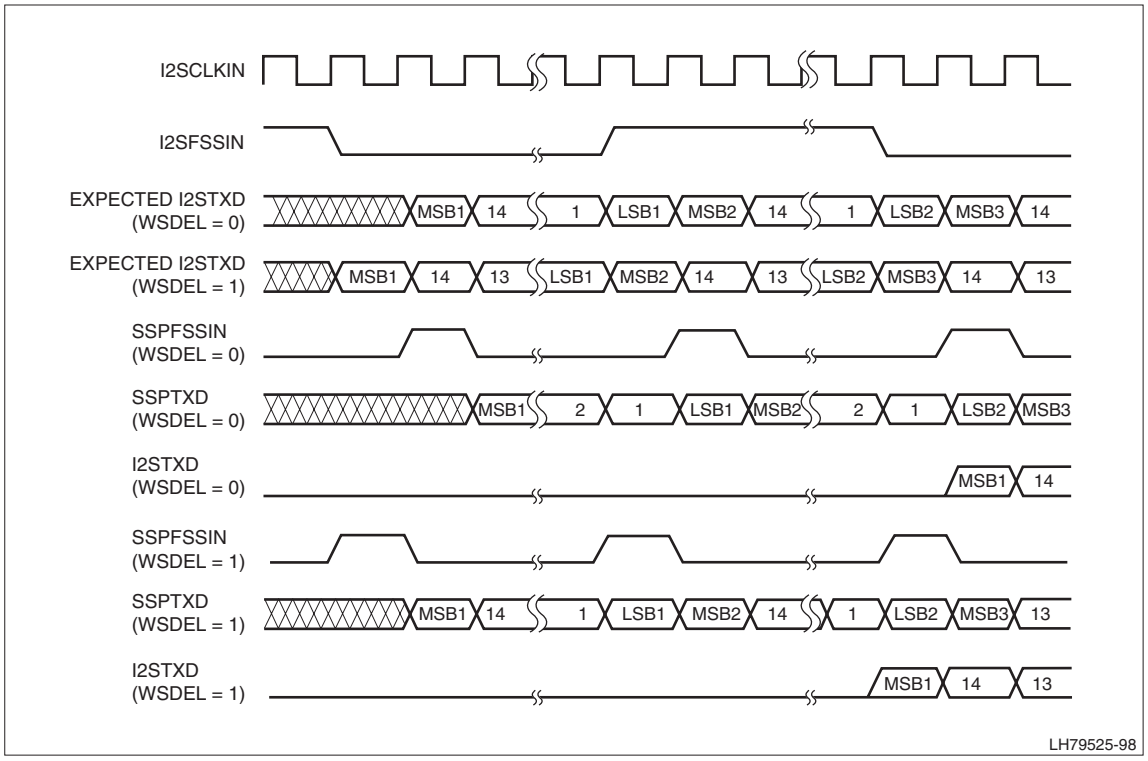


Figure 10-8. I^2S Slave Mode Transmission Timing Diagram

10.1.4 Reception

10.1.4.1 Master Mode Reception

During Master Mode reception, the I^2S converter supplies the clock (PB3/SSPCLK/I2SCLK) and frame output (PB2/SSPFRM/I2SWS). In response to signals from the I^2S converter, the external CODEC sends data on SSPRX_I2SRXD_IN.

PB2/SSPFRM/I2SWS is formed by toggling PB2/SSPFRM/I2SWS each time a pulse is received by the I^2S converter from the SSP on SSPFSSOUT. If WSDEL is set to 1, then the level is delayed by a clock. In this case, the data cannot be received from the external CODEC and transmitted to the SSP in time, as depicted by Figure 10-9. For this reason, the data received from the external CODEC in slave mode is delayed by the I^2S converter before being transmitted to the SSP on SSPTXD. This results in a one-frame lag in the reception of data in master mode, regardless of the value of WSDEL. Until the delay pipe is filled, the SSP will receive a logic low on SSPRXD, causing the first two entries in the SSP Receive FIFO to be filled with 0x000.

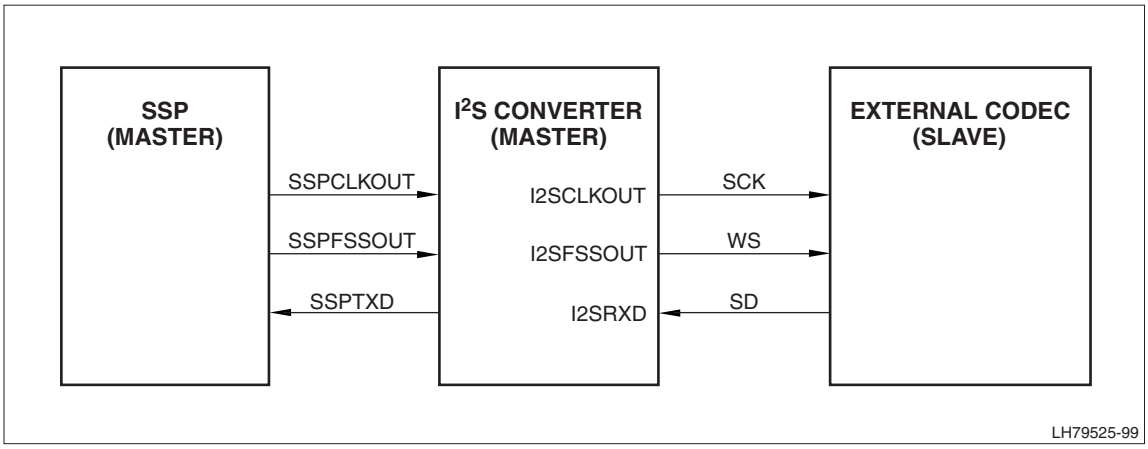


Figure 10-9. I²S Master Mode Reception Block Diagram

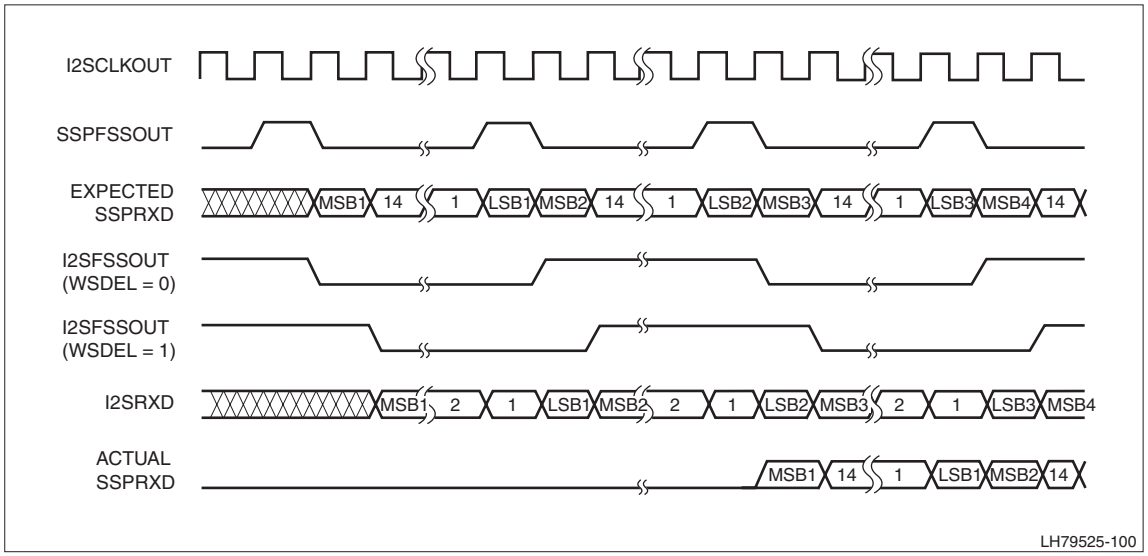


Figure 10-10. I²S Master Mode Reception Timing Diagram

10.1.4.2 Slave Mode Reception

During Slave Mode reception, the I^2S converter receives its clock (PB3/SSPCLK/I2SCLK), frame input (PB2/SSPFRM/I2SWS) and data (PB4/SSPRX/I2SRXD/UARTRX0/ UARTIRRX0) from the external CODEC. The slave mode clock received by the SSP is the I^2S slave mode clock input, PB3/SSPCLK/I2SCLK, inverted as indicated by the CTRL:CLKINV bit.

The received frame is converted to a pulse and sent to the SSP. This conversion is accomplished by generating a pulse to the SSP for every edge detected on PB2/SSPFRM/I2SWS. If WSDEL is 0, the pulse is delayed by one clock.

The data received by the I^2S converter from the external CODEC is delayed by one clock and sent to the SSP on SSPRXD.

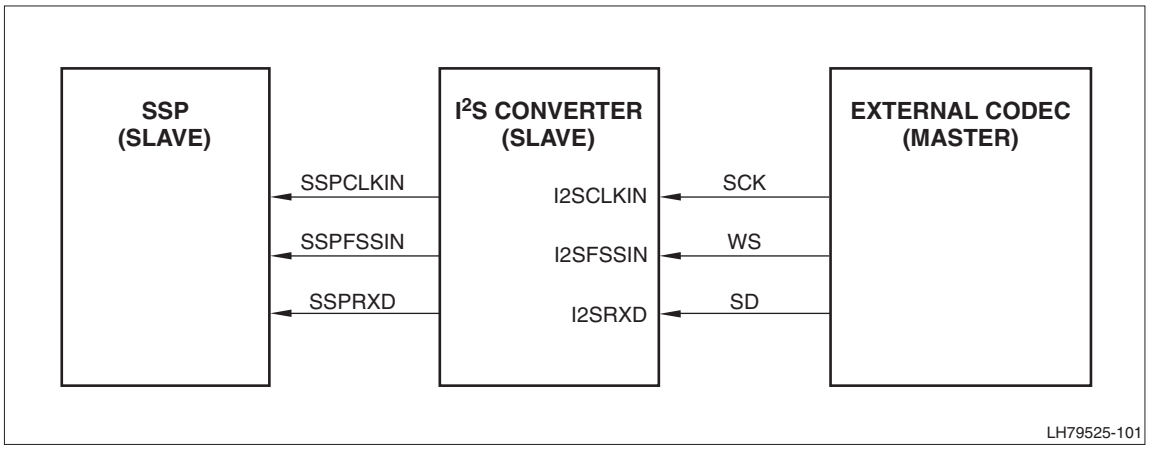


Figure 10-11. I^2S Slave Mode Reception Block Diagram

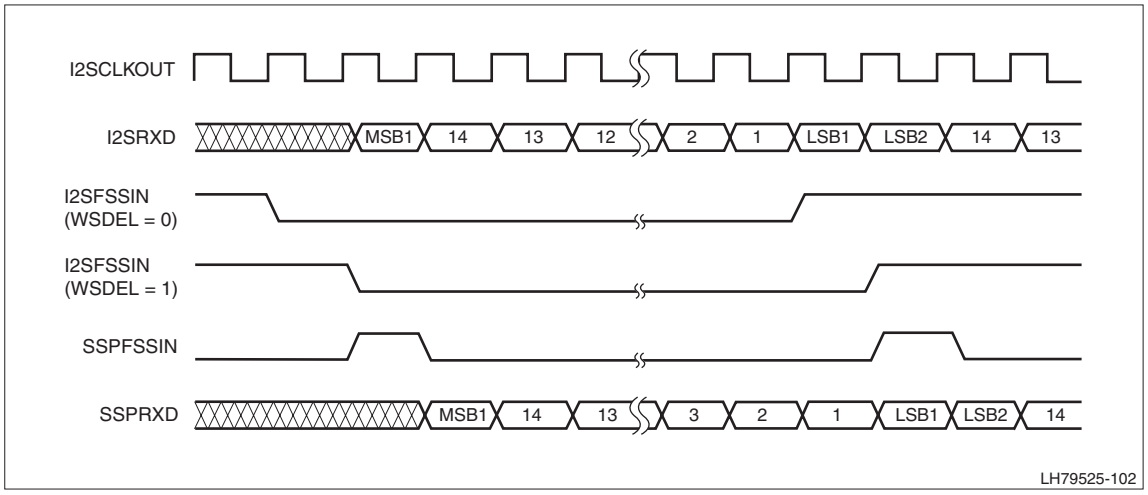


Figure 10-12. I^2S Slave Mode Reception Timing Diagram

10.1.5 Suppression of SSPFSSIN

The assertion of SSPFSSIN to the SSP is suppressed under the following conditions:

- When the channel indicated by the Transmit FIFO differs from the channel expected by the External Codec. Since the I^2S Converter is assumed to operate in full-duplex mode, the channels indicated by the TXFIFO and RXFIFO should always match, so only the Transmit FIFO is monitored. When the channel in the Transmit FIFO matches the channel expected by the External Codec, the SSPFSSIN pulse is not suppressed.
- During a Transmit FIFO Underrun. If the I^2S Converter asserts SSPFSSIN to the SSP while the TXFIFO is empty, then the SSP will continually re-transmit whatever data was last in TXFIFO(0) and indicate the channel associated with TXFIFO(0). Due to the suppression logic described above, every other pulse to the SSP would be suppressed and the data value transmitted by the SSP would alternate from the last 16-bit value in TXFIFO(0) and the last bit in the SSP shift register. This violates the requirement that known data be sent during a TXIFO Underrun. Therefore, asserting SSPFSSIN will be suppressed during a TXFIFO Underrun while the I^2S Converter is enabled. Additionally, a logic 0 will be fed into the I^2S Converter Frame Delay pipe so that any remaining valid data is flushed out, followed by logic 0.
- When the I^2S Converter is operating in master mode.

The suppression of SSPFSSIN and forcing 0 on the shift register input only occurs when the I^2S Converter is enabled in slave mode. If the I^2S is disabled, the SSP is not altered.

10.1.6 Channel Management

The Word Select (WS) signal from the I^2S specification is carried on the PB2/SSPFRM/I2SWS/IN signal. This signal transitions one SSPCLK_I2SCLK_OUT/IN before the MSB of a new data item is sent/received and its state describes if the data is left or right channel. The equivalent TI mode signal, SSPFSSOUT/IN, is a one SSPCLKOUT/IN long pulse that signals the beginning of data for one clock.

There are several mechanisms to control and report the behavior of the WS signal.

The WS bit may be delayed, with respect to the data, to transition on the same clock as the MSB of the data via the WSDEL control bit. The function of WS may be inverted via the WSINV control bit. The value of WS at the pin can be sampled via the WS status bit.

10.1.7 Interrupts

The I^2S Converter can assert seven types of interrupts. Only the single combined interrupt, I2SINTR, goes to the VIC:

- SSPPE — I^2S SSP Protocol Error Interrupt request (Frame size out of bounds), generated by I^2S Converter, locally maskable
- ECPE — I^2S External Codec Protocol Error Interrupt request (Frame size out of bounds), generated by I^2S Converter, locally maskable
- TXUE — SSP Transmit FIFO Underrun Error Interrupt request, generated by I^2S Converter, locally maskable
- SSPRXINTR — SSP Receive FIFO Service Interrupt request, generated by SSP
- SSPTXINTR — SSP Transmit FIFO Service Interrupt request, generated by SSP
- SSPRORINTR — SSP Receive Overrun Interrupt request, generated by SSP
- SSPRXTOINTR — SSP Receive FIFO Timeout Interrupt request, generated by SSP

All seven interrupts are combined into a single interrupt: I2SINTR. This interrupt supersedes SSPINTR.

The status of the seven individual interrupt sources can be read from the MIS or RIS Register. Only the I^2S converter specific interrupts can be masked in the I2S IMSC register. The SSP mask bits are present, but only as read-only status bits.

10.1.7.1 SSP Protocol Error Interrupt

SSPPE is the SSP Protocol Error Interrupt. This Interrupt is asserted when the SSP and I^2S are enabled and the SSP is configured for the wrong frame length. This is a new interrupt and only applies to I^2S transactions.

10.1.7.2 External Codec Protocol Error Interrupt

ECPE is the External Codec Protocol Error Interrupt. This Interrupt is asserted when the I^2S is operating in slave mode and a frame is transmitted or received with the wrong frame length. This is a new interrupt and only applies to I^2S transactions.

10.1.7.3 Transmit FIFO Underrun Interrupt

TXUE is the Transmit FIFO Underrun Interrupt. This interrupt is asserted when the FIFO is empty, but a new transmission is begun, causing an underrun of the FIFO. This is a new I^2S Converter interrupt, but it is valid for other SSP modes, regardless of whether the I^2S Converter is enabled. Since the SSP shuts off the master clock once the Transmit FIFO runs out of data, this error only applies to slave mode transmission.

In the event of a Transmit FIFO Underrun while the I^2S is enabled, a 0 will be fed into the transmit frame delay pipe so that remaining valid data bits will be shifted out, followed by a 0. The frame pulse generated to the SSP block by the I^2S Converter is suppressed.

If the I^2S Converter is disabled during an underrun condition, the behavior of the SSP is not altered (SSPFSSIN is not suppressed, and logic '0' is not fed into the slave delay pipe).

10.1.7.4 Receive Interrupt

SSPRXINTR is the Receive Interrupt. This interrupt is asserted when there are four or more valid entries in the receive FIFO. The interrupt is cleared by reading the receive FIFO until there are three or fewer entries. This interrupt originates in the SSP.

10.1.7.5 Transmit Interrupt

SSPTXINTR is the Transmit Interrupt. This interrupt is asserted when the FIFO is less than or equal to half full (when there is space for four or more entries). The interrupt is cleared when there are five or more entries in the transmit FIFO. This interrupt originates in the SSP.

10.1.7.6 Receive Overrun Interrupt

SSPRORINTR is the Receive Overrun Interrupt. This interrupt is asserted when the FIFO is already full and an additional data frame is received, causing an overrun of the FIFO. Data is over-written in the Shift Register, but not the FIFO. This interrupt originates in the SSP.

10.1.7.7 Receive Timeout Interrupt

SSPRXTOINTR is the Receive Timeout Interrupt. This interrupt is asserted if the receive FIFO does not generate a further service request interrupt (SSPRXINTR) within a fixed number of HCLK periods.

10.1.7.8 I2SINTR

The MMPE, SMPE, TXUE, SSPINTR interrupts are combined into the single output I2SINTR. This interrupt is an OR function of the individual interrupt sources. This combined interrupt is the only one going to the Vectored Interrupt Controller (VIC).

The combined I^2S Interrupt is asserted if any of the seven individual interrupts from the SSP and I^2S Converter are asserted and enabled. The I^2S Interrupt supersedes the SSPINTR from the SSP.

10.2 Register Reference

This section describes the registers used in I^2S Converter.

10.2.1 Memory Map

The base address for the I^2S Converter is 0xFFFC8000.

Locations at offsets 0x018 through 0xFFF are reserved and must not be used during normal operation. Table 10-1 shows the memory mapping.

Table 10-1. I^2S Converter Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	CTRL	Control Register
0x004	STAT	Status Register
0x008	IMSC	Interrupt Mask Set and Clear Register.
0x00C	RIS	Raw Interrupt Status register
0x010	MIS	Masked Interrupt Status Register
0x014	ICR	Interrupt Clear Register
0x018 - 0xFFF	///	Reserved — Do not access

10.2.2 Register Descriptions

Note that SSP register bits duplicated in the I^2S Converter will lag the SSP version of the bit by one clock.

10.2.2.1 Control Register (CTRL)

This register allows control of various I^2S Converter functions, including Loopback, clock inversion, WS control, enabling the converter and selecting its mode. Notice that some of the functions apply only to I^2S transactions; for SSP transactions, those bits are not active. Explanation of the 'WS' function appears following Table 10-3.

Table 10-2. CTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///										LOOP	CLKINV	WSDEL	WSINV	I2SEN	I2SLBM
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC8000 + 0x000															

Table 10-3. CTRL Register Definitions

BITS	NAME	DESCRIPTION
31:6	///	**Reserved** Reading returns 0. Write the reset value.
5	LOOP	**Loopback Mode** Applies only to I^2S Transactions. Note that two frames of 0x0000 will still be received in master mode. 1 = Transmit and Receive internally connected for Loopback Mode 0 = Normal operation
4	CLKINV	**Clock Invert** Applies only to I^2S Transactions. Inverts the polarity of the SSPCLK or I2SCLK, which is output on the PB3SSPCLK/I2SCLK pin. 1 = Invert SSPCLK/I2SCLK 0 = Do not invert SSPCLK/I2SCLK
3	WSDEL	**WS Delay** Applies only to I^2S Transactions 1 = WS transitions with MSB (left justified) 0 = WS transitions one clock before MSB (I^2S justified)
2	WSINV	**WS Invert** Applies only to I^2S Transactions (see Table 10-6) 1 = Invert the function of WS (first sampled driven/latched will have WS = 1) 0 = No change in function of WS (first sampled driven/latched will have WS = 0)
1	I2SEN	**Enable I^2S Converter** When this bit is 0, the registers in the I^2S data and frame paths are cleared, and the master and slave clock inputs are gated off. 1 = Enable I^2S converter, convert between SSP mode and I^2S formats 0 = Disable I^2S converter, pass through SSP signals unchanged
0	I2SEL	**I^2S Select** 1 = I^2S functions are selected 0 = SSP functions are selected

10.2.2.1.1 Implementation of WSDEL

WSDEL is used to delay the assertion of the frame input/output. During master mode, if WSDEL = 1, then the frame output from the SSP is delayed by one clock before being asserted on the PB2/SSPFRM/I2SWS pin. If WSDEL = 0, then frame output is passed through without additional delay. During slave mode, if WDSEL = 0 the frame input is delayed by one clock before being asserted to the SSP so that the SSPFSSIN pulse coincides with the MSB of the data. If WSDEL = 1, the frame input is passed to the SSP without additional delay.

10.2.2.1.2 Implementation of WSINV

As seen in Table 10-4, if the WSINV = 0, a logic low on the PB2/SSPFRM/I2SWS pin indicates that the left channel is transmitting, while a logic high indicates that the right channel is transmitting. If the WSINV bit is set, then a logic low on the PB2/SSPFRM/I2SWS pin indicates that the right channel is transmitting, while a logic high indicates that the left channel is transmitting. After reset, the default value of WSINV is a logic low, indicating the left channel.

Table 10-4. WSINV Functionality

WSINV	PB2/SSPFRM/I2SWS = 0	PB2/SSPFRM/I2SWS = 1
0	Left	Right
1	Right	Left

10.2.2.2 Status Register (STAT)

This register reports various I^2S converter functions. All bits are read only.

Table 10-5. STAT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///							MS	RFF	RFE	TFF	TFE	TXWS	RXWS	WS	LBM
RESET	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC8000 + 0x004															

Table 10-6. STAT Register Definitions

BITS	NAME	DESCRIPTION
31:9	///	**Reserved** Reading returns 0. Write the reset value.
8	MS	**Master or Slave Mode Select (From SSP CTRL1:MS bit)** 1 = Device configured as slave 0 = Device configured as a master (default)
7	RFF	**Receive FIFO Full (From SSP SR:RFF bit)** 1 = The Receive FIFO is full 0 = The Receive FIFO is not full
6	RFE	**Receive FIFO Empty** 1 = The Receive FIFO is empty 0 = The Receive FIFO is not empty
5	TFF	**Transmit FIFO Full** 1 = The Transmit FIFO is full 0 = The Transmit FIFO is not full
4	TFE	**Transmit FIFO Empty (From SSP SR:TFE bit)** 1 = The Transmit FIFO is empty 0 = The Transmit FIFO is not empty
3	TXWS	**WS value for TX word** Applies only to I^2S Transactions 1 = Next word written to SSP TX FIFO will have WS = 1 0 = Next word written to SSP TX FIFO will have WS = 0
2	RXWS	**WS value for RX word** Applies only to I^2S Transactions 1 = Next word read from SSP RX FIFO will have WS = 1 0 = Next word read from SSP RX FIFO will have WS = 0
1	WS	**WS value at pin** This status bit is a real-time report of WS value at the pin. Applies only to I^2S Transactions
0	LBM	**I^2S Loopback Mode** Applies only to I^2S Transactions 1 = Output of I^2S Converter connected to input of I^2S Converter internally 0 = Normal serial port operation enabled

10.2.2.3 Interrupt Mask Set or Clear Register (IMSC)

On a Read, this register gives the current value of the mask on the relevant interrupt. Writing 1 to the particular bit sets the mask, enabling the interrupt to be read. Writing 0 clears the corresponding mask. All bits are cleared to 0 when reset.

Table 10-7. IMSC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									SSPPEM	ECPEM	TXUEM	TXIM	RXIM	RTIM	RORIM
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC8000 + 0x008															

Table 10-8. IMSC Register Definitions

BITS	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	SSPPEM	**SSP Protocol Error mask** 1 = Master Mode Protocol Error condition interrupt enabled 0 = Master Mode Protocol Error condition interrupt is masked
5	ECPEM	**External Codec Protocol Error mask** 1 = Slave Mode Protocol Error condition interrupt enabled 0 = Slave Mode Protocol Error condition interrupt is masked
4	TXUEM	**Transmit Underrun Error mask** 1 = Tx Underrun condition interrupt enabled 0 = Tx Underrun condition interrupt is masked
3	TXIM	**Transmit FIFO Interrupt mask (From SSP IMSC:TXIM bit)** 1 = Tx FIFO half empty or less condition interrupt enabled 0 = Tx FIFO half empty or less condition interrupt is masked
2	RXIM	**Receive FIFO Interrupt mask (From SSP IMSC:RXIM bit)** 1 = Rx FIFO half full or more condition interrupt enabled 0 = Rx FIFO half full or more condition interrupt is masked
1	RTIM	**Receive Timeout Interrupt mask (From SSP IMSC:RTIM bit)** 1 = Rx FIFO not empty and no read prior to timeout period interrupt is enabled 0 = Rx FIFO not empty and no read prior to timeout period interrupt is masked
0	RORIM	**Receive Overrun Interrupt mask (From SSP IMSC:RORIM bit)** 1 = Rx FIFO written to while full condition interrupt is enabled 0 = Rx FIFO written to while full condition interrupt is masked

10.2.2.4 Raw Interrupt Status Register (RIS)

This register provides the current raw status value of the corresponding interrupt prior to masking. Writing has no effect. For each bit, 1 = TRUE and 0 = FALSE.

The SSPPERIS, ECPERIS and TXUERIS interrupts are set as soon as the given error conditions are met (a rising edge on the error detection logic). Once cleared by a write to the appropriate ICR bit, the interrupt bit will not be set again until a new error has been detected (the next rising edge on the error detection logic). This prevents the interrupt from being immediately re-set for the same error. The condition causing the error must be resolved and asserted again to trigger a new interrupt.

Table 10-9. RIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									SSPPERIS	ECPERIS	TXUERIS	TXRIS	RXRIS	RTRIS	RORRIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC8000 + 0x00C															

Table 10-10. RIS Register Definitions

BITS	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	SSPPERIS	**SSP Protocol Error raw interrupt status** Indicates that the SSP is configured for a data size other than 16 bits. Applies only to I^2S transactions. Applies to both Slave and Master Mode operation.
5	ECPERIS	**External Codec Protocol Error raw interrupt status** Indicates that the external CODEC (the source of the frame input in slave mode) is configured for a data size other than 16 bits. Applies only to slave mode I^2S transactions.
4	TXUERIS	**Transmit Underrun Error raw interrupt status** Transmission has begun while the transmit FIFO is empty. Applies to SSP and slave mode I^2S transactions.
3	TXRIS	**Transmit FIFO raw interrupt status (from SSP RIS:TXRIS bit)** Gives the raw interrupt state (prior to masking) of the Transmit FIFO interrupt.
2	RXRIS	**Receive FIFO raw interrupt status (from SSP RIS:RXRIS bit)** Gives the raw interrupt state (prior to masking) of the Receive FIFO interrupt.
1	RTRIS	**Receive timeout raw interrupt status (from SSP RIS:RTRIS bit)** Gives the raw interrupt state (prior to masking) of the Receive Timeout interrupt
0	RORRIS	**Receive overrun raw interrupt status (from SSP RIS:RORRIS bit)** Gives the raw interrupt state (prior to masking) of the Receive Overrun interrupt

10.2.2.5 Masked Interrupt Status Register (MIS)

This register provides the current masked status value of the corresponding interrupt. Writing has no effect; all bits are read only. For each bit, 1 = TRUE and 0 = FALSE.

Table 10-11. MIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									SSPPEMIS	ECPEMIS	TXUEMIS	TXMIS	RXMIS	RTMIS	RORMIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC8000 + 0x010															

Table 10-12. MIS Register Definitions

BITS	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	SSPPEMIS	**SSP Protocol Error masked interrupt status** Gives the Master Mode Protocol Error masked interrupt state.
5	ECPEMIS	**External Codec Protocol Error masked interrupt status** Gives the Slave Mode Protocol Error masked interrupt state
4	TXUEMIS	**Transmit Underrun Error masked interrupt status** Gives the Transmit Underrun Error masked interrupt state.
3	TXMIS	**Transmit FIFO masked interrupt status (from SSP MIS:TXMIS bit)** Gives the Transmit FIFO masked interrupt state.
2	RXMIS	**Receive FIFO masked interrupt status (from SSP MIS:RXMIS bit)** Gives the Receive FIFO masked interrupt state.
1	RTMIS	**Receive timeout masked interrupt status (from SSP MIS:RTMIS bit)** Gives the Receive Timeout masked interrupt state.
0	RORMIS	**Receive overrun masked interrupt status (from SSP MIS:RORMIS bit)** Gives the Receive Overrun masked interrupt state.

10.2.2.6 Interrupt Clear Register (ICR)

This register is write only. Writing 1 causes the corresponding interrupt to be cleared. Writing 0 has no effect. The value written cannot be read back.

Table 10-13. ICR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									SSPPEC	ECPEC	TXUEC	///			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	RO	RO	RO	RO
ADDR	0xFFFC8000 + 0x014															

Table 10-14. ICR Register Definitions

BITS	NAME	DESCRIPTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	SSPPEC	**SSP Protocol Error interrupt clear** Clears the SSP Protocol Error Interrupt.
5	ECPEC	**External Codec Protocol Error interrupt clear** Clears the External Codec Protocol Error Interrupt.
4	TXUEC	**Transmit Underrun Error interrupt clear** Clears the Transmit Underrun Error Interrupt.
3:0	///	**Reserved** (see SSP ICR register)

Chapter 11
I/O Configuration

The I/O configuration controls:

- Pin Muxing: Provides registers to program the pin muxing on the device.
- Pull-up/Pull-down Resistors: Provides registers to control the pull-up and pull-down resistors on certain pins of the chip.

11.1 Theory of Operation

The I/O Configuration (IOCON) is an AMBA slave block that connects to the APB. It provides registers for programming pin muxing and for controlling the pull-up and pull-down resistors on certain pins of the chip.

These registers should be programmed before any others so that pins are correctly configured before enabling functional blocks that use those pins.

11.2 Register Reference

This section describes the registers used in I/O configuration. In all cases, when the MUX register is programmed its corresponding Resistor register (if it exists) must be programmed. The Resistor registers are not automatically configured.

11.2.1 Memory Map

The base address for the IOCON is 0xFFFE5000. Table 11-1 shows the register locations in the memory map.

Table 11-1. IOCON Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	MUXCTL1	**Muxing Control 1** For pins from PI2/ETHERCOL to PL0/LCDVD14
0x04	RESCTL1	**Resistor Control 1** Assignment for pins From PI2/ETHERCOL to PL0/LCDVD14
0x08	///	**Reserved** Do not access
0x0C	///	**Reserved** Do not access
0x10	MUXCTL3	**Muxing Control 3** For pin INT4/CTCLK/BATCNTL
0x14	RESCTL3	**Resistor Control 3** Assignment for pin INT4/CTCLK/BATCNTL
0x18	MUXCTL4	**Muxing Control 4** For PA7/CTCAP2B/CTCMP2B/SCL to PA2/CTCAP0A/ CTCMP0A
0x1C	RESCTL4	**Resistor Control 4** Assignment for pins from PA7/CTCAP2B/CTCMP2B/SCL to PA2/CTCAP0A/CTCMP0A
0x20	MUXCTL5	**Muxing Control 5** For pins from PA1/INT3/UARTTX2/UARTIRTX2 to PB2/SSPFRM
0x24	RESCTL5	**Resistor Control 5** Assignment for pins from PA1/INT3/UARTTX2/UARTIRTX2 to PB2/SSPFRM
0x28	MUXCTL6	**Muxing Control 6** For pins from PB1/DREQ/nUARTRTS0 to PB0/nDACK/nUARTCTS0
0x2C	RESCTL6	**Resistor Control 6** Assignment for pins from PB1/DREQ/nUARTRTS0 to PB0/nDACK/nUARTCTS0
0x30	MUXCTL7	**Muxing Control 7** For pins from PC7/A23 to PC0/A16
0x34	RESCTL7	**Resistor Control 7** Assignment for pins from PC7/A23 to PC0/A16
0x38	///	**Reserved** Do not access
0x3C	///	**Reserved** Do not access
0x40	///	**Reserved** Do not access
0x44	///	**Reserved** Do not access
0x48	MUXCTL10	**Muxing Control 10** For pins from PN3/D25 to PK5/D21
0x4C	RESCTL10	**Resistor Control 10** Assignment for pins from PN3/D25 to PK5/D21
0x50	MUXCTL11	**Muxing Control 11** For pins from PD4/D12 to PD1/D9
0x54	RESCTL11	**Resistor Control 11** Assignment for pins from PD4/D12 to PD1/D9
0x58	MUXCTL12	**Muxing Control 12** For pins from PK0/D16 to PD0/D8
0x5C	RESCTL12	**Resistor Control 12** Assignment for pins from PK0/D16 to D2
0x60	///	**Reserved** Do not access
0x64	RESCTL13	**Resistor Control 13** Assignment for pins from D1 to D0
0x68	MUXCTL14	**Muxing Control 14** For Pins Ranging From nCS3/PM3 to nBLE3/PM5
0x6C	///	**Reserved** Do not access

Table 11-1. IOCON Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x70	MUXCTL15	**Muxing Control 15** For Pin nBLE0/PM4
0x74	RESCTL15	**Resistor Control 15** Assignment for pin nBLE0/PM4
0x78	///	**Reserved** Do not access
0x7C	///	**Reserved** Do not access
0x80	///	**Reserved** Do not access
0x84	RESCTL17	**Resistor Control 17** Assignment for pin SDCLK
0x88	///	**Reserved** Do not access
0x8C	///	**Reserved** Do not access
0x90	MUXCTL19	**Muxing Control 19** For pins From PE7/nWAIT/nDEOT to PL5/D29
0x94	RESCTL19	**Resistor Control 19** Assignment for pins from PE7/nWAIT/nDEOT to PL5/D29
0x98	MUXCTL20	**Muxing Control 20** For pins from PE2/LCDPS to PF6/LCDEN/LCDSPL
0x9C	RESCTL20	**Resistor Control 20** Assignment for pins from PE2/LCDPS to PF6/LCDEN/LCDSPL
0xA0	MUXCTL21	**Muxing Control 21** For pins from PF5/LCDVD11 to PF2/LCDVD8
0xA4	RESCTL21	**Resistor Control 21** Assignment for pins from PF5/LCDVD11 to PF2/LCDVD8
0xA8	MUXCTL22	**Muxing Control 22** For pins from PF1/LCDVD7 to PG2/LCDVD0
0xAC	RESCTL22	**Resistor Control 22** Assignment for pins from PF1/LCDVD7 to PG2/LCDVD0
0xB0	MUXCTL23	**Muxing Control 23** For pins from PG1/ETHERTXCLK to PH2/ETHERRXCLK
0xB4	RESCTL23	**Resistor Control 23** Assignment for pins from PG1/ETHERTXCLK to PH2/ETHERRXCLK
0xB8	MUXCTL24	**Muxing Control 24** For pins from PH1/ETHERRXDV to PI3/ETHERCRS
0xBC	RESCTL24	**Resistor Control 24** Assignment for pins from PH1/ETHERRXDV to PI3/ETHERCRS
0xC0	MUXCTL25	**Muxing Control 25** For pins from AN6/PJ7/INT7 to AN3/LR/Y-/PJ0
0xC4	///	**Reserved** Do not access

11.2.2 Register Definitions

11.2.2.1 Multiplexing Control 1 Register (MUXCTL1)

This Register allows software to configure pins PI2/ETHERCOL through PL0/LCDVD14. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-2. MUXCTL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						PI2		PI1		PI0		PL1		PL0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x00															

Table 11-3. MUXCTL1 Fields

BIT	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9:8	PI2	**PI2/ETHERCOL Assignment** 00 = PI2 01 = ETHERCOL 10 = Reserved 11 = Reserved
7:6	PI1	**PI1/ETHERMDIO Assignment** 00 = PI1 01 = ETHERMDIO 10 = Reserved 11 = Reserved
5:4	PI0	**PI0/ETHERMDC Assignment** 00 = PI0 01 = ETHERMDC 10 = Reserved 11 = Reserved
3:2	PL1	**PL1/LCDVD15 Assignment (LH79524 Only)** 00 = PL1 01 = LCDVD15 10 = Reserved 11 = Reserved
1:0	PL0	**PL0/LCDVD14 Assignment (LH79524 Only)** 00 = PL0 01 = LCDVD14 10 = Reserved 11 = Reserved

11.2.2.2 Resistor Configuration Control 1 Register (RESCTL1)

The RESCTL1 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-4. RESCTL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						PI2		PI1		PI0		PL1		PL0	
RESET	0	0	0	0	0	0	0	1	1	0	0	1	0	1	0	1
RW	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x04															

Table 11-5. RESCTL1 Fields

BIT	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9:8	PI2	**Pin PI2/ETHERCOL Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PI1	**Pin PI1/ETHERMDIO Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
5:4	PI0	**Pin PI0/ETHERMDC Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PL1	**Pin PL1/LCDVD15 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PL0	**Pin PL0/LCDVD14 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.3 Multiplexing Control 3 Register (MUXCTL3)

The MUXCTL3 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-6. MUXCTL3 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														INT4	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE5000 + 0x10															

Table 11-7. MUXCTL3 Fields

BIT	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	INT4	**CTCLK/INT4/BATCNTL Pin Assignment** 00 = CTCLK 01 = INT4 10 = BATCNTL 11 = Reserved

11.2.2.4 Resistor Configuration Control 3 Register (RESCTL3)

The RESCTL3 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-8. RESCTL3 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														INT4	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE5000 + 0x14															

Table 11-9. RESCTL3 Fields

BIT	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	INT4	**Pin INT4/CTCLK/BATCNTL Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.5 Multiplexing Control 4 Register (MUXCTL4)

The MUXCTL4 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-10. MUXCTL4 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				PA7		PA6		PA5		PA4		PA3		PA2	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x18															

Table 11-11. MUXCTL4 Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11:10	PA7	**PA7/CTCAP2B/CTCMP2B/SCL Assignment** 00 = PA7 01 = CTCAP2B 10 = CTCMP2B 11 = SCL
9:8	PA6	**PA6/CTCAP2A/CTCMP2A/SDA Assignment** 00 = PA6 01 = CTCAP2A 10 = CTCMP2A 11 = SDA
7:6	PA5	**PA5/CTCAP1B/CTCMP1B Assignment** 00 = PA5 01 = CTCAP1B 10 = CTCMP1B 11 = Reserved
5:4	PA4	**PA4/CTCAP1A/CTCMP1A Assignment** 00 = PA4 01 = CTCAP1A 10 = CTCMP1A 11 = Reserved
3:2	PA3	**PA3/CTCAP0B/CTCMP0B Assignment** 00 = PA3 01 = CTCAP0B 10 = CTCMP0B 11 = Reserved
1:0	PA2	**PA2/CTCAP0A/CTCMP0A Assignment** 00 = PA2 01 = CTCAP0A 10 = CTCMP0A 11 = Reserved

11.2.2.6 Resistor Configuration Control 4 Register (RESCTL4)

The RESCTL4 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-12. RESCTL4 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				PA7		PA6		PA5		PA4		PA3		PA2	
RESET	0	0	0	0	1	0	1	0	0	1	0	1	0	1	0	1
RW	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x1C															

Table 11-13. RESCTL4 Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11:10	PA7	**Pin PA7/CTCAP2B/CTCMP2B/SCL Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PA6	**Pin PA6/CTCAP2A/CTCMP2A/SDA Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PA5	**Pin PA5/CTCAP1B/CTCMP1B Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
5:4	PA4	**Pin PA4/CTCAP1A/CTCMP1A Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PA3	**Pin PA3/CTCAP0B/CTCMP0B Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PA2	**Pin PA2/CTCAP0A/CTCMP0A Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.7 Multiplexing Control 5 Register (MUXCTL5)

The MUXCTL5 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-14. MUXCTL5 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PA1		PA0		PB7		PB6		PB5		PB4		PB3		PB2	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x20															

Table 11-15. MUXCTL5 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PA1	**PA1/INT3/UARTTX2/UARTIRTX2 Assignment** 00 = PA1 01 = INT3 10 = UARTTX2 11 = UARTIRTX2
13:12	PA0	PA0/INT2/UARTRX2/UARTIRRX2 Assignment 00 = PA0 01 = INT2 10 = UARTRX2 11 = UARTIRRX2
11:10	PB7	PB7/INT1/UARTTX0/UARTIRTX0 Assignment 00 = PB7 01 = INT1 10 = UARTTX0 11 = UARTIRTX0
9:8	PB6	PB6/INT0/UARTRX0/UARTIRRX0 Assignment 00 = PB6 01 = INT0 10 = UARTRX0 11 = UARTIRRX0
7:6	PB5	PB5/SSPTX/UARTTX1/UARTIRTX1 Assignment 00 = PB5 01 = SSPTX 10 = UARTTX1 11 = UARTIRTX1
5:4	PB4	PB4/SSPRX/UARTRX1/UARTIRRX1 Assignment 00 = PB4 01 = SSPRX 10 = UARTRX1 11 = UARTIRRX1
3:2	PB3	PB3/SSPCLK Assignment 00 = PB3 01 = SSPCLK 10 = Reserved 11 = Reserved
1:0	PB2	PB2/SSPFRM Assignment 00 = PB2 01 = SSPFRM 10 = Reserved 11 = Reserved

11.2.2.8 Resistor Configuration Control 5 Register (RESCTL5)

The RESCTL5 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-16. RESCTL5 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PA1		PA0		PB7		PB6		PB5		PB4		PB3		PB2	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x24															

Table 11-17. RESCTL5 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PA1	**Pin PA1/INT3/UARTTX2/UARTIRTX2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PA0	**Pin PA0/INT2/UARTRX2/UARTIRRX2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PB7	**Pin PB7/INT1/UARTTX0/UARTIRTX0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PB6	**Pin PB6/INT0/UARTRX0/UARTIRRX0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PB5	**Pin PB5/SSPTX/UARTTX1/UARTIRTX1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-17. RESCTL5 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PB4	**Pin PB4/SSPRX/UARTRX1/UARTIRRX1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PB3	**Pin PB3/SSPCLK Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PB2	**Pin PB2/SSPFRM Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.9 Multiplexing Control 6 Register (MUXCTL6)

The MUXCTL6 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-18. MUXCTL6 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												PB1		PB0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x28															

Table 11-19. MUXCTL6 Fields

BIT	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:2	PB1	**PB1/DREQ/nUARTRTS0 Assignment** 00 = PB1 01 = DREQ 10 = nUARTRTS0 11 = Reserved
1:0	PB0	**PB0/nDACK/nUARTCTS0 Assignment** 00 = PB0 01 = nDACK 10 = nUARTCTS0 11 = Reserved

11.2.2.10 Resistor Configuration Control 6 Register (RESCTL6)

The RESCTL6 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-20. RESCTL6 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												PB1		PB0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x2C															

Table 11-21. RESCTL6 Fields

BIT	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:2	PB1	**Pin PB1/DREQ/nUARTRTS0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PB0	**Pin PB0/nDACK/nUARTCTS0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.11 Multiplexing Control 7 Register (MUXCTL7)

The MUXCTL7 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-22. MUXCTL7 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PC7		PC6		PC5		PC4		PC3		PC2		PC1		PC0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x30															

Table 11-23. MUXCTL7 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PC7	**PC7/A23/nFRE Assignment** 00 = PC7 01 = A23 10 = nFRE 11 = Reserved
13:12	PC6	**PC6/A22/nFWE Assignment** 00 = PC6 01 = A22 10 = nFWE 11 = Reserved
11:10	PC5	**PC5/A21 Assignment** 00 = PC5 01 = A21 10 = Reserved 11 = Reserved
9:8	PC4	**PC4/A20 Assignment** 00 = PC4 01 = A20 10 = Reserved 11 = Reserved
7:6	PC3	**PC3/A19 Assignment** 00 = PC3 01 = A19 10 = Reserved 11 = Reserved

Table 11-23. MUXCTL7 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PC2	**PC2/A18 Assignment** 00 = PC2 01 = A18 10 = Reserved 11 = Reserved
3:2	PC1	**PC1/A17 Assignment** 00 = PC1 01 = A17 10 = Reserved 11 = Reserved
1:0	PC0	**PC0/A16 Assignment** 00 = PC0 01 = A16 10 = Reserved 11 = Reserved

11.2.2.12 Resistor Configuration Control 7 Register (RESCTL7)

The RESCTL7 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-24. RESCTL7 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PC7		PC6		PC5		PC4		PC3		PC2		PC1		PC0	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x34															

Table 11-25. RESCTL7 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PC7	**Pin PC7/A23/nFRE Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PC6	**Pin PC6/A2nFWE Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PC5	**Pin PC5/A21 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PC4	**Pin PC4/A20 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PC3	**Pin PC3/A19 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-25. RESCTL7 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PC2	**Pin PC2/A18 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PC1	**Pin PC1/A17 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PC0	**Pin PC0/A16 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.13 Multiplexing Control 10 Register (MUXCTL10)

The MUXCTL10 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-26. MUXCTL10 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PN3		PN2		PD7		PK7		PD6		PK6		PD5		PK5	
LH79525 RESET	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
LH79524 RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x48															

Table 11-27. MUXCTL10 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PN3	**PN3/D25 Assignment (LH79524 Only)** 00 = PN3 01 = D25 10 = Reserved 11 = Reserved
13:12	PN2	**PN2/D24 Assignment (LH79524 Only)** 00 = PN2 01 = D24 10 = Reserved 11 = Reserved
11:10	PD7	**PD7/D15 Assignment** 00 = PD7 01 = D15 10 = Reserved 11 = Reserved
9:8	PK7	**PK7/D23 Assignment (LH79524 Only)** 00 = PK7 01 = D23 10 = Reserved 11 = Reserved
7:6	PD6	**PD6/D14 Assignment** 00 = PD6 01 = D14 10 = Reserved 11 = Reserved

Table 11-27. MUXCTL10 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PK6	**PK6/D22 Assignment (LH79524 Only)** 00 = PK6 01 = D22 10 = Reserved 11 = Reserved
3:2	PD5	**PD5/D13 Assignment** 00 = PD5 01 = D13 10 = Reserved 11 = Reserved
1:0	PK5	**PK5/D21 Assignment (LH79524 Only)** 00 = PK5 01 = D21 10 = Reserved 11 = Reserved

11.2.2.14 Resistor Configuration Control 10 Register (RESCTL10)

The RESCTL10 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-28. RESCTL10 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PN3		PN2		PD7		PK7		PD6		PK6		PD5		PK5	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x4C															

Table 11-29. RESCTL10 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PN3	**Pin PN3/D25 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PN2	**Pin PN2/D24 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PD7	**Pin PD7/D15 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PK7	**Pin PK7/D23 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PD6	**Pin PD6/D14 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-29. RESCTL10 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PK6	**Pin PK6/D22 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PD5	**Pin PD5/D13 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PK5	**Pin PK5/D21 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.15 Multiplexing Control 11 Register (MUXCTL11)

The MUXCTL11 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-30. MUXCTL11 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PD4		PK4		PD3		PK3		PD2		PK2		PK1		PD1	
RESET 8-Bit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RESET 16-Bit	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1
RESET 32-Bit	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x50															

Table 11-31. MUXCTL11 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PD4	**PD4/D12 Assignment** 00 = PD4 01 = D12 10 = Reserved 11 = Reserved
13:12	PK4	**PK4/D20 Assignment** (LH79524 Only) 00 = PK4 01 = D20 10 = Reserved 11 = Reserved
11:10	PD3	**PD3/D11 Assignment** 00 = PD3 01 = D11 10 = Reserved 11 = Reserved
9:8	PK3	**PK3/D19 Assignment** (LH79524 Only) 00 = PK3 01 = D19 10 = Reserved 11 = Reserved

Table 11-31. MUXCTL11 Fields (Cont'd)

BIT	NAME	DESCRIPTION
7:6	PD2	**PD2/D10 Assignment** 00 = PD2 01 = D10 10 = Reserved 11 = Reserved
5:4	PK2	**PK2/D18 Assignment** (LH79524 Only) 00 = PK2 01 = D18 10 = Reserved 11 = Reserved
3:2	PK1	**PK1/D17 Assignment** (LH79524 Only) 00 = PK1 01 = D17 10 = Reserved 11 = Reserved
1:0	PD1	**PD1/D9 Assignment** 00 = PD1 01 = D9 10 = Reserved 11 = Reserved

11.2.2.16 Resistor Configuration Control 11 Register (RESCTL11)

The RESCTL11 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-32. RESCTL11 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PD4		PK4		PD3		PK3		PD2		PK2		PK1		PD1	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x54															

Table 11-33. RESCTL11 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PD4	**Pin PD4/D12 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PK4	**Pin PK4/D20 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PD3	**Pin PD3/D11 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PK3	**Pin PK3/D19 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PD2	**Pin PD2/D10 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-33. RESCTL11 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PK2	**Pin PK2/D18 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PK1	**Pin PK1/D17 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PD1	**Pin PD1/D9 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.17 Multiplexing Control 12 Register (MUXCTL12)

The MUXCTL12 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-34. MUXCTL12 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PK0		PD0		///											
RESET 8-Bit	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RESET 16-Bit	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
RESET 32-Bit	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x58															

Table 11-35. MUXCTL12 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PK0	**PK0/D16 Assignment (LH79524 Only)** 00 = PK0 01 = D16 10 = Reserved 11 = Reserved
13:12	PD0	**PD0/D8 Assignment** 00 = PD0 01 = D8 10 = Reserved 11 = Reserved
11:0	///	**Reserved** Reading returns 0. Write the reset value.

11.2.2.18 Resistor Configuration Control 12 Register (RESCTL12)

The RESCTL12 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-36. RESCTL12 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PK0		PD0		D7		D6		D5		D4		D3		D2	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x5C															

Table 11-37. RESCTL12 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PK0	**Pin PK0/D16 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PD0	**Pin PD0/D8 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	D7	**Pin PD7 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	D6	**Pin PD6 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	D5	**Pin PD5 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-37. RESCTL12 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	D4	**Pin PD4 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	D3	**Pin PD3 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	D2	**Pin PD2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.19 Resistor Configuration Control 13 Register (RESCTL13)

The RESCTL13 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-38. RESCTL13 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												D1		D0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x64															

Table 11-39. RESCTL13 Fields

BIT	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:2	D1	**Pin D1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	D0	**Pin D0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.20 Multiplexing Control 14 Register (MUXCTL14)

The MUXCTL14 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-40. MUXCTL14 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	nCS3		nCS2		nCS1		nCS0		///		nBLE3		nBLE2		nBLE1	
LH79525 RESET	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
LH79524 RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x68															

Table 11-41. MUXCTL14 Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
15:14	nCS3	**nCS3/PM3 Assignment** 00 =nCS3 01 = PM3 10 = Reserved 11 = Reserved
13:12	nCS2	**nCS2/PM2 Assignment** 00 =nCS2 01 = PM2 10 = Reserved 11 = Reserved
11:10	nCS1	**nCS1/PM1 Assignment** 00 =nCS1 01 = PM1 10 = Reserved 11 = Reserved
9:8	nCS0	**nCS0/PM0 Assignment** 00 =nCS0 01 = PM0 10 = Reserved 11 = Reserved
7:6	///	**Reserved** Reading returns 0. Write the reset value.
5:4	nBLE3	**nBLE3/PM7 Assignment** 00 =nBLE3 01 = PM7 10 = Reserved 11 = Reserved

Table 11-41. MUXCTL14 Fields (Cont'd)

BIT	NAME	DESCRIPTION
3:2	nBLE2	**nBLE2/PM6 Assignment** 00 =nBLE2 01 = PM6 10 = Reserved 11 = Reserved
1:0	nBLE1	**nBLE3/PM5 Assignment** 00 =nBLE1 01 = PM5 10 = Reserved 11 = Reserved

11.2.2.21 Multiplexing Control 15 Register (MUXCTL15)

The MUXCTL15 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-42. MUXCTL15 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														nBLE0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE5000 + 0x70															

Table 11-43. MUXCTL15 Fields

BIT	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	nBLE0	**nBLE0/PM4 Assignment** 00 =nBLE0 01 = PM4 10 = Reserved 11 = Reserved

11.2.2.22 Resistor Configuration Control 15 Register (RESCTL15)

The RESCTL15 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-44. RESCTL15 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														nBLE0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE5000 + 0x74															

Table 11-45. RESCTL15 Fields

BIT	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	nBLE0	**Pin nBLE0/PM4 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.23 Resistor Configuration Control 17 Register (RESCTL17)

The RESCTL17 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-46. RESCTL17 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														SDCLK	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE5000 + 0x84															

Table 11-47. RESCTL17 Fields

BIT	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	SDCLK	**Pin SDCLK Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.24 Multiplexing Control 19 Register (MUXCTL19)

The MUXCTL19 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-48. MUXCTL19 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PE7		PE6		PL7		PE5		PL6/D30		PE4		PE3		PL5	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RESET 32-Bit	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x90															

Table 11-49. MUXCTL19 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PE7	**PE7/nWAIT/nDEOT Assignment** 00 = PE7 01- = nWAIT 10 = nDEOT 11 = Reserved
13:12	PE6	**PE6/LCDVEEN/LCDMOD Assignment** 00 = PE6 01 = LCDVEEN 10 = LCDMOD 11 = Reserved
11:10	PL7	**PL7/D31 Assignment (LH79524 Only)** 00 = PL7 01 = D31 10 = Reserved 11 = Reserved
9:8	PE5	**PE5/LCDVDDEN Assignment** 00 = PE5 01 = LCDVDDEN 10 = Reserved 11 = Reserved
7:6	PL6	**PL6/D30 Assignment (LH79524 Only)** 00 = PL6 01 = D30 10 = Reserved 11 = Reserved

Table 11-49. MUXCTL19 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PE4	**PE4/LCDSPLEN/LCDREV Assignment** 00 = PE4 01 = LCDSPLEN 10 = LCDREV 11 = Reserved
3:2	PE3	**PE3/LCDCLS Assignment** 00 = PE3 01 = LCDCLS 10 = Reserved 11 = Reserved
1:0	PL5	**PL5/D29 Assignment (LH79524 Only)** 00 = PL5 01 = D29 10 = Reserved 11 = Reserved

11.2.2.25 Resistor Configuration Control 19 Register (RESCTL19)

The RESCTL19 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-50. RESCTL19 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PE7		PE6		PL7		PE5		PL6		PE4		PE3		PL5	
RESET	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x94															

Table 11-51. RESCTL19 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PE7	**Pin PE7/nWAIT/nDEOT Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PE6	**Pin PE6/LCDVEEN/LCDMOD Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PL7	**Pin PL7/D31 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PE5	**Pin PE5/LCDVDDEN Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PL6	**Pin PL6/D30 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-51. RESCTL19 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PE4	**Pin PE4/LCDSPLEN/LCDREV Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PE3	**Pin PE3/LCDCLS Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PL5	**Pin PL5/D29 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.26 Multiplexing Control 20 Register (MUXCTL20)

The MUXCTL20 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-52. MUXCTL20 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PE2		PL4		PE1		PN1		PE0		PN0		PF7		PF6	
LH79525 RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
LH79524 RESET	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x98															

Table 11-53. MUXCTL20 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PE2	**PE2/LCDPS Assignment** 00 = PE2 01 = LCDPS 10 = Reserved 11 = Reserved
13:12	PL4	**PL4/D28 Assignment (LH79524 Only)** 00 = PL4 01 = D28 10 = Reserved 11 = Reserved
11:10	PE1	**PE1/LCDDCLK Assignment** 00 = PE1 01 = LCDDCLK 10 = Reserved 11 = Reserved
9:8	PN1	**PN1/D27 Assignment (LH79524 Only)** 00 = PN1 01 = D27 10 = Reserved 11 = Reserved
7:6	PE0	**PE0/LCDLP/LCDHRLP Assignment** 00 = PE0 01 = LCDLP 10 = LCDHRLP 11 = Reserved

Table 11-53. MUXCTL20 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PN0	**PN0/D26 Assignment (LH79524 Only)** 00 = PN0 01 = D26 10 = Reserved 11 = Reserved
3:2	PF7	**PF7/LCDFP/LCDSPS Assignment** 00 = PF6 01 = LCDFP 10 = LCDSPS 11 = Reserved
1:0	PF6	**PF6/LCDEN/LCDSPL Assignment** 00 = PF6 01 = LCDEN 10 = LCDSPL 11 = Reserved

11.2.2.27 Resistor Configuration Control 20 Register (RESCTL20)

The RESCTL20 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-54. RESCTL20 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PE2		PL4		PE1		PN1		PE0		PN0		PF7		PF6	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0x96															

Table 11-55. RESCTL20 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PE2	**Pin PE2/LCDPS Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PL4	**Pin PL4/D28 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PE1	**Pin PE1/LCDDCLK Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PN1	**Pin PN1/D27 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PE0	**Pin PE0/LCDLP/LCDHRLP Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-55. RESCTL20 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PN0	**Pin PN0/D26 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PF7	**Pin PF7/LCDFP/LCDSPS Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PF6	**Pin PF6/LCDEN/LCDSPL Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.28 Multiplexing Control 21 Register (MUXCTL21)

The MUXCTL21 Register allows software to configure a number of LH79524/LH79525 pins. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-56. MUXCTL21 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				PF5		PL3		PF4		PL2		PF3		PF2	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xA0															

Table 11-57. MUXCTL21 Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11:10	PF5	**PF5/LCDVD11 Assignment** 00 = PF5 01 = LCDVD11 10 = Reserved 11 = Reserved
9:8	PL3	**PL3/LCDVD13 Assignment (LH79524 Only)** 00 = PL3 01 = LCDVD13 10 = Reserved 11 = Reserved
7:6	PF4	**PF4/LCDVD10 Assignment** 00 = PF4 01 = LCDVD10 10 = Reserved 11 = Reserved
5:4	PL2	**PL2/LCDVD12 Assignment (LH79524 Only)** 00 = PL2 01 = LCDVD12 10 = Reserved 11 = Reserved
3:2	PF3	**PF3/LCDVD9 Assignment** 00 = PF3 01 = LCDVD9 10 = Reserved 11 = Reserved
1:0	PF2	**PF2/LCDVD8 Assignment** 00 = PF2 01 = LCDVD8 10 = Reserved 11 = Reserved

11.2.2.29 Resistor Configuration Control 21 Register (RESCTL21)

The RESCTL21 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors. Bits marked 'LH79524 Only' read as 0 with all writes 'reserved' on the LH79525.

Table 11-58. RESCTL21 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				PF5		PL3		PF4		PL2		PF3		PF2	
RESET	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1
RW	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xA4															

Table 11-59. RESCTL21 Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11:10	PF5	**Pin PF5/LCDVD11 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PL3	**Pin PL3/LCDVD13 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PF4	**Pin PF4/LCDVD10 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
5:4	PL2	**Pin PL2/LCDVD12 Resistor Assignment (LH79524 Only)** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PF3	**Pin PF3/LCDVD9 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PF2	**Pin PF2/LCDVD8 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.30 Multiplexing Control 22 Register (MUXCTL22)

The MUXCTL22 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-60. MUXCTL22 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PF1		PF0		PG7		PG6		PG5		PG4		PG3		PG2	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xA8															

Table 11-61. MUXCTL22 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PF1	**PF1/LCDVD7 Assignment** 00 = PF1 01 = LCDVD7 10 = Reserved 11 = Reserved
13:12	PF0	**PF0/LCDVD6 Assignment** 00 = PF0 01 = LCDVD6 10 = Reserved 11 = Reserved
11:10	PG7	**PG7/LCDVD5 Assignment** 00 = PG7 01 = LCDVD5 10 = Reserved 11 = Reserved
9:8	PG6	**PG6/LCDVD4 Assignment** 00 = PG6 01 = LCDVD4 10 = Reserved 11 = Reserved
7:6	PG5	**PG5/LCDVD3 Assignment** 00 = PG5 01 = LCDVD3 10 = Reserved 11 = Reserved

Table 11-61. MUXCTL22 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PG4	**PG4/LCDVD2 Assignment** 00 = PG4 01 = LCDVD2 10 = Reserved 11 = Reserved
3:2	PG3	**PG3/LCDVD1 Assignment** 00 = PG3 01 = LCDVD1 10 = Reserved 11 = Reserved
1:0	PG2	**PG2/LCDVD0 Assignment** 00 = PG2 01 = LCDVD0 10 = Reserved 11 = Reserved

11.2.2.31 Resistor Configuration Control 22 Register (RESCTL22)

The RESCTL22 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-62. RESCTL22 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PF1		PF0		PG7		PG6		PG5		PG4		PG3		PG2	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xAC															

Table 11-63. RESCTL22 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PF1	**Pin PF1/LCDVD7 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PF0	**Pin PF0/LCDVD6 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PG7	**Pin PG7/LCDVD5 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PG6	**Pin PG6/LCDVD4 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PG5	**Pin PG5/LCDVD3 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-63. RESCTL22 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PG4	**Pin PG4/LCDVD2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PG3	**Pin PG3/LCDVD1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PG2	**Pin PG2/LCDVD0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.32 Multiplexing Control 23 Register (MUXCTL23)

The MUXCTL23 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-64. MUXCTL23 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PG1		PG0		PH7		PH6		PH5		PH4		PH3		PH2	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xB0															

Table 11-65. MUXCTL23 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PG1	**PG1/ETHERTXCLK Assignment** 00 = PG1 01 = ETHERTXCLK 10 = Reserved 11 = Reserved
13:12	PG0	**PG0/ETHERTXEN Assignment** 00 = PG0 01 = ETHERTXEN 10 = Reserved 11 = Reserved
11:10	PH7	**PH7/ETHERTX3 Assignment** 00 = PH7 01 = ETHERTX3 10 = Reserved 11 = Reserved
9:8	PH6	**PH6/ETHERTX2 Assignment** 00 = PH6 01 = ETHERTX2 10 = Reserved 11 = Reserved
7:6	PH5	**PH5/ETHERTX1 Assignment** 00 = PH5 01 = ETHERTX1 10 = Reserved 11 = Reserved

Table 11-65. MUXCTL23 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PH4	**PH4/ETHERTX0 Assignment** 00 = PH4 01 = ETHERTX0 10 = Reserved 11 = Reserved
3:2	PH3	**PH3/ETHERTXER Assignment** 00 = PH3 01 = ETHERTXER 10 = Reserved 11 = Reserved
1:0	PH2	**PH2/ETHERRXCLK Assignment** 00 = PH2 01 = ETHERRXCLK 10 = Reserved 11 = Reserved

11.2.2.33 Resistor Configuration Control 23 Register (RESCTL23)

The RESCTL23 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-66. RESCTL23 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PG1		PG0		PH7		PH6		PH5		PH4		PH3		PH2	
RESET	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xB4															

Table 11-67. RESCTL23 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	PG1	**Pin PG1/ETHERTXCLK Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
13:12	PG0	**Pin PG0/ETHERTXEN Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PH7	**Pin PH7/ETHERTX3 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PH6	**Pin PH6/ETHERTX2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PH5	**Pin PH5/ETHERTX1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

Table 11-67. RESCTL23 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	PH4	**Pin PH4/ETHERTX0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PH3	**Pin PH3/ETHERTXER Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PH2	**Pin PH2/ETHERRXCLK Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.34 Multiplexing Control 24 Register (MUXCTL24)

The MUXCTL24 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-68. MUXCTL24 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PH1		PH0		PI7		PI6		PI5		PI4		PI3	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xB8															

Table 11-69. MUXCTL24 Fields

BIT	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13:12	PH1	**PH1/ETHERRXDV Assignment** 00 = PH1 01 = ETHERRXDV 10 = Reserved 11 = Reserved
11:10	PH0	**PH0/ETHERRX3 Assignment** 00 = PH0 01 = ETHERRX3 10 = Reserved 11 = Reserved
9:8	PI7	**PI7/ETHERRX2 Assignment** 00 = PI7 01 = ETHERRX2 10 = Reserved 11 = Reserved
7:6	PI6	**PI6/ETHERRX1 Assignment** 00 = PI6 01 = ETHERRX1 10 = Reserved 11 = Reserved
5:4	PI5	**PI5/ETHERRX0 Assignment** 00 = PI5 01 = ETHERRX0 10 = Reserved 11 = Reserved
3:2	PI4	**PI4/ETHERRXER Assignment** 00 = PI4 01 = ETHERRXER 10 = Reserved 11 = Reserved
1:0	PI3	**PI3/ETHERCRS Assignment** 00 = PI3 01 = ETHERCRS 10 = Reserved 11 = Reserved

11.2.2.35 Resistor Configuration Control 24 Register (RESCTL24)

The RESCTL24 Register allows software to configure a number of the LH79524/LH79525 pull-up/pull-down resistors.

Table 11-70. RESCTL24 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		PH1		PH0		PI7		PI6		PI5		PI4		PI3	
RESET	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xBC															

Table 11-71. RESCTL24 Fields

BIT	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13:12	PH1	**Pin PH1/ETHERR XDV Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
11:10	PH0	**Pin PH0/ETHERRX3 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
9:8	PI7	**Pin PI7/ETHERRX2 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
7:6	PI6	**Pin PI6/ETHERRX1 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
5:4	PI5	**Pin PI5/ETHERRX0 Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
3:2	PI4	**Pin PI4/ETHERRXER Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved
1:0	PI3	**Pin PI3/ETHERCRS Resistor Assignment** 00 = No Pull-Down or Pull-Up 01 = Pull-Down 10 = Pull-Up 11 = Reserved

11.2.2.36 Multiplexing Control 25 Register (MUXCTL25)

The MUXCTL25 Register allows software to configure a number of LH79524/LH79525 pins.

Table 11-72. MUXCTL25 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	AN6		AN7		AN5		AN8		AN2		AN9		AN4		AN3	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE5000 + 0xC0															

Table 11-73. MUXCTL25 Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Writing to these BIT has no effect. Reading returns 0.
15:14	AN6	**AN6/PJ7/INT7 Assignment** 00 = AN6 01 = PJ7 10 = INT7 11 = Reserved
13:12	AN7	**AN7/PJ6/INT6 Assignment** 00 = AN7 01 = PJ6 10 = INT6 11 = Reserved
11:10	AN5	**AN5/PJ5/INT5 Assignment** 00 = AN5 01 = PJ5 10 = INT5 11 = Reserved
9:8	AN8	**AN8/PJ4 Assignment** 00 = AN8 01 = PJ4 10 = Reserved 11 = Reserved
7:6	AN2	**AN2/LL/Y+/PJ3 Assignment** 00 = AN2/LL/Y+ 01 = PJ3 10 = Reserved 11 = Reserved

Table 11-73. MUXCTL25 Fields (Cont'd)

BIT	NAME	DESCRIPTION
5:4	AN9	**AN9/PJ2 Assignment** 00 = AN9 01 = PJ2 10 = Reserved 11 = Reserved
3:2	AN4	**AN4/WIPER/PJ1 Assignment** 00 = AN4/WIPER 01 = PJ1 10 = Reserved 11 = Reserved
1:0	AN3	**AN3/LR/Y-/PJ0 Assignment** 00 = AN3/LR/Y- 01 = PJ0 10 = Reserved 11 = Reserved

Chapter 12
Real Time Clock

The Real Time Clock (RTC) can be used as a basic alarm, a long time-base counter, a time base for a true real-time clock, or as a wake-up interrupt generator to transfer the LH79524/LH79525 from Standby, Sleep, Stop1, or Stop2 mode to Active mode. The RTC can be programmed to issue an interrupt when the RTC count matches a programmed value. Figure 12-1 shows a block diagram of the RTC.

The RTC is identical in both the LH79524 and LH79525; all descriptions in this chapter apply to both MCUs.

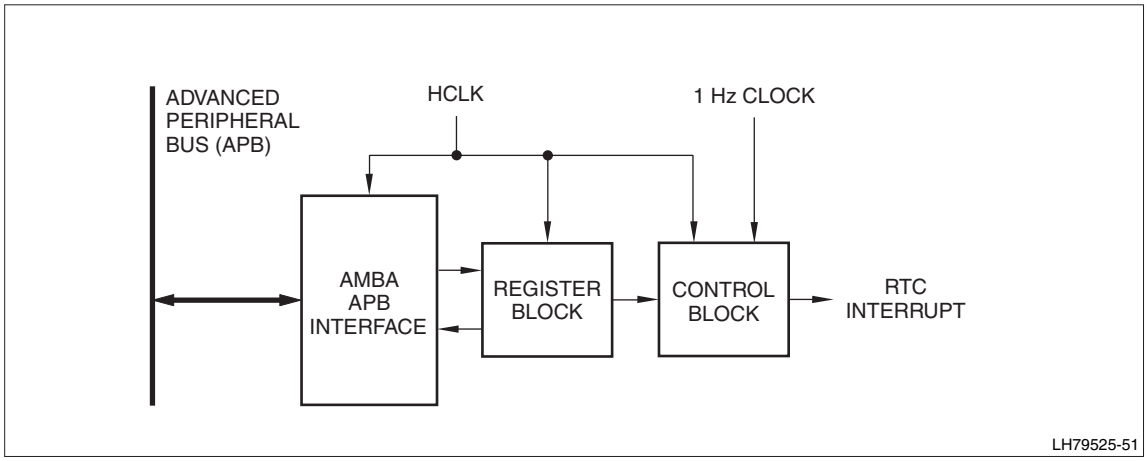


Figure 12-1. RTC Block Diagram

12.1 Theory of Operation

The RTC is based on a free-running 32-bit up-counter, clocked at 1 Hz. The RTC must be enabled in the Reset, Clock and Power Controller (RCPC) block, programming the PCKLCTRL0:RTC bit to 1. If the RTC is not used, program the PCKLCTRL0:RTC bit to 0, and the external 32.768 kHz crystal need not be provided. The XTAL32IN signal floats, therefore it should be tied LOW. (Clock generation is discussed in greater detail in Chapter 13, Reset, Clock, and Power Controller.)

Following reset, the RTC is disabled (as is the RTC Interrupt). Software must enable the RTC by programming the CR:START to 1. Once started, the 32-bit counter begins counting from 0x00000001, and upon reaching 0xFFFFFFFF, the counter wraps to 0x00000000 and continues incrementing. It cannot be stopped once it is started.

The value of the counter can be read by software from the Data Register (DR). This value changes every second. The RTC can be programmed to generate the RTC Interrupt when a value, programmed into the Match Register (MR), is reached. Program the Load Register (LR) to the timing start value. The elapsed real time is the difference between the value programmed in the LR and the value in the MR, in seconds (taking into account count wrapping).

To use the RTC Interrupt, it must be unmasked (programming IMSC:IMSC to 1). In addition, the interrupt must be enabled in the Vectored Interrupt Controller (VIC). The VIC register/bit INTENABLE:(bit 15) = 1. Other VIC registers that must be programmed include bit 15 of the INTSELECT register, and to assign a interrupt vector address using VECTADDRx and VECTCTRLx.

The interrupt is generated from HCLK so there is no delay between reaching the Match value and generating the RTC Interrupt.

Before using the RTC following reset, clear any pending interrupts (write 0x0001 to the ICR register) before enabling interrupts and/or RTC counting.

The RTC is reset by nRESETIN or by a software reset.

12.1.1 Configuring the RTC for Use

To configure the RTC:

1. Set the initial counter value by writing the value to the LR. This value becomes valid on the next CLK1HZ rising edge.
2. Set the interrupt trigger value by writing the value to the MR.
3. Clear pending interrupts to eliminate spurious interrupts that may have been generated at reset by writing 0x0001 to the ICR register.
4. Enable the RTC by programming the CR to 0x0001. Program the IMSC to 0x0001 to unmask the RTC Interrupt as the RTC Interrupt is masked following reset.
5. The RTC Interrupt can be masked by programming the IMSC to 0x0000 (the reset value).

The difference in values between LR and MR is the number of seconds that will elapse between starting the counter and interrupt generation.

The interrupt can be read in the Raw Interrupt Status Register (RIS). If the interrupt is not masked in the IMSC, it is also asserted to the VIC. Software can clear the interrupt by writing to the Interrupt Clear Register (ICR).

Note that the counter will continue to match the MR contents until updated one full second later on the next CLK1HZ rising edge. If software attempts to clear the interrupt during this period, another interrupt will be immediately generated. This should be taken into account when programming the RTC.

12.2 Register Reference

12.2.1 Memory Map

The base address for the RTC is: 0xFFFE0000

Table 12-1. RTC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	DR	Data Register
0x04	MR	Match Register
0x08	LR	Load Register
0x0C	CR	Control Register
0x10	IMSC	Interrupt Mask Set and Clear Register
0x14	RIS	Raw Interrupt Status Register
0x18	MIS	Masked Interrupt Status Register
0x1C	ICR	Interrupt Clear Register
0x20 - 0xFF	///	Reserved — Do not access.

12.2.2 Register Descriptions

12.2.2.1 Data Register (DR)

DR is the Read Data Register. The current value of the RTC counter can be read from this register.

Table 12-2. DR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	RTCDR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RTCDR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE0000 + 0x00															

Table 12-3. DR Fields

BIT	NAME	DESCRIPTION
31:0	RTCDR	**RTC Data Register** Contains the current counter value, in hexadecimal.

12.2.2.2 Match Register (MR)

MR is the Match Register. Program the value at which the RTC Interrupt will be generated into this register. The difference between this value and the value in the Load Register is the time in seconds, between count initiation and interrupt generation. The current setting can be read.

Table 12-4. MR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	RTCMR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RTCMR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE0000 + 0x04															

Table 12-5. MR Fields

BIT	NAME	DESCRIPTION
31:0	RTCMR	**RTC Match Register** Contains the match value in hexadecimal.

12.2.2.3 Load Register (LR)

LR is the Load Register. Program this register with the value from which to initiate the count sequence. The count begins on the next rising edge of the 1 Hz clock. Note that counting may not begin for up to one second.

Reading this register returns the last value written.

Table 12-6. LR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	RTCLR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RTCLR															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE0000 + 0x08															

Table 12-7. LR Fields

BIT	NAME	DESCRIPTION
31:0	RTCLR	**RTC Load Register** Hexadecimal start count value

12.2.2.4 Control Register (CR)

CR allows software to enable the RTC and determine its operational status.

Table 12-8. CR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															START
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW
ADDR	0xFFFE0000 + 0x0C															

Table 12-9. CR Fields

BIT	NAME	DESCRIPTION
31:1	///	**Reserved** Unpredictable values when read. Write the reset value.
0	START	**RTC Start** The RTC can be enabled by writing a 1 to this bit. Once enabled, any writes to this bit have no effect on the RTC until a system reset. Reading returns the status of the RTC. 1 = RTC enabled 0 = RTC disabled (read only)

12.2.2.5 Interrupt Mask Set or Clear Register (IMSC)

IMSC controls the masking of the interrupt generated by the RTC. Reading this register returns the current mask value of the RTC Interrupt.

Table 12-10. IMSC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															IMSC
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW
ADDR	0xFFFE0000 + 0x10															

Table 12-11. IMSC Fields

BITS	NAME	DESCRIPTION
31:1	///	**Reserved** Reading returns 0. Values written cannot be read.
0	IMSC	**Interrupt Mask Set or Clear** 1 = Interrupts unmasked, asserted to VIC when generated 0 = Interrupts masked and not asserted to the VIC

12.2.2.6 Raw Interrupt Status Register (RIS)

Reading this register gives the current raw status value of the RTC interrupt prior to masking. Writing has no effect.

Table 12-12. RIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															RIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE0000 + 0x14															

Table 12-13. RIS Fields

BITS	NAME	DESCRIPTION
31:1	///	**Reserved** Reading returns 0. Values written cannot be read.
0	RIS	**Raw Interrupt Status** Contains the raw state (prior to masking) of the RTC Interrupt. 1 = RTC Interrupt asserted 0 = RTC Interrupt not asserted

12.2.2.7 Masked Interrupt Status Register (MIS)

Reading the MIS register gives the current masked status value of the RTC interrupt. Writing has no effect.

Table 12-14. MIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															MIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE0000 + 0x18															

Table 12-15. MIS Fields

BITS	NAME	DESCRIPTION
31:1	///	**Reserved** Reading returns 0. Values written cannot be read.
0	MIS	**Masked Interrupt Status** Contains the masked interrupt state of the RTC Interrupt. 1 = RTC Interrupt unmasked and asserted 0 = RTC Interrupt masked or not asserted

12.2.2.8 Interrupt Clear Register (ICR)

Writing 1 to the ICR bit clears the RTC interrupt. Writing 0 has no effect. This register cannot be read.

Table 12-16. ICR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///															ICR
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO
ADDR	0xFFFE0000 + 0x1C															

Table 12-17. ICR Fields

BITS	NAME	DESCRIPTION
31:1	///	**Reserved** Unpredictable values when read. Values written cannot be read.
0	ICR	**Interrupt Clear** Write only; unpredictable values when read. 1 = Clears RTC Interrupt 0 = No effect

Chapter 13
Reset, Clock, and Power Controller

The Reset, Clock, and Power Controller (RCPC) manages the operating mode, generates appropriately prescaled clocks, and correctly times reset execution. The RCPC:

- Manages five Power Modes for minimizing power consumption: Active, Standby, Sleep, Stop1, and Stop2
- Generates the System Clock (HCLK) from either the System PLL clock or the PLL-bypassed (System Clock Oscillator) clock, prescaled by 2, 4, 6, 8, 10, 12…30
- Generates the CPU clock (FCLK) from either the System PLL clock or the PLL-bypassed (System Clock Oscillator) clock, prescaled by 2, 4, 6, 8, 10…30
- Generates the three UART clocks from System Clock Oscillator clock
- Generates the 1 Hz Real Time Clock (RTC)
- Generates the Liquid Crystal Display (LCDDCLK) clock from HCLK, prescaled by 1, 2, 4, 8, 16, 32, 64, 128, or 256
- Generates the Synchronous Serial Port (SSPCLK) clock from HCLK or the System Clock Oscillator clock, prescaled by 1, 2, 4, 8, 16, 32, or 64, 128, or 256
- Select the USB clock from HCLK or the USB PLL Clock, prescaled by 1, 2, or 4
- Generates the ADC clock from HCLK or the System Clock Oscillator clock, prescaled by 1, 2, 4, 8, 16, 32, or 64, 128, or 256
- Provides a selectable external clock output (CLKOUT)
- Generates system reset based on an external reset, watchdog timer reset, or soft reset
- Configures eight HIGH/LOW-level or rising/falling edge-trigger external interrupts and converts them to HIGH-level trigger interrupt outputs required by the Vectored Interrupt Controller (VIC)
- Generates remap outputs used by the memory map decoder
- Contains the Chip ID register
- Supports external or Watchdog reset status.
- Configures the PLL to generate System Clock from the System Clock Oscillator

Figure 13-1 shows a block diagram of the RCPC.

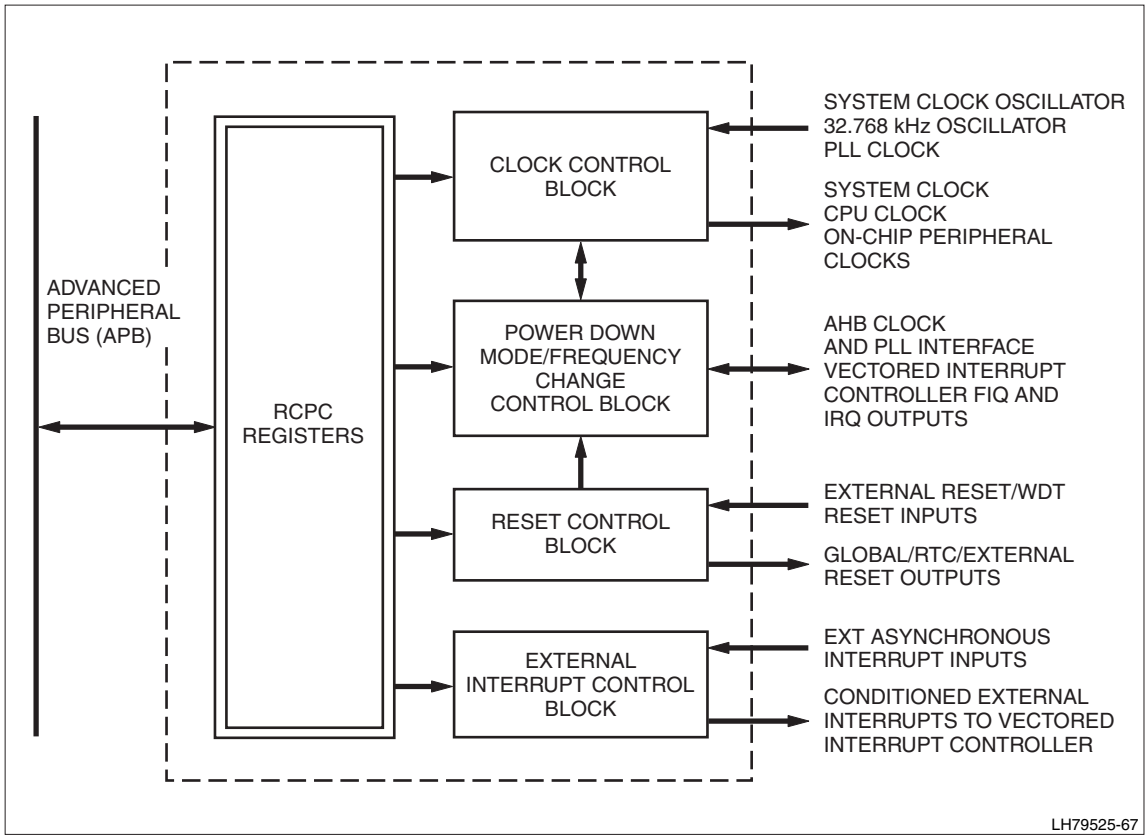


Figure 13-1. RCPC Block Diagram

13.1 Theory of Operation

The RCPC allows users to control system reset, clocks, power management, and external interrupt conditioning via the AMBA APB interface. This control includes:

- Enabling and disabling various clocks
- Managing power-down sequencing
- Selecting the sources for various clocks, and any predivision.

The RCPC ensures an orderly start-up until the System Clock crystal oscillator stabilizes and the Phase Lock Loop (System PLL) acquires lock. In addition, if users want to change the System PLL or System Clock frequency during normal operation, the RCPC ensures a seamless transition between the old and new frequencies. The same protection is not available, however, when changing the frequency of individual peripheral clocks; as a result, the peripheral must be disabled before changing frequency.

The RCPC also manages five Power Modes:

- Active
- Standby
- Sleep
- Stop1
- Stop2.

These modes reduce power consumption as needed, with each mode providing greater power savings. Active Mode is the normal operating mode. The other modes are entered via software control. The RCPC returns to Active Mode upon receiving an interrupt.

Seven external interrupt sources pass through the RCPC before being sent to the VIC. The interrupts entering the RCPC can be individually programmed to be either level-sensitive or edge-triggered, and either active-HIGH or active-LOW. All interrupts exiting the RCPC are converted to a format compatible with the VIC.

13.1.1 System PLL and USB PLL Reset

The System PLL and the USB PLL are reset only by a valid reset signal on nRESETIN. The Watchdog Timer and Software Reset **do not** reset the PLLs.

13.1.2 Reset Generation

The RCPC generates System Reset output. The nRESETOUT output pin is driven by the System Reset. The System Reset is asserted by any of these events:

- An external reset (a logic LOW signal on the external nRESETIN input pin)
- A signal from the internal Watchdog Timer (WDT)
- A Soft Reset

The reset latency depends on the System PLL lock state. If the System PLL is locked when an external reset is asserted, the System Reset output holds eight System Clock (HCLK) cycles after the external reset is released. Since the WDT and Soft Reset can be generated only if the System Clock is running, the System PLL must be locked. If the System PLL is not locked when an external reset is deasserted, the RCPC waits until the System PLL acquires lock and holds eight System Clock cycles before releasing the system reset output.

13.1.3 Clock Generation

The RCPC generates the System Clock, CPU clock, and on-chip peripheral clocks from:

- The System Clock crystal (connected to the XTALIN input pin and XTALOUT output pin)
- The 32.768 kHz crystal (connected to the XTAL32IN input pin and XTAL32OUT output pin)
- The internally generated PLL clocks.

There are two on-chip programmable PLLs, one for the System Clock and the other for the USB clock generation. The System PLL frequency can range from 10 MHz to 304.819 MHz by programming the SYSPLLCNTL Register. The USB PLL frequency can range from 20 MHz to 304.819 MHz by programming the USBPLLCTL Register. The System Clock and CPU clock are derived from the System PLL clock according to the value programmed in the SYSCLKPRE Register and CPUCLKPRE Register.

13.1.3.1 Enabling Clocks Prior to Programming Registers

The System Clocks connected to the DMA Controller, Ethernet Controller, External SDRAM Controller, USB Device, and LCD Controller are not active after reset. To activate these Clocks, program the PCLKCTRL 1 Register. Note that each of these clocks must be enabled before programming any registers in the particular block.

The RCPC and PLL interface guarantee that reprogramming the PLL and System Clocks results in an ordely frequency change. For the USB PLL and other clocks, program the Clock Select and frequency before enabling the peripheral clocks as the RCPC does not guarantee clean clock outputs when changing the clock source or USB PLL frequency.

13.1.3.2 Peripheral Block Clocks

The USB has two clocks from the RCPC. The System Clock (HCLK) controls AHB transfers and is enabled in the AHBCLKCTRL Register. The second clock originates in the USB PLL, and is programmed in the RCPC with the PCLKSEL1 and PCLKCTRL 1 registers.

Figure 13-2 schematically shows the progression of the clock divider chain for the USB Clock, as well as the registers that control each divider and the clock source. The registers referenced in the figure are defined in Section 13.2.

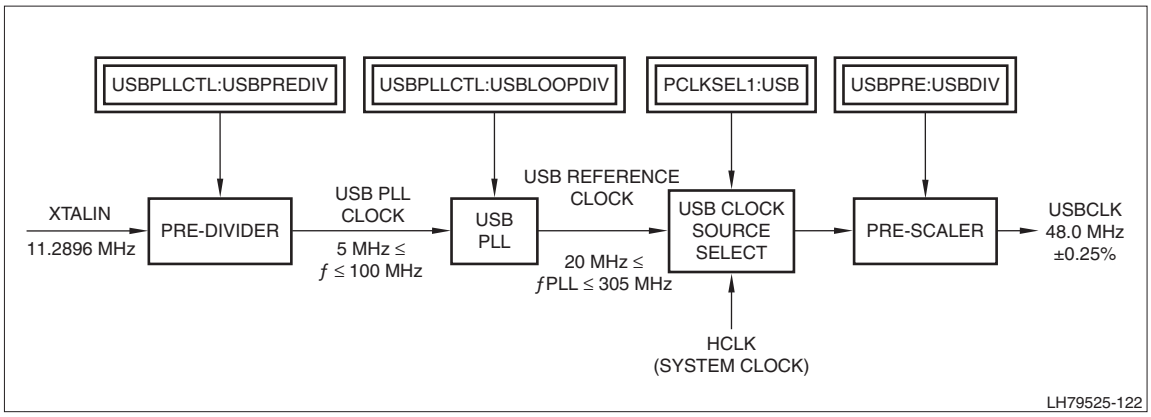


Figure 13-2. USB Clock Divider Chain

The RTC clock is generated from the 32.768 kHz crystal oscillator output. The 32.768 kHz oscillator's output is divided by 32,768 to produce the 1 Hz RTC clock. The UART clocks are generated from the System Clock crystal oscillator. To activate the RTC and UART clocks, program the PCLKCTRL0 Register.

The LCD data clock (LCDDCLK) is generated from the System Clock frequency. The SSP and ADC clocks are generated from either the System Clock or the System Clock Oscillator clock according to the value programmed in the PCLKSEL1 Register. These clocks are prescalable according to the values programmed in the SSPPRE Register, LCDPRE Register, and ADCPRE Register. To activate these clocks, program the PCLKCTRL1 Register.

Table 13-1 describes each clock and that clock's maximum frequency.

13.1.3.3 External Clock Generation (CLKOUT)

An external clock output signal is availabe on the CLKOUT pin. This signal is capable of driving 8 mA to external loads. The CLKOUT signal can be programmed to provide one of three clocks: FCLK, HCLK, or the system crystal oscillator frequency (nominally 11.2896 MHz). Prior to using the CLKOUT signal for external devices, software must choose the CLKOUT source using the CTRL:OUTSEL field (see Section 13.2.2.1).

Table 13-1. LH79524/LH79525 Clocks and Maximum Frequencies

NAME	FREQUENCY (MAX.)	DESCRIPTION
Oscillator Clock (CLK OSC)	20.0 MHz	External crystal oscillator input; used as the source for the three UARTs. Also an input to the PLL.
PLL System Clock (CLK PLL)	304.819 MHz	The PLL System clock originates in the PLL and is input to the RCPC; asynchronous to all other clocks.
PLL USB Clock	48.0 MHz	Originates in the PLL, can be gated and prescaled in the RCPC. This input to the USB is required to be 48 MHz; asynchronous to all other clocks.
32.768 kHz Oscillator Clock	32.768 kHz	External 32.768 kHz crystal oscillator input.
AHB Clock (HCLK)	50.803 MHz	Originates in the RCPC and is connected to AHB and APB peripherals. This clock can be separately prescaled. The clock is halted HIGH when the RCPC is in any power down mode other than Standby mode. This clock is synchronous to FCLK_CPU.
AHB Fast CPU Clock (FCLK CPU)	76.205 MHz	The AHB Fast CPU clock originates in the RCPC and is connected to the ARM720T core. This clock can be separately prescaled. The prescaled value can range from CLK_PLL/4 to CLK_PLL/30. The clock is halted HIGH when the RCPC is in any power down mode other than Standby. This clock is synchronous to HCLK and HCLK_CPU.
Ethernet Clock	50.803 MHz	This clock is generated in the RCPC and connected to the Ethernet block. This clock can be separately enabled/disabled, but is always the same frequency as HCLK. This clock is synchronous to HCLK.
DMA Clock	50.803 MHz	The DMA clock originates in the RCPC and is connected to the DMA block. This clock can be separately enabled and disabled, but always has the same frequency as HCLK. It is synchronous to HCLK.
External Memory Controller Clock	50.803 MHz	The EMC clock originates in the RCPC and connects to the EMC block. It can be separately enabled/disabled, but always has the same frequency as HCLK. This clock is synchronous to HCLK.
SSP Clock (SSPCLK)	50.803 MHz	The SSP clock originates in the RCPC and is connected to the SSP block. This clock can be separately enabled, disabled and prescaled. The prescaled value range is HCLK/1 to HCLK/256. This clock is asynchronous to all other clocks.
CLCD Clock (LCDDCLK)	50.803 MHz	The CLCD clock originates in the RCPC and is connected to the CLCD block. This clock can be separately enabled, disabled and prescaled. The prescaled value can range from HCLK/1 to HCLK/256. This clock is asynchronous to all other clocks.
UART[2:0] Clock	20.0 MHz	The UART clocks originate in the RCPC and are connected to the UART blocks. These are all separate and can be separately enabled/disabled. They will always have the same frequency as CLK_OSC. This clock is asynchronous to all other clocks.
RTC Clock	1.0 Hz	The RTC clock originates in the RCPC and is connected to the RTC block. This clock can be separately enabled and disabled. It will always have a frequency of 1 Hz. This clock is asynchronous to all other clocks.

13.1.4 Power Modes

The RCPC supports five Power Modes:

- Active mode
- Standby mode
- Sleep mode
- Stop1 mode
- Stop2 mode.

Table 13-2 shows which clocks are enabled in the various Power Modes.

Table 13-2. Clock and Enable States for Different Power Modes

DEVICE	ACTIVE	STANDBY	SLEEP	STOP1	STOP2
RTC Oscillator (32.768 kHz)	ON	ON	ON	ON	ON
System Clock Oscillator	ON	ON	ON	ON	OFF
System and USB PLLs	ON	ON	ON	OFF	OFF
System Clock	ON	ON	OFF	OFF	OFF
CPU Clock	ON	OFF	OFF	OFF	OFF

13.1.4.1 Active Mode

Active Mode is the normal Power Mode. The MCU enters this mode after start-up and upon exiting any other Power Mode. After an External Reset, Watchdog Timer Reset, or Soft Reset is released, the System Reset is held active for an extra eight System Clock cycles after the System Clock PLL is locked. The RTC oscillator is active, as it is in all modes.

13.1.4.2 Standby Mode

Standby Mode stops the clocks to the CPU and WDT while the rest of the device remains active. Standby Mode is entered when software writes 0b001 to the PWRDWNSEL field of the CTRL Register. When an interrupt is received, the RCPC exits Standby Mode and ensures an orderly transition to Active Mode. An interrupt should be held active until the RCPC exits Standby Mode. The RTC oscillator is active, as it is in all modes.

13.1.4.3 Sleep Mode

Sleep Mode stops all System Clocks, keeping only the PLLs, RTC Oscillator, and System Clock Oscillator active. This mode is entered when software writes 0b010 to the PWRDWNSEL field of the CTRL Register. When an interrupt is received, the RCPC exits Sleep Mode and ensures an orderly transition to Active Mode. An interrupt should be held active until the RCPC exits Sleep Mode. The RTC oscillator is active, as it is in all modes.

13.1.4.4 Stop1 Mode

Stop1 Mode stops all System Clocks and disables the PLLs, but keeps the System Clock Oscillator and RTC Oscillator active. This mode is entered when software writes 0b011 to the PWRDWNSEL field of the CTRL Register. When an interrupt is received, the RCPC exits Stop1 Mode and ensures an orderly transition to Active Mode. An interrupt should be held active until the RCPC exits Stop1 Mode.

13.1.4.5 Stop2 Mode

Stop2 Mode stops all System Clocks and disables both the PLLs and the System Clock Oscillator that feeds it. However, the 32.768 kHz internal oscillator remains active. This mode is entered when software writes 0b100 to the PWRDWNSEL field of the CTRL Register. When an interrupt is received, the RCPC exits Stop2 Mode and ensures an orderly transition to Active Mode. An interrupt should be held active until the RCPC exits Stop2 Mode.

13.1.4.6 Power Control in JTAG Mode

When using JTAG, the MCU cannot be placed in a low-power mode. This MCU will go into the low power mode, but is then immediately awakened by debug interrupts. Therefore, when in JTAG, the MCU should not be commanded to go into a low power mode.

13.2 Register Reference

This section provides the RCPC register memory mapping and bit fields.

13.2.1 Memory Map

The base address for the RCPC is: 0xFFFE2000.

Table 13-3. RCPC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	CTRL	RCPC Control Register
0x04	CHIPID	Chip Identification Register
0x08	REMAP	Remap Control Register
0x0C	SOFTRESET	Soft Reset Register
0x10	RSTSTATUS	Reset Status Register
0x14	RSTSTATUSCLR	Reset Status Clear Register
0x18	SYSCLKPRE	System Clock Prescaler Register
0x1C	CPUCLKPRE	CPU Clock Prescaler Register
0x20	///	Reserved — Do not access
0x24	PCLKCTRL0	Peripheral Clock Control 0 Register
0x28	PCLKCTRL1	Peripheral Clock Control 1 Register
0x2C	AHBCLKCTRL	AHB Clock Control
0x30	PCLKSEL0	Peripheral Clock Select Register 0
0x34	PCLKSEL1	Peripheral Clock Select Register 1
0x38	///	Reserved — Do not access
0x3C	SILICONREV	Silicon Revision Register
0x40	LCDPRE	LCD Prescaler Register
0x44	SSPPRE	SSP Prescaler Register
0x48	ADCPRE	ADC Prescaler Register
0x4C	USBPRE	USB Prescaler Register
0x50-0x7C	///	Reserved — Do not access
0x80	INTCONFIG	External Interrupt Configuration Register
0x84	INTCLR	External Interrupt Clear Register
0x88	CORECONFIG	Core Clock Configuration Register
0x8C	///	Reserved — Do not access
0xC0	SYSPLLCNTL	System PLL Control Register
0xC4	USBPLLCTL	USB PLL Control Register
0xC8	///	Reserved — Do not access

13.2.2 Register Descriptions

Except where noted, all registers are both writable and readable. Unless noted, unpredictable behavior results from writing anything but the reset values to any reserved location.

13.2.2.1 Control Register (CTRL)

The Control Register allows programming the Power Mode, Clock-Out source, and write protecting RCPC registers.

Table 13-4. CTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						LOCK	///	///	OUTSEL		PWRDWNSEL			///	
RESET	0	0	0	0	0	0	1	0	0	1	1	0	0	0	1	1
RW	RO	RO	RO	RO	RO	RO	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x00															

Table 13-5. CTRL Fields

BITS	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9	LOCK	**Lock** 0 = All RCPC registers that are accessible through the APB, other than this bit and the INTCLR Register, are write protected. 1 = All RCPC APB-accessible registers are write enabled. (default)
8	///	**Reserved** Reading returns 0. Write the reset value.
7	///	**Reserved** Read as 0. Always write 0. Writing a 1 causes unpredictable results.
6:5	OUTSEL	**CLKOUT Source Select** Select the source clock for the CLKOUT external clock pin. 00 = System Clock Oscillator (nominally 11.2896 MHz) 01 = Off* 10 = FCLK 11 = HCLK ***IMPORTANT:** For A.0 silicon, selection 01 is an **invalid** value. For A.1 silicon, the output is held LOW.
4:2	PWRDWNSEL	**Power Down Mode Select** These bits always read 0b000 because the RCPC clears them automatically at wakeup. 000 = Active Mode 001 = Standby Mode 010 = Sleep Mode 011 = Stop1 Mode 100 = Stop2 Mode Other values = undefined.
1:0	///	**Reserved** Reading returns 1. Always write 1.

13.2.2.2 Identification Register (CHIPID)

CHIPID is the Identification Register. This Read Only register contains the last three digits of the part number encoded as a 3 digit Binary Coded Decimal (BCD). The CHIPID register is used in conjunction with the SILICONREV register to provide the MCU part number (CHIPID:PARTNO) and the revision number of the silicon (SILICONREV:REVNO).

Table 13-6. CHIPID Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PARTNO												///			
RESET LH79524	0	1	0	1	0	0	1	0	0	1	0	0	0	0	0	0
RESET LH79525	0	1	0	1	0	0	1	0	0	1	0	1	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE2000 + 0x04															

Table 13-7. CHIPID Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:4	PARTNO	**Part Number Digits** Specifies the last three digits of the part number. 0x524 = LH79524 0x525 = LH79525
3:0	///	**Reserved** Reading returns 0. Write the reset value.

13.2.2.3 Remap Control Register (REMAP)

This REMAP Register provides a remapping feature for the system memory map. Figure 13-3 through Figure 13-6 show the effects of the REMAP bits.

Table 13-8. REMAP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														REMAP	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFE2000 + 0x08															

Table 13-9. REMAP Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1:0	REMAP	**REMAP** Remaps external and internal memory, system peripherals, and registers, as shown Figure 13-3 through Figure 13-6.

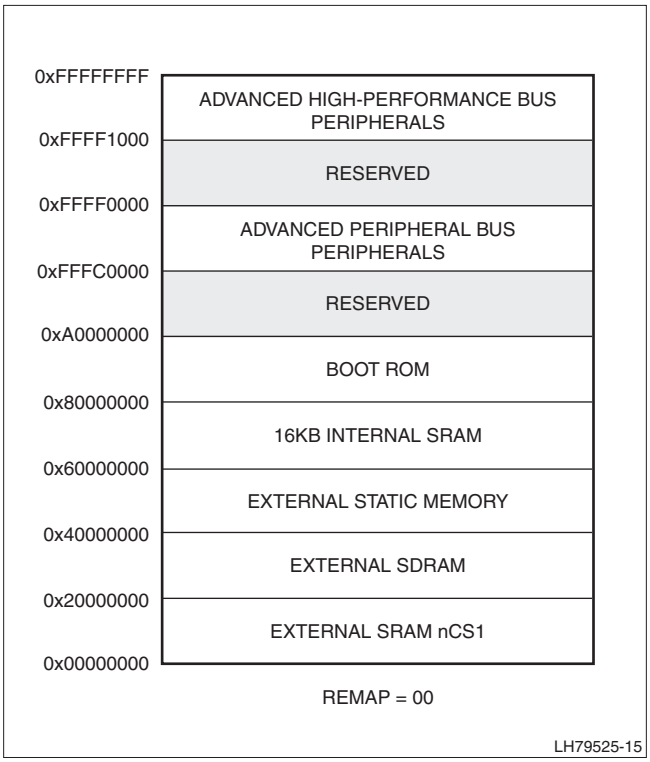


Figure 13-3. Remap = 0b00

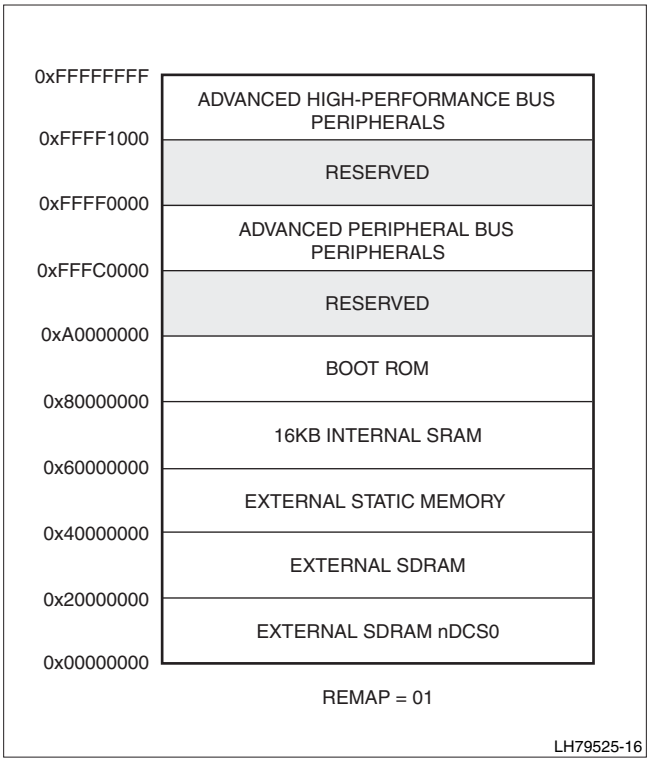


Figure 13-4. Remap = 0b01

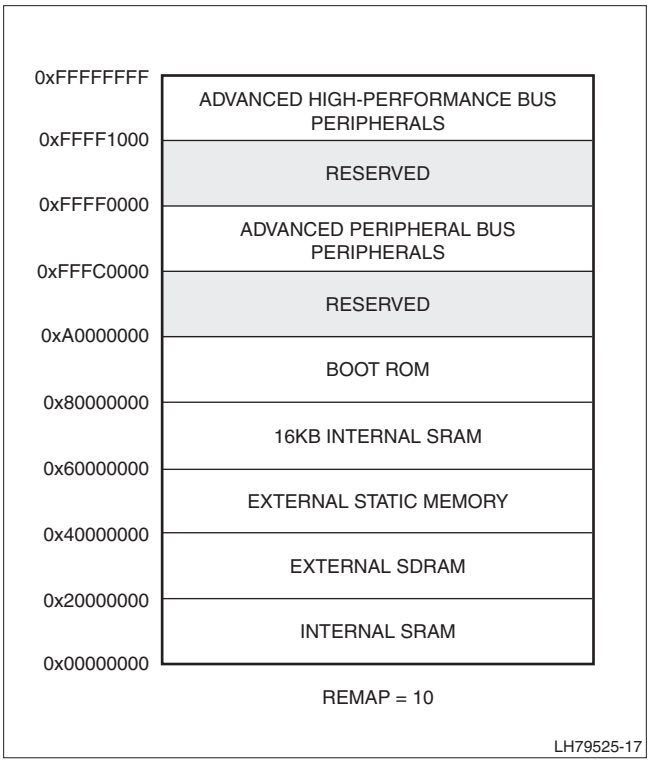


Figure 13-5. Remap = 0b10

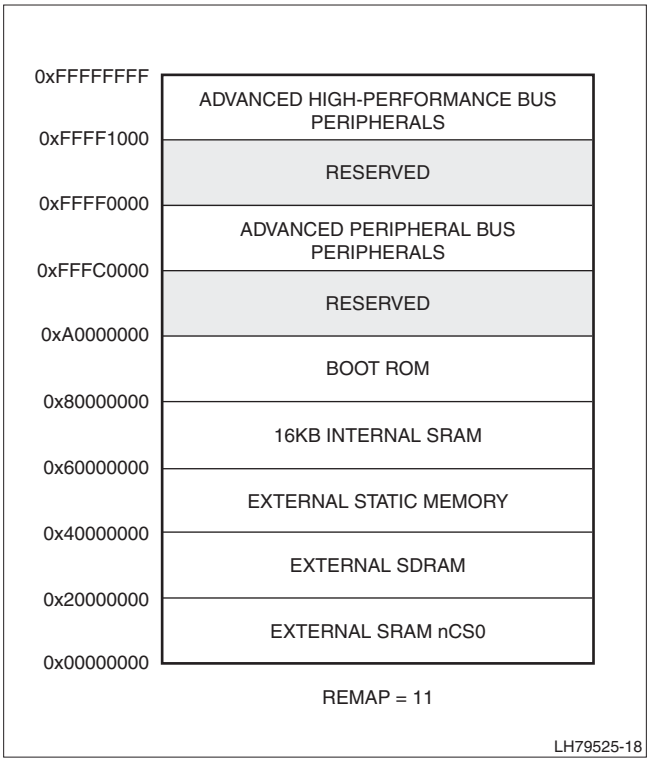


Figure 13-6. Remap = 0b11

13.2.2.4 Software Reset Register (SOFTRESET)

This register allows software to initiate a System Reset. To reset, software programs 0xDEAD into the lower 16 bits. SOFTRESET resets the entire chip, except the System PLL and the USB PLL.

Table 13-10. SOFTRESET Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SRST															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x0C															

Table 13-11. SOFTRESET Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	SRST	**Activate Software Reset** Writing 0xDEAD (0b1101 1110 1010 1101) to this field activates a System Reset. Do not write any other value to these bits.

13.2.2.5 Reset Status Register (RSTSTATUS)

This register provides the reset status of the MCU, containing both the external reset status and the WDT timeout reset status. Following external reset, the EXT bit is 1 and the WDTO bit is 0. At WDT timeout, only the WDTO bit is 1. The EXT and WDTO bits remain 1 until they are cleared by writing to the Reset Status Clear register.

Table 13-12. RSTSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														WDTO	EXT
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE2000 + 0x10															

Table 13-13. RSTSTATUS Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1	WDTO	**WDT Timeout** 1 = WDT timeout has occurred 0 = No WDT timeout has occurred since the flag was last cleared
0	EXT	**External Reset** 1 = External reset has occurred 0 = No external reset has occurred since the flag was last cleared

13.2.2.6 Reset Status Clear Register (RSTSTATUSCLR)

This Write Only register clears the two Reset Status flags in the RSTSTATUS register. Writing 1 to this register causes the corresponding bit in the RSTSTATUS to be cleared to 0. Writing 0 has no effect on the corresponding bit in the Reset Status register. Writing to reserved bits has no effect on the RCPC.

Table 13-14. RSTSTATUSCLR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														TOCLR	EXTCLR
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFE2000 + 0x14															

Table 13-15. RSTSTATUSCLR Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reads undefined. Write 0 only.
1	TOCLR	**Clear WDT Timeout** Write 1 to clear the WDTO status bit. Reads return unpredictable results. 1 = Clears WDTO bit in the RSTSTATUS Register to 0 0 = No effect
0	EXTCLR	**Clear External Reset** Write 1 to clear the EXT status bit. Reads return unpredictable results. 1 = Clears EXT bit in the RSTSTATUS Register to 0 0 = No effect

13.2.2.7 System Clock Prescaler Register (SYSCLKPRE)

HCLK is the System Clock. This register allows a divisor to be programmed that is used to divide the system PLL frequency to derive HCLK. The prescaled HCLK frequency is defined by:

$$f(\text{HCLK}) = f\left(\frac{f(\text{SystemPLL})}{2 \times HDIV}\right)$$

Following reset, the prescaler is programmed to divide by 30. Table 13-18 shows example values for HDIV.

Table 13-16. SYSCLKPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												HDIV			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x18															

Table 13-17. SYSCLKPRE Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	HDIV	**HCLK Divisor** Program with the divisor for the HCLK prescaler. All HDIV combinations are valid except 0b0000.

Table 13-18. SYSCLKPRE Register Values

HDIV	DIVISOR	f(HCLK)
0b0000	—	Invalid
0b0001	2	f(System PLL)/2
0b0010	4	f(System PLL)/4
0b0011	6	f(System PLL)/6
0b0100	8	f(System PLL)/8
0b0101	10	f(System PLL)/10
:	:	:
0b1111 (default)	30	f(System PLL)/30

13.2.2.8 CPU Clock Prescaler Register (CPUCLKPRE)

FCLK is the CPU Clock. This register allows a divisor to be programmed that is used to divide the system PLL frequency to derive FCLK. The prescaled FCLK frequency is defined by:

$$f(\text{FCLK}) = f\left(\frac{f(\text{SystemPLL})}{2 \times FDIV}\right)$$

Following reset, the prescaler is programmed to divide by 30. Table 13-21 shows example values for FDIV. The CPU clock must always be greater than or equal to the system bus clock. Thus when changing clock dividers, the CPU frequency should be increased BEFORE, decreased AFTER, and always kept at least equal to the system bus frequency.

Table 13-19. CPUCLKPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												FDIV			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x1C															

Table 13-20. CPUCLKPRE Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	FDIV	**FCLK Divisor** Program with the divisor for the HCLK prescaler. All FDIV combinations are valid except 0b0000.

Table 13-21. CPUCLKPRE Register Values

FDIV	DIVISOR VALUE	*f*(FCLK)
0b0000	—	Invalid
0b0001	2	*f*(System PLL)/2
0b0010	4	*f*(System PLL)/4
0b0011	6	*f*(System PLL)/6
0b0100	8	*f*(System PLL)/8
0b0101	10	*f*(System PLL)/10
:	:	:
0b1111 (default)	30	*f*(System PLL)/30

13.2.2.9 Peripheral Clock Control Register 0 (PCLKCTRL0)

This register controls the RTC, UART0, UART1, and UART2 peripheral clocks. Programming a bit to 1 disables the corresponding peripheral's clock. These clocks are more fully described in Table 13-1.

Table 13-22. PCLKCTRL0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						RTC	///						U2	U1	U0
RESET	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
RW	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x24															

Table 13-23. PCLKCTRL0 Fields

BITS	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9	RTC	**RTC Clock** 1 = Disables the RTC input clock 0 = Enables the RTC input clock
8:3	///	**Reserved** Reading returns 1. Write the reset value.
2	U2	**UART2 Clock** Enables and disables the internal clock to UART2. 1 = Disables the UART2 clock 0 = Enables the UART2 clock
1	U1	**UART1 Clock** Enables and disables the internal clock to UART1. 1 = Disables the UART1 clock 0 = Enables the UART1 clock
0	U0	**UART0 Clock** Enables and disables the internal clock to UART0. 1 = Disables the UART0 clock 0 = Enables the UART0 clock

13.2.2.10 Peripheral Clock Control Register 1 (PCLKCTRL1)

This register controls the USB, ADC, LCD, and SSP peripheral clocks. Programming a bit to 1 disables the corresponding peripheral's clock. The SSP Clock, USB Clock, and the LCD Data Clock are more fully described in Table 13-1. The ADC Clock is described in Section 2.1.3 of this User's Guide.

Table 13-24. PCLKCTRL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												USB	ADC	SSP	LCD
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x28															

Table 13-25. PCLKCTRL1 Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	USB	**USB Clock** Enables and disables the internal 48 MHz clock to the USB Device peripheral. 1 = Stops the USB Clock 0 = Enables the USB Clock
2	ADC	**ADC Clock** Enables and disables the internal ADC clock generator. 1 = Stops the ADC Clock 0 = Enables the ADC Clock
1	SSP	**SSP Clock (SSPCLK)** Enables and disables the clock presented to the SSPCLK pin. 1 = Stops the SSP Clock 0 = Enables the SSP Clock.
0	LCD	**LCD Data Clock (LCDDCLK)** Enables and disables the clock presented to the Color LCD Contoller hardware and the LCDDCLK pin. 1 = Stops the LCD Data Clock 0 = Enables the LCD Data Clock

13.2.2.11 AHB Clock Control Register (AHBCLKCTRL)

This register controls the AHB clocks to several peripherals. Programming a bit to 1 disables the AHB clock to the corresponding peripheral. Following reset, all AHB clocks are enabled.

For unused peripherals, software should program a 1 to the corresponding bit in this register to reduce overall power consumption.

Table 13-26. AHBCLKCTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											LCD	USB	ETHERNET	SDRAM	DMA
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x2C															

Table 13-27. AHBCLKCTRL Fields

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading returns 0. Write the reset value.
4	LCD	**AHB LCD Clock** 1 = Disables the LCD AHB clock 0 = Enables the LCD AHB peripheral clock
3	USB	**AHB USB Clock** 1 = Disables the USB AHB clock 0 = Enables the USB AHB peripheral clock
2	ETHERNET	**AHB ETHERNET Clock** 1 = Disables the ETHERNET AHB clock 0 = Enables the ETHERNET AHB peripheral clock
1	SDRAM	**AHB External SDRAM Controller Clock** 1 = Disables the SDRAM AHB clock 0 = Enables the SDRAM AHB peripheral clock
0	DMA	**AHB DMA Clock** 1 = Disables the DMA AHB clock 0 = Enables the DMA AHB peripheral clock

13.2.2.12 Peripheral Clock Select Register 0 (PCLKSEL0)

This register allows selection of the clock source for the UARTs.

Table 13-28. PCLKSEL0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													UART2	UART1	UART0
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
ADDR	0xFFFE2000 + 0x30															

Table 13-29. PCLKSEL0 Fields

BITS	NAME	DESCRIPTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	UART2	**UART2 Clock Source** 1 = System Clock (HCLK) 0 = Crystal oscillator output
1	UART1	**UART1 Clock Source** 1 = System Clock (HCLK) 0 = Crystal oscillator output
0	UART0	**UART0 Clock Source** 1 = System Clock (HCLK) 0 = Crystal oscillator output

13.2.2.13 Peripheral Clock Select Register 1 (PCLKSEL1)

This register allows selection of the clock source for the USB, ADC, and SSP peripherals.

Note that the default source for the USB clock following reset is HCLK. For virtually all designs, this must be programmed to the USB PLL following reset. Failing to do this could result in the USB Client not operating properly.

Table 13-30. PCLKSEL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												USB	ADC	SSP	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x34															

Table 13-31. PCLKSEL1 Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	USB	**USB Peripheral Clock Source** Following reset, HCLK is the source. In almost all cases, this should be programmed to 1 following reset. 1 = USB PLL Clock 0 = System Clock (HCLK)
2	ADC	**ADC Peripheral Clock Source** 1 = System Clock Oscillator frequency 0 = System Clock (HCLK)
1	SSP	**SSP Peripheral Clock Source** 1 = System Clock Oscillator frequency 0 = System Clock (HCLK)
0	///	**Reserved** Reading returns 0. Write the reset value.

13.2.2.14 Silicon Revision Register (SILICONREV)

The SILICONREV register is used in conjunction with the CHIPID register to provide the MCU part number (CHIPID:PARTNO) and the revision number of the silicon (SILICONREV:REVNO).

Table 13-32. SILICONREV Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	REVNO															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE2000 + 0x3C															

Table 13-33. SILICONREV Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Do not write.
15:0	REVNO	**Revision Number** The revision number read from this field corresponds to the values below. 0x0 = Undefined 0x1 = Revision A.0 0x2 = Revision A.1

13.2.2.15 LCD Clock Prescaler Register (LCDPRE)

The value in this register is used as a divisor for HCLK to derive the LCD Data Clock (LCDDCLK) frequency. Following reset, the prescaler is programmed to pass the clock through without division. Table 13-36 shows the valid combinations for LCDDIV and the resulting LCDDCLK frequency. All other LCDDIV values are invalid.

Table 13-34. LCDPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								LCDDIV							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x40															

Table 13-35. LCDPRE Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	LCDDIV	**LCD Data Clock Divisor** Program with the divisor for the LCD Data Clock prescaler.

Table 13-36. LCDPRE Register Values

LCDDIV	DIVIDER VALUE	*f*(LCD)
0b00000000 (default)	1	*f*(HCLK)
0b00000001	2	*f*(HCLK)/2
0b00000010	4	*f*(HCLK)/4
0b00000100	8	*f*(HCLK)/8
0b00001000	16	*f*(HCLK)/16
0b00010000	32	*f*(HCLK)/32
0b00100000	64	*f*(HCLK)/64
0b01000000	128	*f*(HCLK))/128
0b10000000	256	*f*(HCLK)/256

13.2.2.16 SSP Clock Prescaler Register (SSPPRE)

The value in this register is used as a divisor for the Source Clock to derive the SSP clock (SSPCLK) frequency. The SSP clock source (System Clock Oscillator, or HCLK) is selected with the PCLKSEL1:SSP bit (see Section 13.2.2.13). Table 13-39 shows the valid combinations for SSPDIV and the resulting SSP clock frequency. Following reset, the prescaler is programmed to pass the clock through without division. All other SSPDIV values are invalid.

Table 13-37. SSPPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SSPDIV							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x44															

Table 13-38. SSPPRE Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	SSPDIV	**SSP Divisor** Program with the clock source divisor for the SSP Clock prescaler (see Table 13-39).

Table 13-39. SSPPRE Register Values

SSPDIV	DIVISOR	*f*(SSP)
0b00000000 (default)	1	*f*(clock source)
0b00000001	2	*f*(clock source)/2
0b00000010	4	*f*(clock source)/4
0b00000100	8	*f*(clock source)/8
0b00001000	16	*f*(clock source)/16
0b00010000	32	*f*(clock source)/32
0b00100000	64	*f*(clock source)/64
0b01000000	128	*f*(clock source)/128
0b10000000	256	*f*(clock source)/256

13.2.2.17 ADC Clock Prescaler Register (ADCPRE)

The value in this register is used as a divisor for the Source Clock to derive the ADC clock (ADCCLK) frequency. The ACD clock source (System Clock Oscillator, or HCLK) is selected with the PCLKSEL1:ADC bit (see Section 13.2.2.13). Following reset, the prescaler is programmed to pass the clock through without division. Table 13-42 shows the valid combinations for ADCDIV and the resulting ADC clock frequency. All other ADCDIV values are invalid.

Table 13-40. ADCPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								ADCDIV							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x48															

Table 13-41. ADCPRE Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	ADCDIV	**ADC Clock Divisor** Program with the clock source divisor for the ADC Clock prescaler (see Table 13-42).

Table 13-42. ADCPRE Register Values

ADCDIV	DIVISOR	f(ADCCLK)
0b00000000 (default)	1	f(clock source)
0b00000001	2	f(clock source)/2
0b00000010	4	f(clock source)/4
0b00000100	8	f(clock source)/8
0b00001000	16	f(clock source)/16
0b00010000	32	f(clock source)/32
0b00100000	64	f(clock source)/64
0b01000000	128	f(clock source)/128
0b10000000	256	f(clock source)/256

13.2.2.18 USB Clock Prescaler Register (USBPRE)

The value in this register is used as a divisor for the clock source to derive the USB clock (USBCLK) frequency. The USB clock source (PLL clock, or HCLK) is selected with the PCLKSEL1:USB bit (see Section 13.2.2.13). It is important to note that this bit defaults to select HCLK, and must be reprogrammed to use the USB PLL as the source for most designs.

The on-board USB Device requires a 48 MHz clock for Full Speed (12 Mbp/s) operation. The reset value of USBDIV is 0x00, resulting in division by 1. If the USB PLL is programmed for higher-frequency operation to improve jitter, this field must be programmed to the appropriate divisor following reset (see Section 13.2.2.23).

Table 13-45 shows the valid combinations for USBDIV and the resulting USB clock frequency. All other USBDIV values are invalid.

Table 13-43. USBPRE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								USBDIV							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x4C															

Table 13-44. USBPRE Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	USBDIV	**USB Clock Divisor** Program with the clock source divisor for the USB Clock prescaler (Table 13-45).

Table 13-45. USBPRE Register Values

USBDIV	DIVISOR	f(ADC)
00000000 (default)	1	f(clock source)
00000001	2	f(clock source)/2
00000010	4	f(clock source)/4

13.2.2.19 External Interrupt Configuration Register (INTCONFIG)

This register configures the individual external interrupts to be either edge-sensitive or level-sensitive, and either active HIGH or active LOW. Following reset, all bits are 0 and configure the external interrupts to be active LOW, level sensitive.

The corresponding edge-trigger interrupt should be cleared before enabling the edge-trigger interrupt to avoid a false interrupt.

Table 13-46. INTCONFIG Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	INT7		INT6		INT5		INT4		INT3		INT2		INT1		INT0	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0x80															

Table 13-47. INTCONFIG Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:14	INT7	**Configures External Interrupt INT7** 00 = Configures INT7 to be a LOW, level trigger. 01 = Configures INT7 to be a HIGH, level trigger. 10 = Configures INT7 to be a falling-edge trigger. 11 = Configures INT7 to be a rising-edge trigger.
13:12	INT6	**Configures External Interrupt INT6** 00 = Configures INT6 to be a LOW, level trigger. 01 = Configures INT6 to be a HIGH, level trigger. 10 = Configures INT6 to be a falling-edge trigger. 11 = Configures INT6 to be a rising-edge trigger.
11:10	INT5	**Configures External Interrupt INT5** 00 = Configures INT5 to be a LOW, level trigger. 01 = Configures INT5 to be a HIGH, level trigger. 10 = Configures INT5 to be a falling-edge trigger. 11 = Configures INT5 to be a rising-edge trigger.
9:8	INT4	**Configures External Interrupt INT4** 00 = Configures INT4 to be a LOW, level trigger. 01 = Configures INT4 to be a HIGH, level trigger. 10 = Configures INT4 to be a falling-edge trigger. 11 = Configures INT4 to be a rising-edge trigger.

Table 13-47. INTCONFIG Fields

BITS	NAME	DESCRIPTION
7:6	INT3	**Configures External Interrupt INT3** 00 = Configures INT3 to be a level trigger, active LOW. 01 = Configures INT3 to be a level trigger, active HIGH. 10 = Configures INT3 to be a falling-edge trigger. 11 = Configures INT3 to be a rising-edge trigger.
5:4	INT2	**Configures External Interrupt INT2** 00 = Configures INT2 to be a level trigger, active LOW. 01 = Configures INT2 to be a level trigger, active HIGH. 10 = Configures INT2 to be a falling-edge trigger. 11 = Configures INT2 to be a rising-edge trigger.
3:2	INT1	**Configures External Interrupt INT1** 00 = Configures INT1 to be a level trigger, active LOW. 01 = Configures INT1 to be a level trigger, active HIGH. 10 = Configures INT1 to be a falling-edge trigger. 11 = Configures INT1 to be a rising-edge trigger.
1:0	INT0	**Configures External Interrupt INT0** 00 = Configures INT0 to be a level trigger, active LOW. 01 = Configures INT0 to be a level trigger, active HIGH. 10 = Configures INT0 to be a falling-edge trigger. 11 = Configures INT0 to be a rising-edge trigger.

13.2.2.20 External Interrupt Clear Register (INTCLR)

This register individually clears active external interrupts. This register can clear edge-triggered interrupts only. Writing to undefined bits has no effect on the RCPC. Note that the reset state is indeterminate since this is write only.

Table 13-48. INTCLR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								INT 7	INT 6	INT 5	INT 4	INT 3	INT 2	INT 1	INT 0
RESET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
RW	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFE2000 + 0x84															

Table 13-49. INTCLR Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading is indeterminate. Write the reset value.
7	INT7	**Clear INT7 Interrupt** 1 = Clears the active edge-triggered interrupt INT7 0 = No effect
6	INT6	**Clear INT6 Interrupt** 1 = Clears the active edge-triggered interrupt INT6 0 = No effect
5	INT5	**Clear INT5 Interrupt** 1 = Clears the active edge-triggered interrupt INT5 0 = No effect
4	INT4	**Clear INT4 Interrupt** 1 = Clears the active edge-triggered interrupt INT4 0 = No effect
3	INT3	**Clear INT3 Interrupt** 1 = Clears the active edge-triggered interrupt INT3 0 = No effect
2	INT2	**Clear INT2 Interrupt** 1 = Clears the active edge-triggered interrupt INT2 0 = No effect
1	INT1	**Clear INT1 Interrupt** 1 = Clears the active edge-triggered interrupt INT1 0 = No effect
0	INT0	**Clear INT0 Interrupt** 1 = Clears the active edge-triggered interrupt INT0 0 = No effect

13.2.2.21 Core Clock Configuration Register (CORECONFIG)

This register can be programmed to select either the Standard Mode or the FastBus extension for the ARM720T bus interface. In Standard mode, either a synchronous or asynchronous operation can be selected. When changing from Standard Mode to FastBus, the CPU clock must always be greater than or equal to the system bus clock.

Table 13-50. CORECONFIG Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														CCLK	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFE2000 + 0x88															

Table 13-51. CORECONFIG Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reading is indeterminate. Write the reset value.
1:0	CCLK	**Core Clock Configuration** Program this field to configure the ARM720T core clock. 00 = Standard Mode, asynchronous operation 01 = FastBus extension mode 10 = Standard Mode, synchronous operation 11 = FastBus extension mode

13.2.2.22 System PLL Control Register (SYSPLLCTL)

This register controls the System PLL frequency. System PLL frequency is calculated by:

$$\text{SystemPLLfrequencySystem} = \frac{\text{SystemClockOscillatorFrequency} \times \text{SYSLOOPDIV}}{\text{SYSPREDIV}}$$

The maximum System PLL frequency is 304.819 MHz.

Table 13-52. SYSPLLCTL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		///	SYSFRANGE	SYSPREDIV						SYSLOOPDIV					
RESET	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0xC0															

Table 13-53. SYSPLLCTL Fields

BITS	NAME	DESCRIPTION
31:14	///	**Reserved** Reading returns 0. Write the reset value.
13	///	**Reserved** Reading returns 0. *Write 1 only*.
12	SYSFRANGE	**System PLL Output Frequency Range Select** 1 = 100 MHz to 304.819 MHz (best jitter performance achieved) 0 = 20 MHz to 100 MHz
11:6	SYSPREDIV	**System PLL Pre-Divider** Prescales the System PLL Reference clock. The divisor chosen must satisfy the equation: (System Clock Oscillator frequency) ÷ (SYSPREDIV) ≥ 5 MHz
5:0	SYSLOOPDIV	**System PLL Loop-Divider** Prescales the System PLL Feedback clock. The divisor can be programmed from 1 to 63.

13.2.2.23 USB PLL Control Register (USBPLLCTL)

This register controls the USB PLL frequency and power down. The USB PLL frequency is calculated by:

$$USBPLLFrequency = \left(\frac{(SystemClockOscillatorFrequency) \times USBLOOPDIV}{USBPREDIV}\right)$$

The maximum USB PLL frequency is 304.819 MHz.

Since the jitter is better with the PLL running above 100 MHz, it is best to program the PLL to a frequency greater than 100 MHz and divide it by four using the USBPRE:USBDIV bit.

Here is a programming example using the typical crystal frequency of 11.2896 MHz:

Target frequency (48 MHz) × 4 = 192 MHz
Program USBLOOPDIV = 17 and USBPREDIV = 1
47.9808 ÷ 48.000 MHz = 0.04% Well within USB 2.0 Specification of ±0.25%
192 MHz = (11.2896 MHz × USLOOPDIV) ÷ USBPREDIV
192 MHz ÷ 11.2896 MHz = USBLOOPDIV ÷ USBPREDIV = 17.0068
(11.2896 MHz ÷ 17)/1 = 191.9232 MHz
191.9232 MHz/(USBDIV = 4) = 47.9808 MHz

To reduce power consumption when not using USB, USBPREDIV should be programmed to 2 while the USBLOOPDIV should be programmed to 1.

Table 13-54. USBPLLCTL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///		///	USBFRANGE	USBPREDIV						USBLOOPDIV					
RESET	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	1
RW	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFE2000 + 0xC4															

Table 13-55. USBPLLCTL Fields

BITS	NAME	DESCRIPTION
31:14	///	**Reserved** Writing to these bits has no effect. Reading returns 0.
13	///	**Reserved** Write the reset value.
12	USBFRANGE	**USB PLL Output Frequency Range Select** 1 = 100 MHz - 304.819 MHz (best jitter performance achieved) 0 = 20 MHz - 100 MHz
11:6	USBPREDIV	**USB PLL Pre-Divider** Prescales the USB PLL Reference clock. The divisor chosen must satisfy the equation: (System Clock Oscillator frequency) ÷ (USBPREDIV) ≥ 5 MHz

Table 13-55. USBPLLCTL Fields (Cont'd)

5:0	USBLOOPDIV	**USB PLL Loop-Divider** Prescales the USB PLL Feedback clock. The divisor can be programmed from 1 to 63.

Chapter 14
Synchronous Serial Port

The Synchronous Serial Port is a master or slave interface that enables synchronous serial communication with slave or master peripherals in one of three modes:

- Motorola SPI
- Texas Instruments DSP-compatible synchronous serial interface
- National Semiconductor Microwire.

The SSP incorporates two 16-bit-wide, 8-entry-deep FIFOs, one for transmitting data and one for receiving data.The transmit FIFO takes data written to it and transmits it on the serial interface. The receive FIFO parallellizes the serial data stream and presents it in a FIFO for access by other devices. If the receive FIFO is not empty, the SSP can assert an interrupt after a specified number of clock ticks elapses following the start of a receive transfer. This feature permits interrupt-driven data transfers that are greater than the FIFO watermark, but not an even multiple of it.

The SSP also features:

- Programmable clock rate
- Programmable data frame size, from 4 to 16 bits. Each frame transmits most-significant bit first.
- Single combined interrupt is an OR function of the individual interrupt requests.
- Loopback Test Mode.

14.1 Theory of Operation

The SSP is a master or slave interface for synchronous serial communication with slave peripheral devices that have Motorola SPI, National Semiconductor Microwire, or Texas Instruments DSP-compatible synchronous serial interfaces. The SSP performs serial-to-parallel conversion on data received from a peripheral device. The transmit and receive paths are buffered with internal FIFO memories. These memories can store eight 16-bit values independently in both transmit and receive modes. Serial data is transmitted on the SSPTX pin and received on the SSPRX pin.

During transmission, data writes to the transmit FIFO via the APB interface. The transmit data is queued up for parallel-to-serial conversion onto the transmit interface. The transmit logic formats the data into one of three basic frame types:

- Motorola SPI
- Texas Instruments DSP-compatible Synchronous Serial Interface
- National Semiconductor Microwire.

Table 14-1 describes these modes.

Table 14-1. Feature Comparison

MODE	DESCRIPTION	DATA TRANSFERS	COMMENT
SPI	Lets the SSP communicate with Motorola SPI-compatible devices.	Full-duplex, 4-wire synchronous	Clock polarity and phase are programmable.
SSI	Lets the SSP communicate with Texas Instruments DSP-compatible Serial Synchronous Interface devices.	Full-duplex, 4-wire synchronous	
Microwire	Lets the SSP communicate with National Semiconductor Microwire-compatible devices.	Half-duplex synchronous, using 8-bit control messages	

For all three formats, the serial clock (SSPCLK) is held inactive while the SSP is idle and transitions at the programmed frequency only during active transmission of data. The SSPCLK pin can be HIGH during idle in SPI Mode if the SPO bit in the Control Register is set.

For Motorola SPI and National Semiconductor Microwire frame formats, the serial frame (SSPFRM) pin is active LOW and asserted (pulled down) during the entire frame transmission. Both formats output data on the falling edge of the clock and latch input data on the rising edge of the clock.

14.1.1 Timing Waveforms

Figure 14-1 shows the standard set of SSP timing waveforms. Timing values for the call-outs on the figure can be found in the Data Sheet.

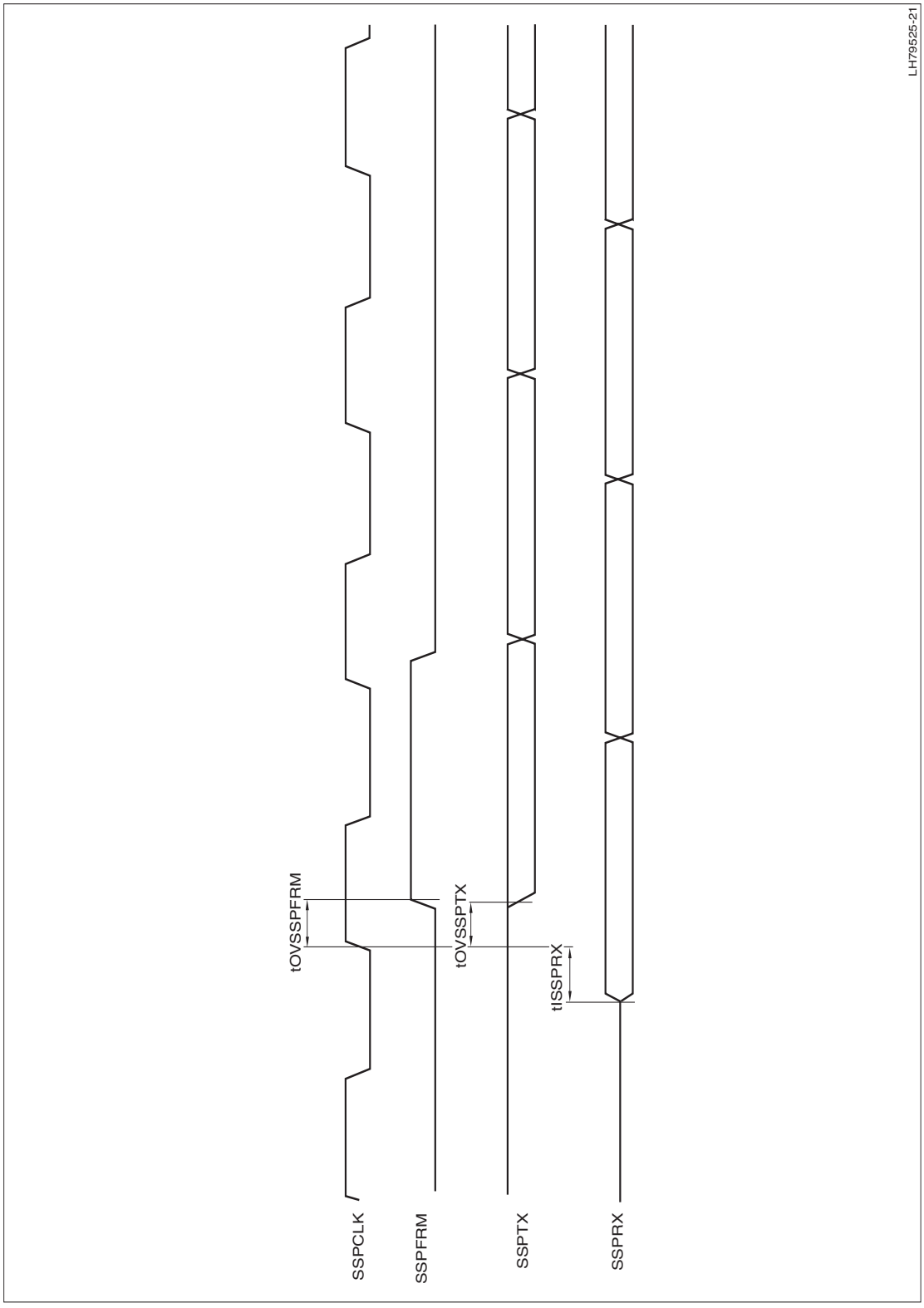


Figure 14-1. SSP Timing Waveform Parameters

14.1.2 Motorola SPI Frame Format

For the Motorola SPI format, the serial frame pin (SSPFRM) is active LOW. The SPO and SPH bits in SSP Control Register 0 influence SSPCLK and SSPFRM operation in Single and Continuous Modes.

Figure 14-2 and Figure 14-3 show the Motorola SPI frame format for single data transfers, continuous data transfers, and when SPH equals 0 or 1.

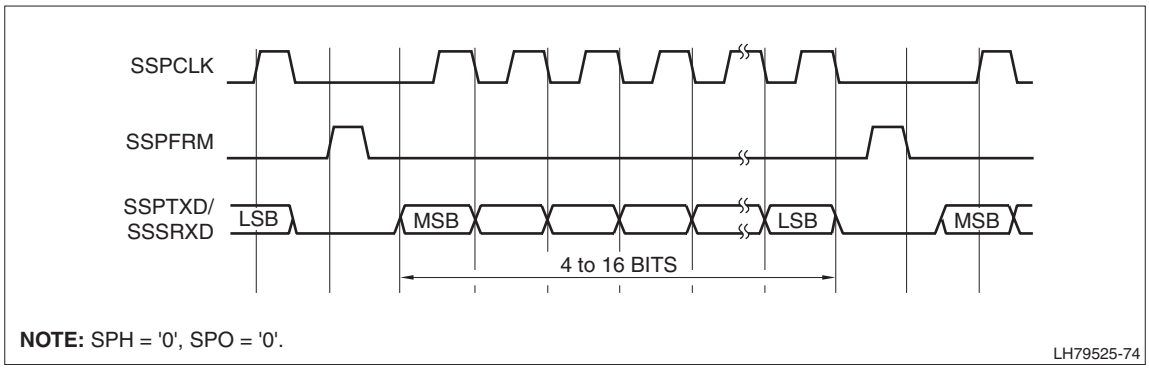


Figure 14-2. Motorola SPI Frame Format (Continuous Transfer)

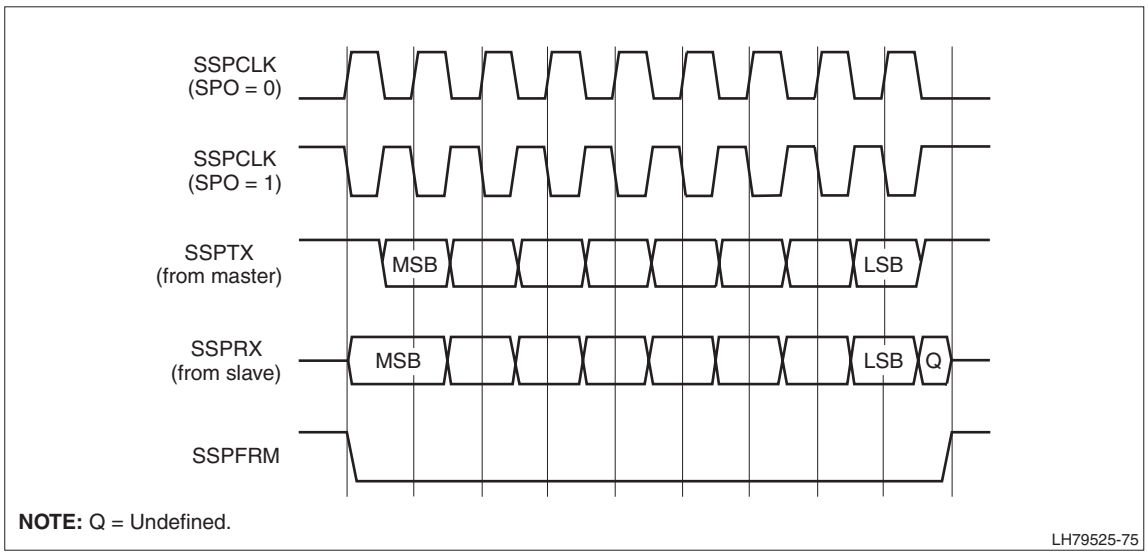


Figure 14-3. Motorola SPI Frame Format with SPH = 0

14.1.3 Texas Instruments Frame Format

For the Texas Instruments DSP-compatible synchronous serial interface frame format, the SSPFRM pin is pulsed for one serial clock period stating at its rising edge, prior to each frame's transmission. For this frame format, the SSP outputs data on the rising edge of the clock and latches input data on the rising edge of the clock.

Figure 14-4 shows the Texas Instruments DSP format for a single transfer. Figure 14-5 shows the same format for continuous transfers.

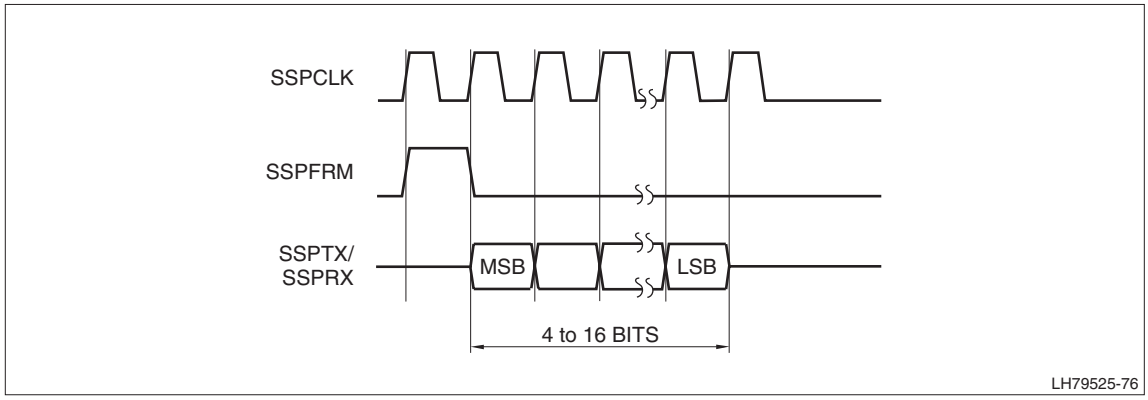


Figure 14-4. Texas Instruments Synchronous Serial Frame Format (Single Transfer)

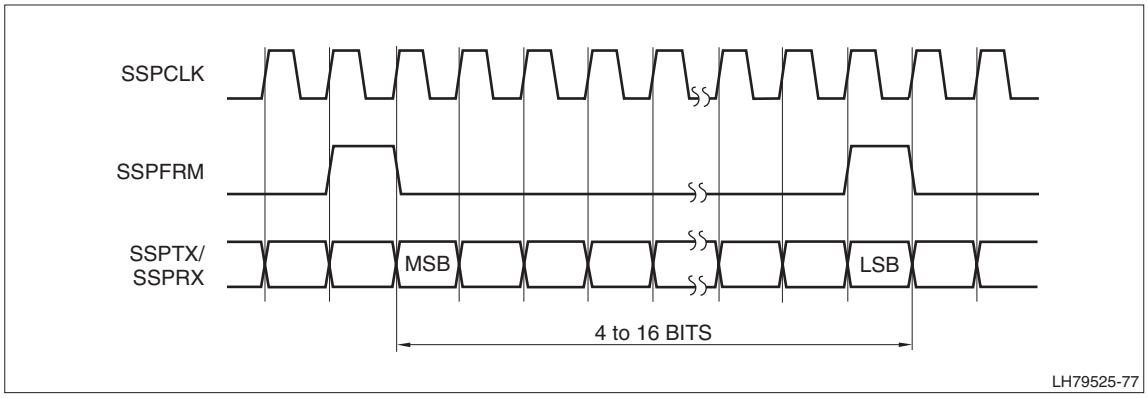


Figure 14-5. Texas Instruments Synchronous Serial Frame Format (Continuous Transfers)

14.1.4 National Semiconductor Frame Format

Unlike the full-duplex transmission capabilities that the other two frame formats support, the National Semiconductor Microwire format uses a special half-duplex, master-slave messaging technique. In this mode:

1. When a frame begins, an 8-bit control message is transmitted to the off-chip slave. During this transmission, the SSP does not receive incoming data.
2. After the message is sent, the off-chip slave decodes it, waits one serial clock after the last bit of the 8-bit control message has been sent, and responds with the requested data. The returned data can be 4 to 16 bits long, making the total frame 13 to 25 bits long.

During reception, data goes through a serial-to-parallel conversion before being placed into the receive FIFO. The data is then read out via the AMBA APB interface.

In the Microwire mode, the SSP slave samples the first bit of receive data on the rising edge of SSPCLK after SSPRX has gone LOW. Masters that drive a free-running SSPCLK must ensure that the SSPRX signal has sufficient setup and hold margins with respect to the rising edge of SSPCLK.

Figure 14-6 shows the National Semiconductor Microwire format for a single transfer. Figure 14-7 shows this format for continuous transfers.

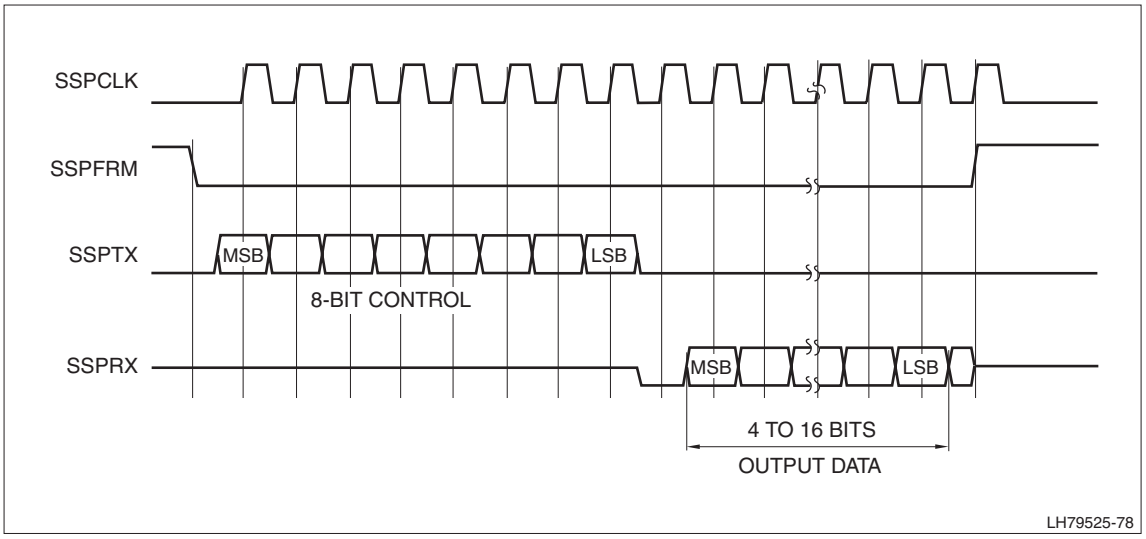


Figure 14-6. Microwire Frame Format (Single Transfer)

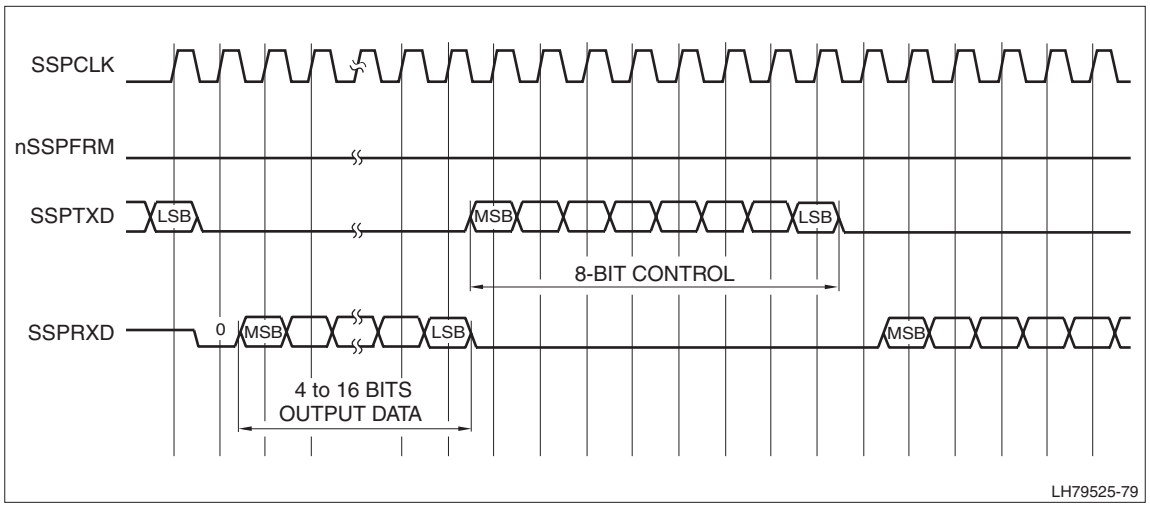


Figure 14-7. Microwire Frame Format (Continuous Transfers)

14.1.5 Clock Generation

The serial bit rate is derived by dividing down the SSP clock coming from the RCPC that is a prescaled version of the system clock (see the RCPC chapter for detailed information about setting up the system clock). The clock is first divided by an even prescale value, DVSR, from 2 to 254, which is programmed in the CPSR Register. The clock is further divided by a value from 1 to 256, which is CPD + 1 (where CPD is the value programmed in the CTRL0 Register). The frequency of the output clock SSPCLK is defined as f(SSPCLK) = (DVSR × (1 + CPD)).

14.1.6 Interrupts

The SSP can assert four types of interrupts. A single combined interrupt, comprising these four signals, goes to the VIC:

- SSPRXINTR — SSP Receive FIFO Service Interrupt, locally maskable
- SSPTXINTR — SSP Transmit FIFO Service Interrupt, locally maskable
- SSPRORINTR — SSP Receive Overrun Interrupt, locally maskable
- SSPRXTOINTR — SSP Receive FIFO Timeout Interrupt

All four interrupts are combined into a single interrupt: SSPINTR. The status of the four interrupt sources can be read from the MIS or RIS Register.

14.1.6.1 Receive Interrupt

SSPRXINTR is the Receive Interrupt. This interrupt is asserted when there are four or more valid entries in the receive FIFO. The interrupt is cleared by reading the receive FIFO until there are three or fewer entries.

14.1.6.2 Transmit Interrupt

SSPTXINTR is the Transmit Interrupt. This interrupt is asserted when the FIFO is less than or equal to half full (when there is space for four or more entries). The interrupt is cleared when there are five or more entries in the transmit FIFO.

This interrupt is not qualified with the Synchronous Serial Port Enable bit (bit [1]) in CTRL1, allowing operation in one of two ways. Data can be written to the transmit FIFO prior to enabling the SSP and the interrupts. Alternatively, the SSP and interrupts can be enabled so that data can be written to the transmit FIFO by an ISR. For more information about Control Register 1, see Section 14.2.2.2. The SSPTXINTR interrupt is always set if the Synchronous Serial Port Enable bit in Control Register 1 is not set (see Section 14.2.2.2).

14.1.6.3 Receive Overrun Interrupt

SSPRORINTR is the Receive Overrun Interrupt. This interrupt is asserted when the FIFO is already full and an additional data frame is received, causing an overrun of the FIFO. Data is over-written in the Shift Register, but not the FIFO.

14.1.6.4 Receive Timeout Interrupt

SSPRXTOINTR is the Receive Timeout Interrupt. This interrupt is asserted when the receive FIFO is not empty and the SSP has remained idle for a fixed 32 bit-clock period (it is not programmable). This interrupt ensures that software knows that data remains in the FIFO and must be read. The interrupt resets if the FIFO is emptied by subsequent Reads or if new data is introduced to the FIFO. In both master and slave modes, the receive timeout interrupt can be reset by writing to the ICR:RTIC bit, reading the contents of the receive FIFO until empty, or if new activity occurs on the respective clock lines.

14.1.6.5 SSPINTR

The SSPRXINTR, SSPTXINTR, SSPRORINTR, and SSPRXTOINTR interrupts are also combined into the single output SSPINTR. This interrupt is an OR function of the individual interrupt sources. This combined interrupt is the only one going to the vectored interrupt controller (VIC).

The combined SSP Interrupt is asserted if any of the four individual interrupts is asserted and enabled.

14.2 Register Reference

This section provides the SSP's register memory mapping and bit fields.

14.2.1 Memory Map

The base address for the SSP is 0xFFFC6000. Locations at offsets 0x028 through 0xFFF are reserved and must not be accessed.

Table 14-2. SSP Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	CTRL0	Control Register 0
0x004	CTRL1	Control Register 1
0x008	DR	Data Register
0x00C	SR	Status Register
0x010	CPSR	Clock Prescale Register
0x014	IMSC	Interrupt Mask Set and Clear Register
0x018	RIS	Raw Interrupt Status Register
0x01C	MIS	Masked Interrupt Status Register
0x020	ICR	Interrupt Clear Register
0x024	DCR	DMA Control Register
0x028 - 0xFFF	///	Reserved — Do not access

14.2.2 Register Descriptions

14.2.2.1 Control Register 0 (CTRL0)

This register, defined in Table 14-3 and Table 14-4, enables or disables the SSP and controls the serial clock rate, its phase, polarity, data size, and frame format. Bits 3:0 reset to 0b0000 and must be programmed to a valid number prior to using the SSP.

Table 14-3. CTRL0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CPD								SPH	SPO	FRF		DSS			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC6000 + 0x000															

Table 14-4. CTRL0 Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:8	CPD	**Clock Prescale Divisor** To generate the bit rate and Serial Clock output (SSPCLK pin), the SSP uses two divisors on the generated 5.6448 MHz Clock Input (when using the recommended 11.2896 MHz crystal): • This programmable clock rate divisor in the CTRL0 register • A programmable prescaler in the Clock Prescaler register Divisor field SSPCLK is calculated as follows: SSPCLK = ƒCLOCK INPUT/(CPSR:DVSR × (1 + CPD)) Program this field to the desired eight-bit CPD value in the above equation. Valid values are 0 - 0xFF.
7	SPH	**SSPCLK Phase** Applicable to Motorola SPI frame format only. 1 = SSPCLK pin is HIGH when data is not being transferred 0 = SSPCLK pin is LOW when data is not being transferred
6	SPO	**SSPCLK Polarity** Applicable to Motorola SPI frame format only. 1 = Data is captured on the second clock edge 0 = Data captured on the first clock edge
5:4	FRF	**Frame Format** Program to select the format for the interface used. 00 = Motorola SPI frame format 01 = TI synchronous serial frame format 10 = National Microwire frame format 11 = Reserved, undefined operation

Table 14-4. CTRL0 Fields

BITS	NAME	DESCRIPTION
3:0	DSS	**Data Size Select** Program with the correct data block size. 0000 = Undefined Operator 0001 = Undefined Operator 0010 = Undefined Operator 0011 = 4-bit data 0100 = 5-bit data 0101 = 6-bit data 0110 = 7-bit data 0111 = 8-bit data 1000 = 9-bit data 1001 = 10-bit data 1010 = 11-bit data 1011 = 12-bit data 1100 = 13-bit data 1101 = 14-bit data 1110 = 15-bit data 1111 = 16 bit data **IMPORTANT:** This field resets to 0b0000 and must be programmed to a valid number prior to using the SSP.

14.2.2.2 Control Register 1 (CTRL1)

CTRL1 is the Control Register 1. CTRL1 contains four bit fields that control various SSP functions.

Table 14-5. CTRL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												SOD	MS	SSE	LBM
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFC6000 + 0x004															

Table 14-6. CTRL1 Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	SOD	**Slave Mode Output Disable** This bit is relevant only in the slave mode (MS = 1). In multiple-slave systems, it is possible for an SSP master to broadcast a message to all slaves in the system while ensuring that only one slave drives data onto its serial output line. In such systems the RX lines from multiple slaves could be tied together. To operate in such systems, the SOD bit can be programmed to 1. 1 = SSP must not drive the SSPTX output in slave mode 0 = SSP can drive the SSPTX output in slave mode
2	MS	**Master or Slave Mode Select** This bit can only be modified when the SSP is disabled, SSE = 0. 1 = device configured as slave 0 = device configured as master
1	SSE	**Synchronous Serial Port Enable** This bit enables and disables the SSP. 1 = SSP operation is enabled 0 = SSP operation is disabled
0	LBM	**Loopback Mode** Program this bit to enable or disable Loopback mode: 1 = Enables Loopback mode, internally connecting the Transmit serial shifter output to the Receive serial shifter input 0 = Enables normal serial port operation, disabling Loopback mode

14.2.2.3 Data Register – Receive/Transmit FIFO Register (DR)

DR is the 16-bit-wide Receive/Transmit FIFO register.

- When DR is read, the entry in the receive FIFO (pointed to by the current FIFO read pointer) is accessed. As data values are removed by the SSP's receive logic from the incoming data frame, they are placed into the entry in the receive FIFO (pointed to by the current FIFO write pointer).
- When DR is written to, the entry in the transmit FIFO (pointed to by the write pointer), is written with the DR data. Data values are removed from the transmit FIFO one value at a time by the transmit logic. Each value is loaded into the transmit serial shifter, then serially shifted out onto the SSPTX pin at the programmed bit rate.

When a data size of less than 16 bits is selected, software must right-justify data written to the transmit FIFO. The transmit logic ignores the unused bits. Received data less than 16 bits is automatically right-justified in the 16-bit wide receive buffer. Software should ignore (by masking) the receive buffer upper unused bits.

When the SSP is programmed for National Microwire frame format in master mode, the default size for transmit data is eight bits (the most-significant byte is ignored). The receive data size is controlled by software. When the SSP is the slave device, it will receive 8 bits of control data and transmit 4 to 16 bits of data. The transmit FIFO and the receive FIFO are not cleared, even when the SSE bit in Control Register 1 is set to 0 (refer to Section 14.2.2.2). This allows the software to fill the transmit FIFO before enabling the SSP.

Table 14-7. DR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	DATA															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC6000 + 0x008															

Table 14-8. DR Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	DATA	**Transmit/Receive FIFO** Right-justify data when the SSP is programmed for a data size that is smaller than 16 bits. Unused bits at the top are ignored by transmit logic. The receive logic automatically right-justifies. Read = Receive FIFO top entry Write = Transmit FIFO top entry

14.2.2.4 Status Register (SR)

SR is the Status Register. This register contains bits that indicate the FIFO fill status and the SSP busy status.

Table 14-9. SR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											BSY	REFI	RNE	TNF	TFE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
RW	W	W	W	W	W	W	W	W	W	W	W	RO	RO	RO	RO	RO
ADDR	0xFFFC6000 + 0x00C															

Table 14-10. SR Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Writing to these bits has no effect. Reading returns 0.
15:5	///	**Reserved** Write as zero. Unpredictable behavior when read.
4	BSY	**SSP Busy Flag** 1 = SSP is transmitting/receiving a frame or the transmit FIFO is non-empty 0 = SSP is idle
3	REFI	**Receive FIFO Full** 1 = Receive FIFO is full 0 = Receive FIFO is not full
2	RNE	**Receive FIFO Not Empty** 1 = Receive FIFO is not empty 0 = Receive FIFO is empty
1	TNF	**Transmit FIFO Not Full** 1 = Transmit FIFO is not full 0 = Transmit FIFO is full
0	TFE	**Transmit FIFO Empty** 1 = Transmit FIFO is empty 0 = Transmit FIFO is not empty

14.2.2.5 Clock Prescale Register (CPSR)

The CPSR Register specifies the division factor by which the input HCLK is internally divided before use. The value programmed into this register is a value from 2 to 254. This register defaults to zero, but is double buffered and reads back 1s after Reset. Because it resets to zero, it must be programmed prior to enabling the SSP.

Table 14-11. CPSR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								DVSR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	WO	WO	WO	WO	WO	WO	WO	WO	RW	RW	RW	RW	RW	RW	RW	RO
ADDR	0xFFFC6000 + 0x010															

Table 14-12. CPSR Fields

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:8	///	**Reserved** Write as zero. Unpredictable behavior when read.
7:0	DVSR	**Clock Prescale Divisor** To generate the bit rate and Serial Clock output (SSPCLK), the SSP uses two divisors on the generated 5.6448 MHz Clock Input (when using the recommended 11.2896 MHz crystal): • This programmable prescaler in the Clock Prescaler register Divisor field • A programmable clock rate divisor in the CTRL0 register (CTRL0:CPD) Program this field to the desired even-number eight-bit value between 2 and 254 for DVSR shown in the equation (note that bit zero is always 0, hence DVSR is always an even number). SSPCLK is calculated as follows: SSPCLK = fCLOCK INPUT/(DVSR × (1 + CPD))

14.2.2.6 Interrupt Mask Set and Clear Register (IMSC)

IMSC is the Interrupt Mask Set and Clear Register. On a read, this register gives the current value of the mask on the relevant interrupt. A write of 1 to the particular bit clears the mask, enabling the interrupt to be read. A write of 0 sets the corresponding mask.

All bits are cleared to 0 when reset.

Table 14-13. IMSC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												TXIM	RXIM	RTIM	RORIM
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFC6000 + 0x014															

Table 14-14. IMSC Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	TXIM	**Transmit FIFO Interrupt Mask** 1 = Interrupt not masked 0 = Interrupt is masked
2	RXIM	**Receive FIFO Interrupt Mask** 1 = Interrupt not masked 0 = Interrupt is masked
1	RTIM	**Receive Timeout Interrupt Mask** 1 = Interrupt not masked 0 = Interrupt is masked
0	RORIM	**Receive Overrun Interrupt Mask** 1 = Interrupt not masked 0 = Interrupt is masked

14.2.2.7 Raw Interrupt Status Register (RIS)

This register provides the current raw status value of the corresponding interrupt prior to masking. A write has no effect.

Table 14-15. RIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												TXRIS	RXRIS	RTRIS	RORRIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC6000 + 0x018															

Table 14-16. RIS Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	TXRIS	**Transmit FIFO Raw Interrupt Status** Gives the raw interrupt state (prior to masking) of the Transmit FIFO interrupt. 1 = Interrupt asserted 0 = Interrupt not asserted
2	RXRIS	**Receive FIFO Raw Interrupt Status** Gives the raw interrupt state (prior to masking) of the Receive FIFO interrupt. 1 = Interrupt asserted 0 = Interrupt not asserted
1	RTRIS	**Receive Timeout Raw Interrupt Status** Gives the raw interrupt state (prior to masking) of the Receive Timeout interrupt. 1 = Interrupt asserted 0 = Interrupt not asserted
0	RORRIS	**Receive Overrun Raw Interrupt Status** Gives the raw interrupt state (prior to masking) of the Receive Overrun interrupt. 1 = Interrupt asserted 0 = Interrupt not asserted

14.2.2.8 Masked Interrupt Status Register (MIS)

MIS is the Masked Interrupt Status Register. When read, this register gives the current masked status value of the corresponding interrupt. A write has no effect.

Table 14-17. MIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												TXMIS	RXMIS	RTMIS	RORMIS
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFC6000 + 0x01C															

Table 14-18. MIS Fields

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	TXMIS	**Transmit FIFO Masked Interrupt Status** Gives the Transmit FIFO masked interrupt state. 1 = Interrupt asserted 0 = Interrupt not asserted or is masked
2	RXMIS	**Receive FIFO Masked Interrupt Status** Gives the Receive FIFO masked interrupt state. 1 = Interrupt asserted 0 = Interrupt not asserted or is masked
1	RTMIS	**Receive Timeout Masked Interrupt Status** Gives the Receive Timeout masked interrupt state. 1 = Interrupt asserted 0 = Interrupt not asserted or is masked
0	RORMIS	**Receive Overrun Masked Interrupt Status** Gives the Receive Overrun masked interrupt state. 1 = Interrupt asserted 0 = Interrupt not asserted or is masked

14.2.2.9 Interrupt Clear Register (ICR)

ICR is the Interrupt Clear Register. This register is write only. On a write of 1, the corresponding interrupt is cleared. Writing 0 has no effect.

Table 14-19. ICR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														RTIC	RORIC
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO
ADDR	0xFFFC6000 + 0x020															

Table 14-20. ICR Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1	RTIC	**Receive Timeout interrupt clear** 1 = Clear the interrupt 0 = No effect
0	RORIC	**Receive Overrun interrupt clear** 1 = Clear the interrupt 0 = No effect

14.2.2.10 DMA Control Register (DCR)

DCR is the DMA Control Register.

The RXDMAE and TXDMAE bits are not automatically cleared for standard Stream 0 through 3 DMA operations, respectively. These bits should be explicitly cleared by software as soon as possible following DMA completion.

On initiating a DMA operation the DMAC:CTRL:ENABLE bit should be set before the RXDMAE bit, is set.

Table 14-21. DCR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///														TXDMAE	RXDMAE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW
ADDR	0xFFFC6000 + 0x024															

Table 14-22. DCR Fields

BITS	NAME	DESCRIPTION
31:2	///	**Reserved** Reading returns 0. Write the reset value.
1	TXDMAE	**Transmit DMA Enable** 1 = DMA for the transmit FIFO is enabled 0 = Transmit DMA disabled
0	RXDMAE	**Receive DMA Enable** 1 = DMA for the receive FIFO is enabled 0 = Receive DMA disabled

Chapter 15
Timers

The LH79524 and LH79525 both have three 16-bit timers:

- Timer 0 has five Capture Registers and two Compare Registers.
- Timer 1 has two Capture Registers and two Compare Registers.
- Timer 2 has two Capture Registers and two Compare Registers.

Throughout this chapter, all descriptions apply to both the LH79524 and LH79525. In this chapter there are also a number of registers with similar names and functions. Descriptions of these registers refer to the name with an ‘x’ replacing the timer number, for example a reference to the CTRLx register refers to all three Timer Control Registers (CTRL0, CTRL1, and CTRL2). In some instances, such as the Timer Capture Registers, ‘x’ represents the timer number, and ‘n’ represents the lettered or numbered register suffix. For example, TxCAPn would represent all the Timer Capture Registers; T1CAPn would represent all the Timer 1 Capture Registers; T1CAPA is the specific register.

The timers are clocked by HCLK, but scaled down internally for use in the PWM and compare functions. All counters are incremented by an internal prescaled counter clock or external clock and can generate an overflow interrupt when the counter increments from 0xFFFF to 0x0000. All three timers have separate internal prescaled counter clocks, with either a common external clock (CTCLK) or a prescaled version of HCLK.

All Capture Registers have edge-selectable inputs and can generate an interrupt, if desired. Each timer can also generate a separate interrupt, which is asserted if any enabled compare, capture, or overflow interrupt condition occurs. The interrupt remains active until all compare, capture, and overflow interrupts are cleared. Each Compare Register can force its associated compare output pin either HIGH or LOW upon a match.

Figure 15-1 shows a block diagram of the three timers.

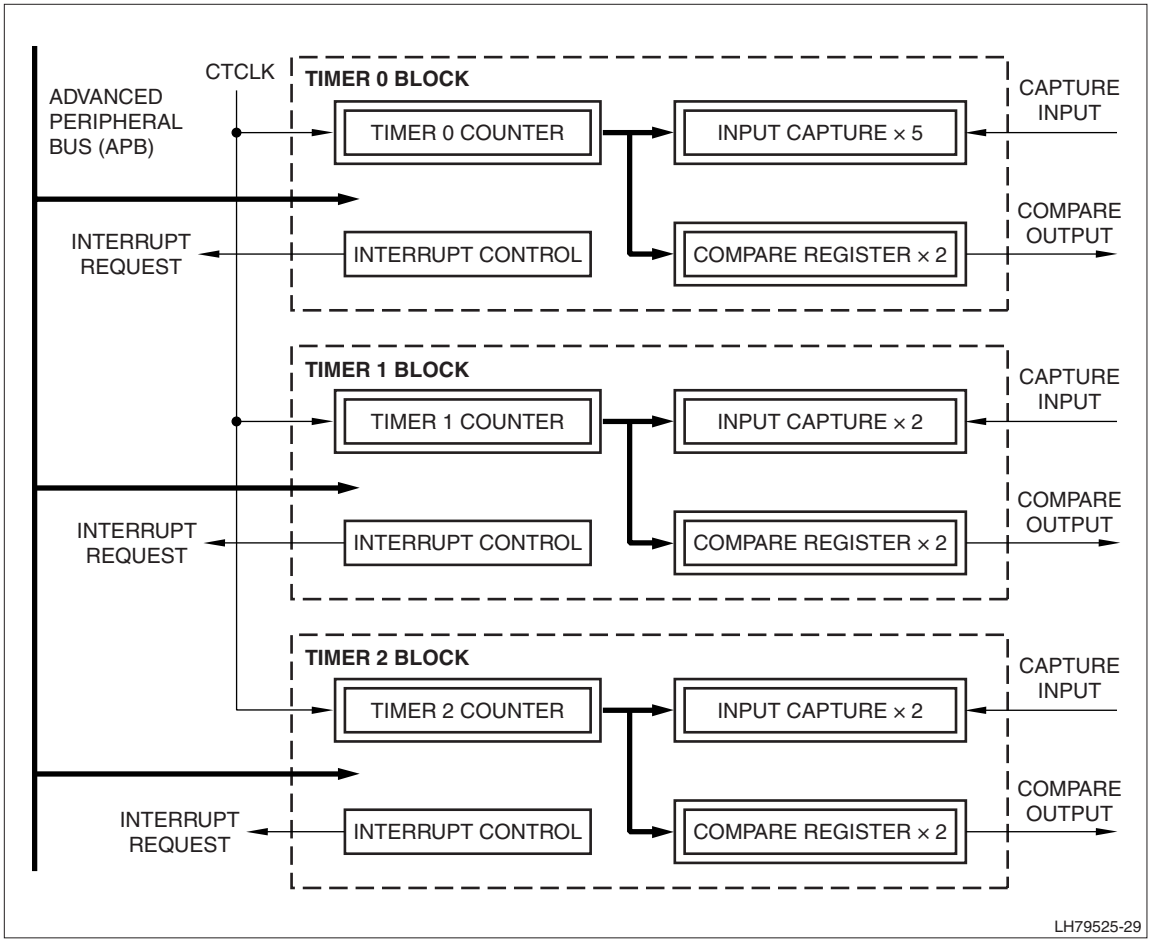


Figure 15-1. Timer Block Diagram

15.1 Theory of Operation

Each counter can use either one of the supported internal divided-by-n HCLKs or an external clock as its count clock. Clock selection is accomplished using the Timer Control Register (CTRLx) for the timer to be programmed.

The clock can be changed only when the counter is in Stop Mode. Attempts to change the count clock while the counter is running are ignored.

To change the count clock:

1. Stop the counter by writing a 0 to CTRLx:CS.
2. Select a desired clock by writing the value to CTRLx:SEL.
3. Start the counter by writing a 1 to CTRLx:CS.

If an external clock on the CTCLK pin is selected in step 2, the timer increments the counter of the corresponding timer on the third rising edge of HCLK after a rising edge by CTCLK. The pulse length of CTCLK must be equal to or longer than, two HCLK periods plus the setup and hold time (see the Data Sheet for timing information). Shorter pulses can cause incorrect counts. If CTCLK is not in phase with HCLK, inaccurate counts can occur.

Figure 15-2 shows the timing of CTCLK with respect to HCLK when the two are in phase. Figure 15-3 shows the timing of CTCLK with respect to HCLK when the two are not in phase.

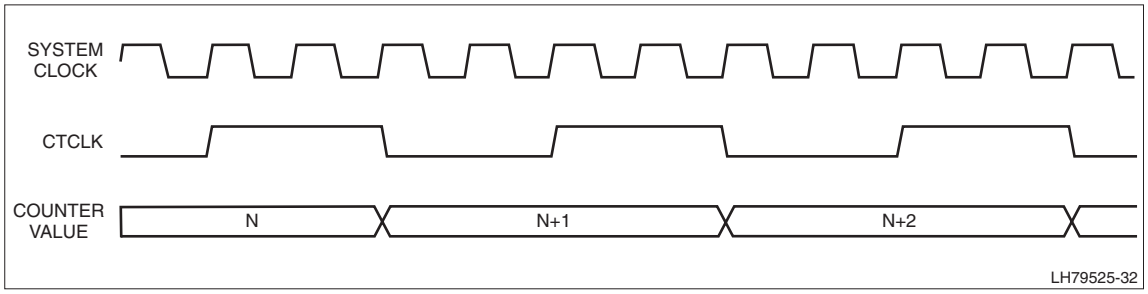


Figure 15-2. Count Clock Timing (HCLK in Phase with CTCLK)

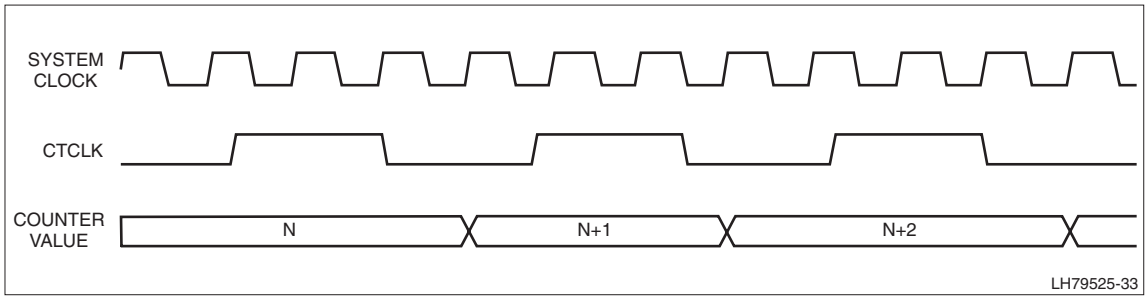


Figure 15-3. Count Clock Timing (HCLK not in Phase with CTCLK)

15.1.1 Counter Clear Upon Compare Match

A compare match occurs when the contents of the Timer Counter Register (CNTx) matches the value of the corresponding Timer Compare Register. When there is a compare match, one of two actions occurs, based on the state of the CNTx:TC.

- If TC is programmed to 0, the counter is not cleared and continues counting.
- If TC is programmed to 1, the counter is cleared on the rising edge of the internal count clock.

When using CMPx as a rising or falling edge trigger ('01' or '10' Output Value Select), the software must manually clear the compare output. The CMPx hardware provides a highly accurate edge interrupt.

This output is not automatically set to the opposing value; that is controlled through the programming of the output value select bits within the CMP_CAP_CTRL register when the interrupt is cleared. In this usage mode, the host can clear the output compare signal by programming the inverse value of the output select bits, set the compare register to 0x01, enable the counter, and wait for the interrupt. This interrupt will signify that the compare output is at the original opposing reference level. At this point the counter is ready to be used for another edge trigger interrupt.

15.1.2 Capture Signal Sampling

The capture signal causes the value of the timer to be captured and stored in the Timer Capture Register (TxCAPn) associated with the particular input pin being used. For example, to sample Timer 0 using a trigger on the CTC0A pin, that count would be stored in the T0CAPA register. Triggers occur on the rising edge of HCLK. The pulse width of a capture signal must be equal to or longer than, two HCLK periods plus the setup time for the signal to be correctly read in. After sampling, the external capture signal is synchronized to the rising edge of HCLK. This synchronization process takes two HCLK periods. After the external capture signal is synchronized, the value of the counter is stored in the appropriate TxCAPn Registers. The external capture signal triggering is edge-selectable and can use a rising or falling edge to capture the counter value.

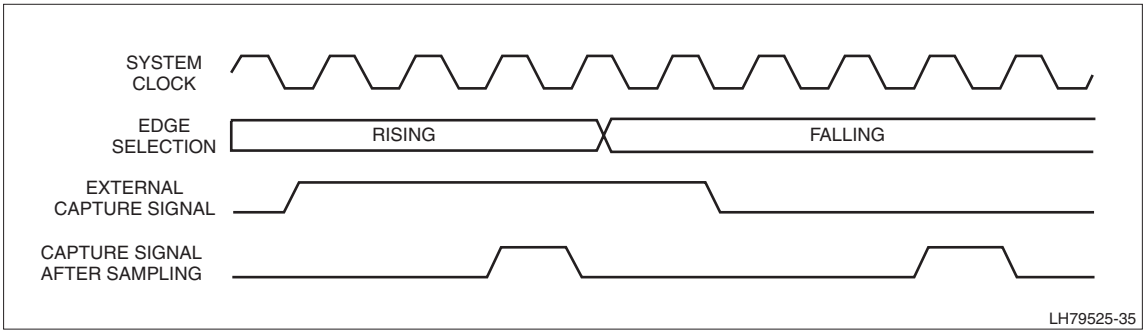


Figure 15-4. Capture Signal Synchronization Timing

15.1.3 PWM Mode

Any of the timers may be configured to implement a Pulse Width Modulator (PWM). In PWM mode, the signal is output on the CTCMPxA pin.

This mode uses a timer's two Timer Compare Registers (TxCMPn) to program the PWM period and duty cycle. TxCMP1 programs the PWM period, and TxCMP0 programs the PWM duty cycle. The period must always be larger than the duty cycle. Figure 15-5 illustrates this more clearly.

- The value in TxCMP1 Register + 1 is the period of the PWM.
- The value in TxCMP0 Register + 1 is the duty cycle of the PWM.

The PWM is clocked by the internal count clock, which is the prescaled HCLK.

To enable PWM Mode for a Timer, program the CTRLx:PWM bit to 1. With PWM Mode enabled, program the CTRLx:TC bit to 1. The TC bit causes the Timer Counter Register (CNTx) to reset to 0x0000 after its count value matches the value of the Timer Compare Register0 (TxCMP0).

The PWM output on the CTCMPxA pin can be programmed to active HIGH or active LOW polarity using the CTRLx:CMP[1:0] fields (see the CTRLx register description, and the example, for programming specifics).

In PWM Mode, PWM CTCMPxB remains LOW or HIGH, depending on the value programmed into the CTRLx:CMP0 field.

Figure 15-5 shows an example of PWM output signal timing. To implement the timing shown in this Figure, the following values are programmed into the registers.

- TxCMP1 = 0x0005 (Period of 6)
- TxCMP0 = 0x0001 (Duty Cycle of 2; 'OFF TIME' in Figure 15-5)

Timer 0 settings:

- CMP_CAP_CTRL:PWM = 1 (Enable PWM mode)
- CMP_CAP_CTRL:TC = 1 (Counter Clear Mode)
- CMP_CAP_CTRL:CMP1 = 01 (Active HIGH PWM polarity)
- CMP_CAP_CTRL:CMP0 = 10 (Active HIGH PWM polarity)

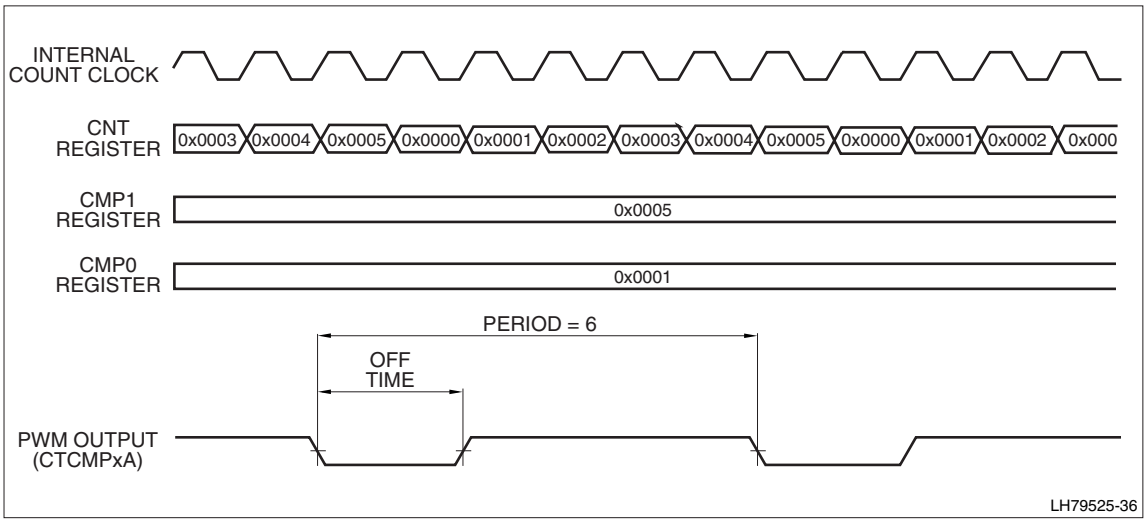


Figure 15-5. PWM Output Signal Timing

15.1.3.1 Timer Interrupts

The timer interrupts are:

- Timer 0 Combined Interrupt — a combined interrupt formed by the logical OR of the two compare, five capture, and one overflow interrupts in Timer 0.
- Timer 1 Combined Interrupt — a combined interrupt formed by the logical OR of the two compare, two capture, and one overflow interrupts in Timer 1.
- Timer 2 Combined Interrupt — a combined interrupt formed by the logical OR of the two compare, two capture, and one overflow interrupts in Timer 2.

If an individual interrupt is enabled and the corresponding interrupt condition (compare, capture, or overflow) occurs, a combined interrupt also occurs. Once the interrupt condition occurs, the combined Interrupt Output signal is asserted active to 1. It remains active until all compare, capture, and overflow interrupts are cleared in the appropriate Status Register, or disabled.

Interrupts can be individually enabled or disabled in the respective timer's INTENx register. The STATUSx registers allow software to read the status of each interrupt. Software can logically AND the STATUSx with the INTENx register to ascertain which enabled interrupts are asserted to the VIC.

15.2 Register Reference

This section describes the location and programming of the Timer registers.

15.2.1 Memory Map

Register offsets in Table 15-1 are relative to the Timer base address 0xFFFC4000.

Table 15-1. Timer 0 Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	CTRL0	Timer 0 Control Register
0x04	CMP_CAP_CTRL	Timer 0 Compare/Capture Control Register
0x08	INTEN0	Timer 0 Interrupt Control Register
0x0C	STATUS0	Timer 0 Status Register
0x10	CNT0	Timer 0 Counter Register
0x14	T0CMP0	Timer 0 Compare Register 0
0x18	T0CMP1	Timer 0 Compare Register 1
0x1C	T0CAPA	Timer 0 Capture Register A
0x20	T0CAPB	Timer 0 Capture Register B
0x24	T0CAPC	Timer 0 Capture Register C
0x28	T0CAPD	Timer 0 Capture Register D
0x2C	T0CAPE	Timer 0 Capture Register E

Table 15-2. Timer 1 Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x30	CTRL1	Timer 1 Control Register
0x34	INTEN1	Timer 1 Interrupt Control Register
0x38	STATUS1	Timer 1 Status Register
0x3C	CNT1	Timer 1 Counter Register
0x40	T1CMP0	Timer 1 Compare Register 0
0x44	T1CMP1	Timer 1 Compare Register 1
0x48	T1CAPA	Timer 1 Capture Register A
0x4C	T1CAPB	Timer 1 Capture Register B

Table 15-3. Timer 2 Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x50	CTRL2	Timer 2 Control Register
0x54	INTEN2	Timer 2 Interrupt Control Register
0x58	STATUS2	Timer 2 Status Register
0x5C	CNT2	Timer 2 Counter Register
0x60	T2CMP0	Timer 2 Compare Register 0
0x64	T2CMP1	Timer 2 Compare Register 1
0x68	T2CAPA	Timer 2 Capture Register A
0x6C	T2CAPB	Timer 2 Capture Register B

15.2.2 Register Descriptions

15.2.2.1 Timer 0 Control Register (CTRL0)

This register allows programming the clock divisor, as well as starting/stopping, and clearing the timer count value.

Table 15-4. CTRL0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											SEL			CS	CCL
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x00															

Table 15-5. CTRL0 Register Definitions

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading this field returns 0. Write the reset value.
4:2	SEL	**Timer 0 Clock Select** Specifies the timer clock divisor. The timer must be stopped (with the CS bit) before programming the divisor. 000 = HCLK/2 001 = HCLK/4 010 = HCLK/8 011 = HCLK/16 100 = HCLK/32 101 = HCLK/64 110 = HCLK/128 111 = CTCLK
1	CS	**Start/Stop Timer 0 Count** Specifies whether Timer 0 count is stopped or started. This bit must be programmed to 0 before programming the SEL bit. For more information, see Section 15.1.1. 1 = Starts Timer 0 0 = Stops Timer 0
0	CCL	**Timer 0 Count Clear** Programming a 1 clears the timer count value. This bit always reads as 0. 1 = Clears CNT0 contents to 0x0000 0 = Ignored; no effect

15.2.2.2 Timer 0 Compare/Capture Control Register (CMP_CAP_CTRL0)

CMP_CAP_CTRL0 allows programming the operating modes of Timer 0.

Table 15-6. CMP_CAP_CTRL0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	PWM	TC	CMP1		CMP0		CAPE		CAPD		CAPC		CAPB		CAPA	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x04															

Table 15-7. CMP_CAP_CTRL0 Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15	PWM	**PWM Output** Allows CTCMP0A to be used as a PWM output. This is done by configuring this bit as well as other bits in this register. Refer to Section 15.1.3 for a complete explanation. 1 = Output pin CTCMP0A is in PWM Mode 0 = Output pin CTCMP0A is in Normal Mode and only uses the CMP0 Register
14	TC	**Timer 0 Counter Operation** Programs Timer 0 as a free running counter or as an interval timer. When 1, the counter clears upon matching the T0CMP1 Register. Refer to Section 15.1.1 for a complete explanation. 1 = Clears counter when CNT0 for Timer 0 and T0CMP1 for Timer 0 match 0 = Inhibits counter clear (operates as free running counter)
13:12	CMP1	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on the CTCMP0B pin when the CNT0 Register matches T0CMP1. 00 = No change occurs to CTCMP0B 01 = Output 0 to CTCMP0B 10 = Output 1 to CTCMP0B 11 = Toggle the output to CTCMP0B PWM Operation: 00 = Invalid 01 = Active HIGH PWM output polarity 10 = Active LOW PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.

Table 15-7. CMP_CAP_CTRL0 Register Definitions

BITS	NAME	DESCRIPTION
11:10	CMP0	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on CTCMP0A when the CNT0 Register matches the T0CMP0 Register. 00 = No change occurs to the output CTCMP0A 01 = Output 0 to CTCMP0A 10 = Output 1 to CTCMP0A 11 = Toggle the output to CTCMP0A PWM Operation: 00 = Invalid 01 = Active LOW PWM output polarity 10 = Active HIGH PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.
9:8	CAPE	**Input Edge Select** Selects the edge used as the capture trigger on the CTCAP0E pin. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP0E is ignored 01 = Rising edge of CTCAP0E 10 = Falling edge of CTCAP0E 11 = Both edges of CTCAP0E
7:6	CAPD	**Input Edge Select** Selects the edge used as the capture trigger on the CTCAP0D pin. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP0D is ignored 01 = Rising edge of CTCAP0D 10 = Falling edge of CTCAP0D 11 = Both edges of CTCAP0D
5:4	CAPC	**Input Edge Select** Selects the edge used as the capture trigger on the CTCAP0C pin. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP0C is ignored 01 = Rising edge of CTCAP0C 10 = Falling edge of CTCAP0C 11 = Both edges of CTCAP0C
3:2	CAPB	**Input Edge Select** Selects the edge used as the capture trigger on the CTCAP0B pin. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP0B is ignored 01 = Rising edge of CTCAP0B 10 = Falling edge of CTCAP0B 11 = Both edges of CTCAP0B
1:0	CAPA	**Input Edge Select** Selects the edge used as the capture trigger on the CTCAP0A pin. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP0A is ignored 01 = Rising edge of CTCAP0A 10 = Falling edge of CTCAP0A 11 = Both edges of CTCAP0A

15.2.2.3 Timer 0 Interrupt Control Register (INTEN0)

This register allows software to enable and disable individual interrupts as needed.

Table 15-8. INTEN0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								CAPE_EN	CAPD_EN	CAPC_EN	CAPB_EN	CAPA_EN	CMP1_EN	CMP0_EN	OVF_EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x08															

Table 15-9. INTEN0 Register Definitions

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7	CAPE_EN	**Timer 0 Interrupt Enable During Capture E Operation** 1 = Interrupt enabled for capture E 0 = Interrupt disabled for capture E
6	CAPD_EN	**Timer 0 Interrupt Enable During Capture D Operation** 1 = Interrupt enabled for capture D 0 = Interrupt disabled for capture D
5	CAPC_EN	**Timer 0 Interrupt Enable During Capture C Operation** 1 = Interrupt enabled for capture C 0 = Interrupt disabled for capture C
4	CAPB_EN	**Timer 0 Interrupt Enable During Capture B Operation** 1 = Interrupt enabled for capture B 0 = Interrupt disabled for capture B
3	CAPA_EN	**Timer 0 Interrupt Enable During Capture A Operation** 1 = Interrupt enabled for capture A 0 = Interrupt disabled for capture A
2	CMP1_EN	**Timer 0 Interrupt Enable Upon Compare 1** 1 = Interrupt enabled for compare 1 0 = Interrupt disabled for compare 1
1	CMP0_EN	**Timer 0 Interrupt Enable Upon Compare 0** 1 = Interrupt enabled for compare 0 0 = Interrupt disabled for compare 0
0	OVF_EN	**Timer 0 Interrupt Overflow Enable** 1 = Interrupt enabled for counter overflows 0 = Interrupt disabled for counter overflows

15.2.2.4 Timer 0 Status Register (STATUS0)

The Status Register bits contain the raw interrupt status of the various interrupt generators. Raw interrupts reflect the state of the interrupt, whether or not it is enabled. To clear a status bit, write a 1 to that bit. This action also clears the corresponding interrupt, with the following exception: if the timer is stopped and the Timer 0 Compare Register (T0CMP0 or T0CMP1) value matches the Timer 0 Counter Register (CNT0), the corresponding status bit cannot be cleared until either the Timer 0 Compare Register or the Timer 0 Counter Register value is changed.

Writing a 0 to any of the status bits has no effect. Similarly, writing a 1 to a bit that currently reads as 0 (no interrupt assertion) does not affect the Status Register or interrupt.

Table 15-10. STATUS0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								CAPE_ST	CAPD_ST	CAPC_ST	CAPB_ST	CAPA_ST	CMP1_ST	CMP0_ST	OVF_ST
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x0C															

Table 15-11. STATUS0 Register Definitions

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7	CAPE_ST	**Timer 0 Capture E Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
6	CAPD_ST	**Timer 0 Capture D Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
5	CAPC_ST	**Timer 0 Capture C Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
4	CAPB_ST	**Timer 0 Capture B Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
3	CAPA_ST	**Timer 0 Capture A Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect

Table 15-11. STATUS0 Register Definitions (Cont'd)

BITS	NAME	DESCRIPTION
2	CMP1_ST	**Timer 0 Compare 1 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
1	CMP0_ST	**Timer 0 Compare 0 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
0	OVF_ST	**Timer 0 Overflow Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect

15.2.2.5 Timer 0 Counter Register (CNT0)

The CNT0 Register is a 16-bit, Read/Write up counter. The counter can be read from, or written to while it is operating. As a result, counts can be read at any time or the current count can be changed.

Table 15-12. CNT0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM0CNT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x10															

Table 15-13. CNT0 Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM0CNT	**Timer 0 Count** Value of the 16-bit up counter.

15.2.2.6 Timer 0 Compare Registers (T0CMPn)

There are two T0CMPn Registers for Timer 0. They are designated:

- T0CMP0
- T0CMP1

Each register is a 16-bit, read/write register. Contents of these registers are compared continuously with the counter CNT0. When both register and counter values match, the timer responds as programmed in the CMP_CAP_CTRL register.

Table 15-14. T0CMPn Registers

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM0CMP															
RESET	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	CMP0: 0xFFFC4000 + 0x14 CMP1: 0xFFFC4000 + 0x18															

Table 15-15. T0CMPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM0CMP	**Timer 0 Compare** 16-bit compare register value.

15.2.2.7 Timer 0 Capture Registers (CAPn)

There are five CAPn Registers for Timer 0. They are designated:

- CAPA
- CAPB
- CAPC
- CAPD
- CAPE

Each register is a 16-bit, Read Only register. When a capture condition occurs, the contents of the counter CNT0 are stored into the associated Capture Register. Capture Registers correspond to the input signals CTCAP0A through CTCAP0E, respectively. The edge of the input signal used to trigger the capturing operation is selected with the CMP_CAP_CTRL Register.

Table 15-16. CAPn Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CAPTURE0															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	CAPA: 0xFFFC4000 + 0x1C CAPB: 0xFFFC4000 + 0x20 CAPC: 0xFFFC4000 + 0x24 CAPD: 0xFFFC4000 + 0x28 CAPE: 0xFFFC4000 + 0x2C															

Table 15-17. CAPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	CAPTURE0	**Timer 0 Capture Register** 16-bit capture register value.

15.2.2.8 Timer 1 Control Register (CTRL1)

This register allows programming various functions, including PWM Mode, clock selection, and starting/stopping Timer 1.

Table 15-18. CTRL1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///	PWM	TC	CMP1		CMP0		CAPB		CAPA		SEL			CS	CCL
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x30															

Table 15-19. CTRL1 Register Definitions

BITS	NAME	DESCRIPTION
31:15	///	**Reserved** Reading this field returns 0. Write the reset value.
14	PWM	**PWM Output** This bit allows the use of CTCMP1A as a PWM output. This is done by programming this bit as well as other bits in this register. Refer to Section 15.1.3 for a complete explanation and an example. 0 = Output CTCMP1A is normal and works only with the T0CMP1 Register. 1 = Output CTCMP1A is in PWM Mode.
13	TC	**Timer 1 Operation** This bit determines whether Timer 1 counter is to operate as either a free running counter or as an interval timer. When 1, the counter clears upon matching CMP1 for Timer 1. This operation is only available with the CMP1 Register for Timer 1. Refer to Section 15.1.1 for a complete explanation. 0 = Inhibit counter clear (operates as free running counter). 1 = Clear counter when CNT1 for Timer 1 matches T1CMP1 for Timer 1.
12:11	CMP1	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on the CTCMP1B pin when the CNT1 Register matches T1CMP1. 00 = No change occurs to CTCMP1B 01 = Output 0 to CTCMP1B 10 = Output 1 to CTCMP1B 11 = Toggle the output to CTCMP1B PWM Operation: 00 = Invalid 01 = Active HIGH PWM output polarity 10 = Active LOW PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.

Table 15-19. CTRL1 Register Definitions (Cont'd)

BITS	NAME	DESCRIPTION
10:9	CMP0	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on CTCMP1A when the CNT1 Register matches the T1CMP0 Register. 00 = No change occurs to the output CTCMP1A 01 = Output 0 to CTCMP1A 10 = Output 1 to CTCMP1A 11 = Toggle the output to CTCMP1A PWM Operation: 00 = Invalid 01 = Active LOW PWM output polarity 10 = Active HIGH PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.
8:7	CAPB	**Input Edge Select** Selects the rising edge, falling edge, both edges, or ignores all changes of the input signal that is used as the capture trigger. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP1B is ignored 01 = Rising edge of CTCAP1B 10 = Falling edge of CTCAP1B 11 = Both edges of CTCAP1B
6:5	CAPA	**Input Edge Select** Selects the rising edge, falling edge, both edges, or ignores all changes of the input signal that is used as the capture trigger. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP1A is ignored 01 = Rising edge of CTCAP1A 10 = Falling edge of CTCAP1A 11 = Both edges of CTCAP1A
4:2	SEL	**Count Clock Select** Specifies the timer clock divisor. The timer must be stopped (with the CS bit) before programming the divisor. 000 = HCLK/2 001 = HCLK/4 010 = HCLK/8 011 = HCLK/16 100 = HCLK/32 101 = HCLK/64 110 = HCLK/128 111 = CTCLK
1	CS	**Start/Stop Timer 1** Specifies whether Timer 1 count is stopped or started. This bit must be programmed to 0 before programming the SEL field in this register. For more information, see Section 15.1.1. 0 = Stop Timer 1 1 = Start Timer 1
0	CCL	**Timer 1 Count Clear** Programming a 1 clears the timer count value. This bit always reads as 0. 1 = Clears CNT1 contents to 0x0000 0 = Ignored; no effect

15.2.2.9 Timer 1 Interrupt Control Register (INTEN1)

This register allows software to enable and disable individual interrupts as needed.

Table 15-20. INTEN1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											CAPB_EN	CAPA_EN	CMP1_EN	CMP0_EN	OVF_EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x34															

Table 15-21. INTEN1 Register Definitions

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading this field returns 0. Write the reset value.
4	CAPB_EN	**Timer 1 Interrupt Enable During Capture 1 Operation** 1 = Interrupt enabled for capture B 0 = Interrupt disabled for capture B
3	CAPA_EN	**Timer 1 Interrupt Enable During Capture A Operation** 1 = Interrupt enabled for capture A 0 = Interrupt disabled for capture A
2	CMP1_EN	**Timer 1 Interrupt Enable Upon Compare 1** 1 = Interrupt enabled for compare 1 0 = Interrupt disabled for compare 1
1	CMP0_EN	**Timer 1 Interrupt Enable Upon Compare 0** 1 = Interrupt enabled for compare 0 0 = Interrupt disabled for compare 0
0	OVF_EN	**Timer 1 Interrupt Overflow Enable** 1 = Interrupt enabled for counter overflows 0 = Interrupt disabled for counter overflows

15.2.2.10 Timer 1 Status Register (STATUS1)

The Status Register bits contain the raw interrupt status of the various interrupt generators. Raw interrupts reflect the state of the interrupt, whether or not it is enabled. To clear a status bit, write a 1 to that bit. This action also clears the corresponding interrupt, with the following exception: if the timer is stopped and the Timer 1 Compare Register (T1CMP0 or T1CMP1) value matches the Timer 1 Counter Register (CNT1), the corresponding status bit cannot be cleared until either the Timer 1 Compare Register or CNT1 value is changed.

Writing a 0 to any of the status bits has no effect. Similarly, writing a 1 to a bit that currently reads as 0 (no interrupt assertion) does not affect the Status Register or interrupt.

Table 15-22. STATUS1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											CAPB_ST	CAPA_ST	CMP1_ST	CMP0_ST	OVF_ST
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x38															

Table 15-23. STATUS1 Register Definitions

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading this field returns 0. Write the reset value.
4	CAPB_ST	**Timer 1 Capture B Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
3	CAPA_ST	**Timer 1 Capture A Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
2	CMP1_ST	**Timer 1 Compare 1 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
1	CMP0_ST	**Timer 1 Compare 0 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
0	OVF_ST	**Timer 1 Overflow Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect

15.2.2.11 Timer 1 Counter Register (CNT1)

The CNT1 Register is a 16-bit, Read/Write up counter. The counter can be read or written to while it is operating. As a result, counts can be read at any time or the current count can be changed.

Table 15-24. CNT1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM1CNT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x3C															

Table 15-25. CNT1 Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM1CNT	**Timer 1 Count** 16-bit up counter value.

15.2.2.12 Timer 1 Compare Registers (T1CMPn)

There are two CMP(n) Registers for Timer 1. They are designated:

- T1CMP0
- T1CMP1

Each register is a 16-bit, Read/Write register. Contents of these registers are compared continuously with the counter CNT1. When both register and counter values match, the timer behaves as programmed in the CTRL1 register.

Table 15-26. T1CMPn Registers

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM1CMP															
RESET	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	T1CMP0: 0xFFFC4000 + 0x40 T1CMP1: 0xFFFC4000 + 0x44															

Table 15-27. T1CMPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM1CMP	**Timer 1 Compare** 16-bit Compare Register Value.

15.2.2.13 Timer 1 Capture Registers (T1CAPn)

There are two T1CAPn Registers for Timer 1. They are designated:

- T1CAPA
- T1CAPB

Each register is a 16-bit, Read Only register. When a capture condition occurs, the contents of the counter CNT1 are stored into the associated Capture Register. Capture Registers correspond to the input signals CTCAP1A through CTCAP1B, respectively. The edge of the input signal used to trigger the capturing operation is determined by programming the CTRL1 Register.

Table 15-28. T1CAPn Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CAPTURE1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	T1CAPA: 0xFFFC4000 + 0x48 T1CAPB: 0xFFFC4000 + 0x4C															

Table 15-29. T1CAPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	CAPTURE1	**Timer 1 Capture** 16-bit Capture Register Value.

15.2.2.14 Timer 2 Control Register (CTRL2)

This register allows programming various functions of the timer, including PWM Mode, clock selection, and start/stop.

Table 15-30. CTRL2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///	PWM	TC	CMP1		CMP0		CAPB		CAPA		SEL			CS	CCL
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x50															

Table 15-31. CTRL2 Register Definitions

BITS	NAME	DESCRIPTION
31:15	///	**Reserved** Reading this field returns 0. Write the reset value.
14	PWM	**PWM Output** This bit allows the use of CTCMP2A as a PWM output. This is done by properly programming this bit as well as other bits in this register. Refer to Section 15.1.3 for a complete explanation of this feature. 0 = Output CTCMP2A is normal and works only with the CMP2 Register. 1 = Output CTCMP2A is in PWM Mode.
13	TC	**Timer 2 Counter Operation** This bit determines whether the counter is to operate as a free running counter or interval counter. When 1, the counter clears upon matching CMP1 for Timer 2. This operation is only available with the CMP1 Register for Timer 2. 0 = Inhibit counter clear (operates as free running counter). 1 = Clear counter when CNT2 for Timer 2 matches CMP1 for Timer 2.
12:11	CMP1	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on the CTCMP2B pin when the CNT2 Register matches T2CMP1. 00 = No change occurs to CTCMP2B 01 = Output 0 to CTCMP2B 10 = Output 1 to CTCMP2B 11 = Toggle the output to CTCMP2B PWM Operation: 00 = Invalid 01 = Active HIGH PWM output polarity 10 = Active LOW PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.

Table 15-31. CTRL2 Register Definitions (Cont'd)

BITS	NAME	DESCRIPTION
10:9	CMP0	**Output Value Select** Timer/Counter Operation: Programs the value (when a compare match occurs) output on CTCMP2A when the CNT2 Register matches the T2CMP0 Register. 00 = No change occurs to the output CTCMP2A 01 = Output 0 to CTCMP2A 10 = Output 1 to CTCMP2A 11 = Toggle the output to CTCMP2A PWM Operation: 00 = Invalid 01 = Active LOW PWM output polarity 10 = Active HIGH PWM output polarity 11 = Invalid **IMPORTANT**: CMP1 and CMP0 *must* be programmed to the same polarity.
8:7	CAPB	**Input Edge Select** Selects the rising edge, falling edge, both edges, or ignores all changes of the input signal that is used as the capture trigger. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP2B is ignored 01 = Rising edge of CTCAP2B 10 = Falling edge of CTCAP2B 11 = Both edges of CTCAP2B
6:5	CAPA	**Input Edge Select** Selects the rising edge, falling edge, both edges, or ignores all changes of the input signal that is used as the capture trigger. This field and the capture function are inactive in PWM mode. 00 = Capture input CTCAP2A is ignored 01 = Rising edge of CTCAP2A 10 = Falling edge of CTCAP2A 11 = Both edges of CTCAP2A
4:2	SEL	**Count Clock Select** Specifies the timer clock divisor. The timer must be stopped (with the CS bit) before programming the divisor. 000 = HCLK/2 001 = HCLK/4 010 = HCLK/8 011 = HCLK/16 100 = HCLK/32 101 = HCLK/64 110 = HCLK/128 111 = CTCLK
1	CS	**Start/Stop Timer 2** Specifies whether Timer 2 count is stopped or started. This bit must be programmed to 0 before programming the SEL field in this register. For more information, see Section 15.1.1. 0 = Stop Timer 2 1 = Start Timer 2
0	CCL	**Timer 2 Count Clear** Programming a 1 clears the timer count value. This bit always reads as 0. 1 = Clears CNT2 contents to 0x0000 0 = Ignored; no effect

15.2.2.15 Timer 2 Interrupt Control Register (INTEN2)

This register allows software to enable and disable individual interrupts as needed.

Table 15-32. INTEN2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											CAPB_EN	CAPA_EN	CMP1_EN	CMP0_EN	OVF_EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x54															

Table 15-33. INTEN2 Register Definitions

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading this field returns 0. Write the reset value.
4	CAPB_EN	**Timer 2 Interrupt Enable During Capture B Operation** 1 = Interrupt enabled for capture B 0 = Interrupt disabled for capture B
3	CAPA_EN	**Timer 2 Interrupt Enable During Capture A Operation** 1 = Interrupt enabled for capture A 0 = Interrupt disabled for capture A
2	CMP1_EN	**Timer 2 Interrupt Enable Upon Compare 1** 1 = Interrupt enabled for compare 1 0 = Interrupt disabled for compare 1
1	CMP0_EN	**Timer 2 Interrupt Enable Upon Compare** 1 = Interrupt enabled for compare 0 0 = Interrupt disabled for compare 0
0	OVF_EN	**Timer 2 Interrupt Overflow Enable** 1 = Interrupt enabled for counter overflows 0 = Interrupt disabled for counter overflows

15.2.2.16 Timer 2 Status Register (STATUS2)

The Status Register bits contain the raw interrupt status of the various interrupt generators. Raw interrupts reflect the state of the interrupt, whether or not it is enabled. To clear a status bit, write a 1 to that bit. This action also clears the corresponding interrupt, with the following exception: if the timer is stopped and the Timer 2 Compare Register (T2CMP0 or T2CMP1) value matches the Timer 2 Counter Register (CNT2), the corresponding status bit cannot be cleared until either the Timer 2 Compare Register or the Timer 2 Counter Register value is changed.

Writing a 0 to any of the status bits has no effect. Similarly, writing a 1 to a bit that currently reads as 0 (no interrupt assertion) does not affect the Status Register or interrupt.

Table 15-34. STATUS2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///											CAPB_ST	CAPA_ST	CMP1_ST	CMP0_ST	OVF_ST
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x58															

Table 15-35. STATUS2 Register Definitions

BITS	NAME	DESCRIPTION
31:5	///	**Reserved** Reading this field returns 0. Write the reset value.
4	CAPB_ST	**Timer 2 Capture B Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
3	CAPA_ST	**Timer 2 Capture A Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
2	CMP1_ST	**Timer 2 Compare 1 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
1	CMP0_ST	**Timer 2 Compare 0 Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect
0	OVF_ST	**Timer 2 Overflow Status** 1 = Read: Interrupt asserted; Write: Clear interrupt 0 = Read: No interrupt asserted; Write: No effect

15.2.2.17 Timer 2 Counter Register (CNT2)

The CNT2 Register is a 16-bit, read/write up counter. The counter can be read or written to while it is operating. As a result, counts can be read at any time or the current count can be changed.

Table 15-36. CNT2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM2CNT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFC4000 + 0x5C															

Table 15-37. CNT2 Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM2CNT	**Timer 2 Counter** 16-bit up-counter value.

15.2.2.18 Timer 2 Compare Registers (T2CMPn)

There are two T2CMPn Registers for Timer 2. They are designated:

- T2CMP0
- T2CMP1

Each register is a 16-bit, Read/Write register. Contents of these registers are compared continuously with the counter CNT2. When both register and counter values match, the timer behaves as programmed in the CTRL2 register.

Table 15-38. T2CMPn Registers

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	TM2CMP															
RESET	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	T2CMP0: 0xFFFC4000 + 0x60 T2CMP1: 0xFFFC4000 + 0x64															

Table 15-39. T2CMPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	TM2CMP	**Timer 2 Compare** 16-bit Compare Register Value.

15.2.2.19 Timer 2 Capture Registers (T2CAPn)

There are two T2CAPn Registers for Timer 2. They are designated:

- T2CAPA
- T2CAPB

Each register is a 16-bit, Read Only register. When a capture condition occurs, the contents of the counter CNT2 are stored into the associated Capture Register. Capture Registers correspond to the input signals CTCAP2A through CTCAP2B, respectively. The edge of the input signal used to trigger the capturing operation is determined by programming the CTRL Register.

Table 15-40. T2CAPn Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CAPTURE2															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	T2CAPA: xFFFC4000 + 0x68 T2CAPB: xFFFC4000 + 0x6C															

Table 15-41. T2CAPn Register Definitions

BITS	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	CAPTURE2	**Timer 2 Capture** 16-bit Capture Register Value.

Chapter 16
UARTs

The LH79524/LH79525 contains three UARTs, UART[2:0]. The UARTs feature:

- Character Length: Programmable number of data bits per character (5, 6, 7, or 8). Even, odd, stick, or no-parity bit generation and detection. 1 or 2 Stop bit generation.
- Optional Nine-bit Mode to tag and recognize characters as either data or address
- FIFOs: The transmit FIFO is 9 bits wide and the receive FIFO is 12 bits wide. Each has programmable service 'trigger levels' (empty/full, 1/8, 1/4, 1/2, 3/4, and 7/8) and overrun protection. The receive FIFO is 12 bits wide to accommodate 8 bits of data and receive flags (framing, parity/9th bit, break error, or overrun) that are placed in the receive FIFO. The transmit FIFO is 9 bits wide to accommodate 8 data bits plus the optional Nine-bit Mode tag. The transmit and receive FIFOs can be disabled to act like a one-byte holding register. Both FIFOs are 32 entries deep.
- Programmable baud rate generator: This enables division of the UART input clock by 16 to 65535 × 16 and generates an internal clock that is common to both transmit and receive portions of the UART. The divisor can be a fractional number.
- Interrupts: The UART can issue an interrupt on transmit and receive FIFO watermarks, on errors, on a received address in Nine-bit Mode, and on receiver timeout. Each interrupt can be individually enabled and masked.
- DMA: Support for Direct Memory Access (DMA); UART0 only.
- Line break: Generate and detect breaks using UART transactions.
- Loopback testing: Programmed by writing the appropriate values to the control registers. Data transmitted on UARTTXD will then be received on the UARTRXD input.
- Independent Clocks: UART clock operates asynchronously to the system clock.
- IrDA support
- RTS and CTS modem control lines on UART0

Figure 16-1 shows a block diagram of UART0, UART1, and UART2.

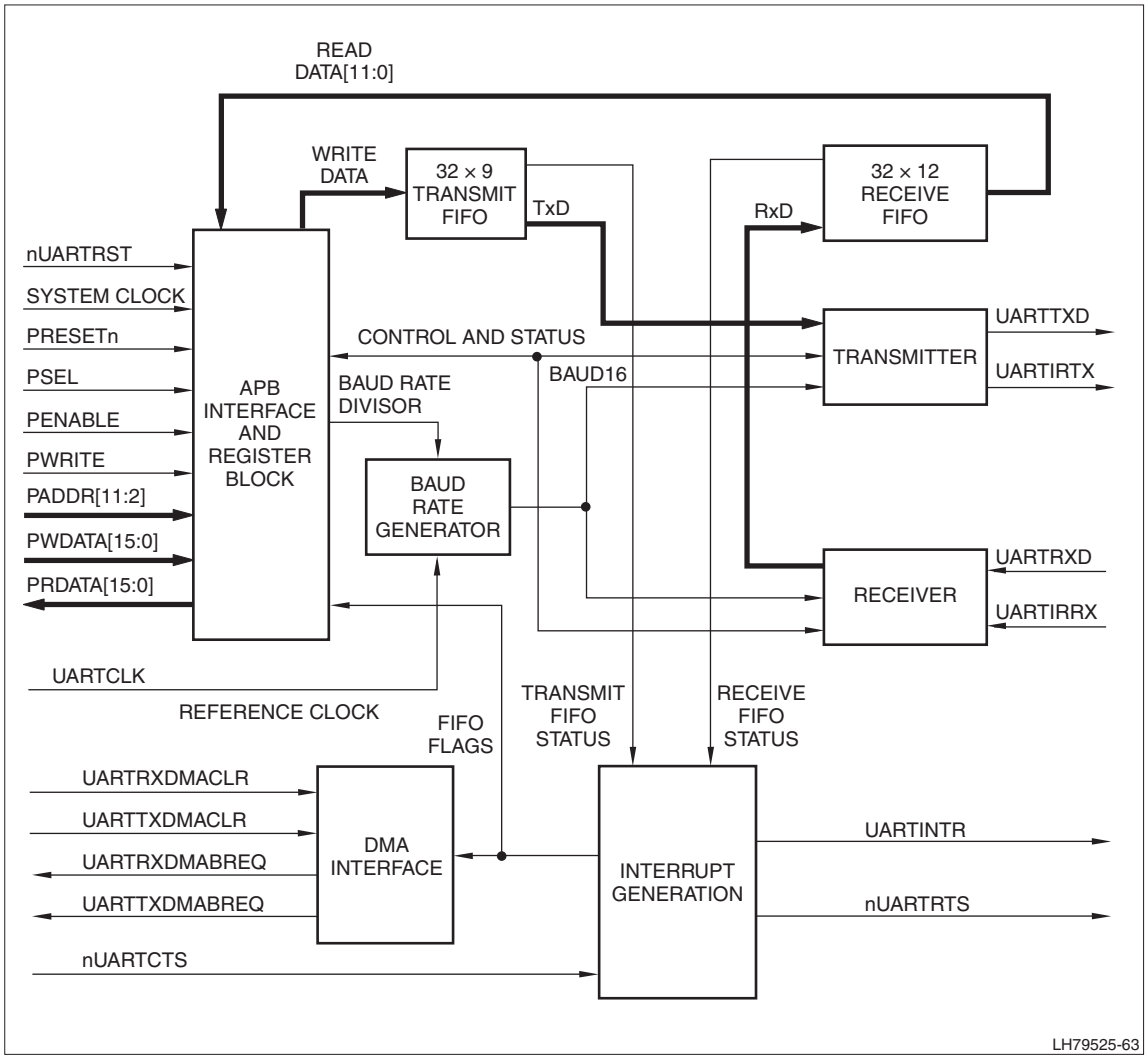


Figure 16-1. UART0, UART1, and UART2 Block Diagram

16.1 Theory of Operation

The three UARTs, UART0, UART1, and UART2, offer similar functionality to the industry-standard 16C550. They perform serial-to-parallel conversion on data received from a peripheral device and parallel-to-serial conversion on data transmitted to the UART. The CPU reads and writes data and control/status information through the AMBA APB interface. The transmit and receive paths are buffered with internal FIFO memories that support programmable 'watermark levels', and overrun protection. These FIFO memories enable up to 32 entries to be stored independently in both transmit and receive modes. All UART Control and Status Registers can be accessed through the APB.

16.1.1 Transmitting Data

When the UART is programmed to transmit and enabled, writing data to the transmit FIFO:

- Causes the UART to start transmitting a data frame with the parameters indicated in the UARTLCR_H Register. Data continues to be transmitted until the transmit FIFO is empty, as indicated by the Transmit FIFO Empty Flag (UARTFR:TXFE).
- Causes the UARTFR:BUSY bit to be asserted. This bit remains HIGH until the transmit FIFO is empty and the last character, including the Stop bits, has been sent.

16.1.2 Receive Data Frame

A UART receive data frame structure consists of:

- A LOW Start bit that indicates the beginning of the frame.
- Five to eight data bits.
- An optional parity error/address received bit, which can be used with available hardware for parity-error checking or can be used as an address received bit.
- One or two Stop bits, which indicate the end of the frame. The number of Stop bits is programmable using the UARTCR_H:STP2 bit.

The UART receiver is in the Idle state (i.e., the input is 1) from the time a Stop bit is sent until the time the next Start bit is received. When the receiver receives an entire frame, the UART transfers the received data and the frame status to the receive FIFO. This buffer can have a depth of either 32 12-bit words (FIFO Mode) or one 12-bit word (Character Mode), as set with the UARTLCR_H:FEN bit.

The Start bit works with the UART bit clock to synchronize the receiver with the source driving the receiver. When the source drives the receiver input from the Idle state to 0, the receiver waits 7/16ths of a bit period, then samples the input three times:

- Once at 7/16ths of the bit period
- Once at 8/16ths of the bit period
- Once at 9/16ths of the bit period.

If the input is 0 for at least two of the three samples, the UART recognizes a Start bit. After recognizing the Start bit, the receiver repeats the following sequence until all data bits, any parity bit, and all Stop bits are detected:

1. Wait 14/16ths of a bit period, then sample the input.
2. Wait 1/16th of a bit period, then sample the input.
3. Wait 1/16th of a bit period, then sample the input.
4. Choose the majority value of the three samples as the input value for that bit period.

After recognizing the final Stop bit, the UART stores the received data frames and associated status bits in the receiver FIFO.

16.1.3 Nine-bit Mode

In Nine-bit Mode, the parity bit of the character frame is used to identify the message as containing an address or data (parity is not calculated).

Enable Nine-bit Mode by setting bit 9 (9BIT) of UARTLCR_H. Then set or clear bit 8 (ADDTX) to tag the next character written to UARTDR as address (ADDTX=1) or data (ADDTX=0). After a write to UARTDR, ADDTX automatically resets to 0 so that subsequent writes to UARTDR will tag the character as data.

When a character is written to the transmit FIFO, the value of ADDTX at the time the UARTDR was written is also stored with the character. This tags each character in the FIFO as either data or address.

When characters are transmitted out of the transmit FIFO in Nine-bit Mode, the value of ADDTX for each character is substituted for the parity bit in the outgoing frame.

During Nine-bit Mode reception, the Parity Error/Address Received bits in UARTDR, UARTRSR and UARTRIS are set if an address was received (the parity bit of the incoming frame is set). They are cleared if data was received (the parity bit of the incoming frame is clear). As with other error status bits, the UARTDR must be read before status is read.

16.1.4 Status Conditions

The UARTs adhere to these status conditions:

- If a UART fails to detect a 1 for all programmed Stop-bit periods following a data frame, the UART sets the framing-error status for that frame.
- When Nine-bit Mode is disabled, enabling parity-error detection causes the UART to compare the parity/address received bit in each frame with the parity required for the hardware. The UART sets the parity-error/address received status for each frame containing a parity error.
- When Nine-bit Mode is enabled, parity-error detection is disabled. The UART sets the parity-error/address received status for each incoming frame containing an address.
- A line break is 0 for all bits (Start bit, data bits, parity bit, and Stop bits). The UART sets the line-break status for each frame containing a line break.
- The Overrun Error bit (UARTRSR:OE) indicates some frames might have been lost immediately preceding the frame with the overrun status.

The overrun status is announced by both the Overrun Error bit (UARTRSR:OE) and an overrun-status bit in a frame in the receiver FIFO. If the receiver FIFO is full and another frame is received, the receiver enters the overrun state and the UART sets the overrun-error bit in the UARTRSR. The Overrun Error bit remains set until software writes a 1 to UARTRCR:OE.

While the receive FIFO remains full, additional data frames at the receive unit are lost and are not stored in the receive FIFO. When the UART can resume storing frames in the receiver FIFO, the receiver exits the overrun state. The overrun-status bit in the receive FIFO is set in the first frame stored after the overrun.

16.1.5 On-Chip DMA Capabilities

UART0 can be programmed to utilize the on-chip DMA to reduce processor bandwidth required to service UART activities. DMA functions support burst transfers on the receive channel, transmission channel, or both. When using DMA, the transfer size **must** be set to 8 bits.

- When DMA is enabled on the receive channel, a DMA request is issued when the receive FIFO reaches its programmed high water mark. Once the DMA block services the request, a new one is issued when the FIFO fills above its high water mark.
- When DMA is enabled on the transmit channel, a request is issued when the transmit channel FIFO falls below its low water mark. The request is reissued if the FIFO remains below that level when the DMA request has been serviced, or the next time that the FIFO falls below that level.

NOTE: After reaching the watermark value and generating and interrrupt when transferring Receive data to memory using DMA, data remains in the Receive FIFO; there will be one entry less than the setting of the watermark. For example, if the watermark is set at 8, seven data entries remain in the Receive FIFO after DMA completes. These elements must be transferred using programmed copying rather than DMA.

DMA requests are masked when the UART issues an error interrupt. If the UART is in the Character Mode, only the DMA Single Transfer Mode (transferring one character per DMA operation) can operate, since only one character can be transferred to or from the FIFOs at any time. As a result, the programmed watermark level is not relevant in Character Mode.

DMA requests and setup are handled by programming the DMA Controller. This is described in Chapter 5 of this User's Guide.

NOTE: Care must be used in selecting receive Watermark values when using DMA transfers. The Watermark must be no less than the number of bytes in a DMA Burst. Setting the Watermark to a lesser value than the DMA Burst size results in storing erraneous data.

16.1.6 Programming the SIR

Serial Infrared (SIR) functions are supported to speeds of 115.2 kbps, half-duplex. The Encoder/Decoder (SIR ENDEC) also supports normal 3/16 bit-durations and low-power bit-durations. For low-power mode bit-durations, the reference clock which is input to the UART can be divided by values from 1 to 512 (decimal), for use as the SIR baud clock. The SIR system is a half-duplex system; the SIR cannot receive while it is transmitting.

The IrDA SIR physical layer specifies a minimum 10 μs delay between transmission and reception. This delay must be implemented by software.

16.1.7 Hardware Flow Control

Hardware flow control is fully selectable, and allows control of the serial data flow by using the nUARTRTS0 output and nUARTCTS0 input signals. Enabling flow control pins is made in the MUXCTL6:PB1 and MUXCTL6:PB0 fields, which is described in Table 11-18.

When the RTS flow control is enabled, the nUARTRTS0 signal is asserted until the receive FIFO is filled up to the programmed watermark level. When the CTS flow control is enabled, the transmitter can only transmit data when the nUARTCTS signal is asserted.

The hardware control is selectable through bits 14 (RTSEN) and 15 (CTSEN) in the Control Register (UARTCR); see Section 16-22. Table 16-1 shows the bit settings to enable RTS and CTS flow control both simultaneously, and independently.

Table 16-1. Control bits to enable and disable hardware flow control

UARTCR:CTSEN	UARTCR:RTSEN	DESCRIPTION
1	1	Both RTS and CTS flow control enabled.
1	0	Only CTS flow control enabled.
0	1	Only RTS flow control enabled.
0	0	Both RTS and CTS flow control disabled.

16.1.7.1 RTS Flow Control

The RTS flow control logic is linked to the programmable receive FIFO watermark levels. When RTS flow control is enabled, the nUARTRTS0 is asserted until the receive FIFO is filled up to the watermark level. When the receive FIFO watermark level is reached, the nUARTRTS0 signal is deasserted, indicating that the FIFO is full and data transmission should cease after transmission of the current character.

The nUARTRTS0 signal is reasserted when the receive FIFO has been emptied to less than the watermark level. If RTS flow control is disabled and the UART is still enabled, data is received until the receive FIFO is full, or until no more data is transmitted to it.

16.1.7.2 CTS Flow Control

If CTS flow control is enabled, the transmitter checks the nUARTCTS0 signal before transmitting the next byte. If the nUARTCTS0 signal is asserted, it transmits the byte; otherwise transmission does not occur. The data continues to be transmitted while nUARTCTS0 is asserted, and the transmit FIFO is not empty. If the transmit FIFO is empty and the nUARTCTS0 signal is asserted, no data is transmitted.

If the nUARTCTS0 signal is deasserted and CTS flow control is enabled, the current character transmission completes before stopping. If CTS flow control is disabled and the UART is enabled, data continues to be transmitted until the transmit FIFO is empty.

16.1.8 Programming Control Registers

A UART must be disabled before any of the Control Registers are programmed. When the UART is disabled in the middle of transmission or reception, it completes the current character operation before stopping.

16.2 Interrupts

UART0, UART1, and UART2 each have a combined interrupt. The individual UART interrupt outputs are ORed together to produce the combined interrupt for that UART. Interrupt conditions within the combined interrupt are individually maskable. The Vectored Interrupt Controller (VIC) must be programmed before using the UART interrupts. Refer to Section 18.1.2 to program the VIC.

16.2.1 UARTINTR

The UARTINTR interrupt is the combined interrupt for each UART. It is asserted if one or more of the other interrupts are asserted.

16.3 Register Reference

This section provides the UARTs' register memory mapping and bit fields.

16.3.1 Memory Map

The base address for UART0 is 0xFFFC0000; the base address for UART1 is 0xFFFC1000; the base address for UART2 is 0xFFFC2000. Table 16-2 shows the memory map for the UART registers

Table 16-2. UART Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	UARTDR	**Data Register**
0x004	UARTRSR/UARTECR	**Receive Status Register** (Read) **Error Clear Register** (Write)
0x008 - 0x014	///	**Reserved** — Do not access
0x018	UARTFR	**Flag Register**
0x01C	///	**Reserved** — Do not access
0x020	UARTILPR	**IrDA Low Power Counter Register**
0x024	UARTIBRD	**Integer Baud Rate Divisor Register**
0x028	UARTFBRD	**Fractional Baud Rate Divisor Register**
0x02C	UARTLCR_H	**Line Control Register, HIGH byte**
0x030	UARTCR	**Control Register**
0x034	UARTIFLS	**Interrupt FIFO Level Select Register**
0x038	UARTIMSC	**Interrupt Mask Set/Clear Register**
0x03C	UARTRIS	**Raw Interrupt Status Register**
0x040	UARTMIS	**Masked Interrupt Status Register**
0x044	UARTICR	**Interrupt Clear Register**
0x048	DMACTRL	**UART0 DMA Control Register**
0x04C - 0x07C	///	**Reserved** — Do not access
0x080 - 0x08C	///	**Reserved** — Do not access
0x090 - 0xFFC	///	**Reserved** — Do not access

16.3.2 Register Definitions

16.3.2.1 Data Register (UARTDR)

UARTDR is the Data Register for words that are to be transmitted or have been received over the serial interface. Writing to this register initiates transmission from the UART.

- If the FIFOs are enabled, data written to this location is pushed to the transmit FIFO.
- If the FIFOs are not enabled, data is stored in the Transmitter Holding Register (THOR), the bottom word of the transmit FIFO.

A read of this register pops the first word from the receive FIFO. This word consists of the received character and the associated error bits.

Table 16-3. UARTDR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///				OE	BE	PEAR	FE	DATA							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x000 UART 1: 0xFFFC1000 + 0x000 UART 2: 0xFFFC2000 + 0x000															

Table 16-4. UARTDR Fields

BIT	NAME	DESCRIPTION
31:12	///	**Reserved** Reading returns 0. Write the reset value.
11	OE	**Receive FIFO Full/Empty** This bit indicates an overrun condition. 1 = Data is received and the receive FIFO is already full 0 = There is an empty space in the FIFO and a new character can be written
10	BE	**Break Error** This bit indicates a Break has occurred. It is cleared to 0 after a Write to the UARTECR Register. In FIFO Mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (Idle state) and the next valid Start bit is received. 1 = A Break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as Start, data, parity and Stop bits) 0 = No Break
9	PEAR	**Parity Error/Address Received** See Table 16-5. In FIFO mode, this error is associated with the character at the top of the FIFO.
8	FE	**Framing Error** This bit indicates that a Framing Error has occurred. In FIFO Mode, this error is associated with the character at the top of the FIFO. 1 = The received character did not have a valid Stop bit (a valid Stop bit is 1) 0 = No Framing Error
7:0	DATA	**Receive/Transmit Data Character** This field contains the receive or transmit data. Read = Receive data character Write = Transmit data character

Table 16-5. Nine-bit Mode/Parity Bit Table

REGISTER:BIT		MEANING
UARTLCR_H: 9BIT	UARTDR: PEAR	
0	0	The parity of the received data character matches the parity selected as defined by the EPS and SPS bits in the UARTLCR_H register. In FIFO mode, this error is associated with the character at the top of the FIFO.
0	1	The parity of the received data character does not match the parity selected as defined by the EPS and SPS bits in the UARTLCR_H register. In FIFO mode, this error is associated with the character at the top of the FIFO.
1	0	The received character is data.
1	1	The received character is an address.

16.3.2.2 Receive Status/Error Clear Register (UARTRSR/UARTECR)

UARTRSR/UARTECR is the Receive Status Register/Error Clear Register. If the status is read from this register, the status bits in this register correspond to the status bits of the last word read from the UARTDR Register (see Section 16.3.2.1). The status information for overrun is set immediately when an overrun condition occurs.

A write to the UARTECR Register clears the Framing, Parity/address Received, Break, and Overrun Errors. All bits clear to 0 on System Reset.

NOTE: The received data character must be read first from UARTDR before reading the error status associated with that data character from UARTRSR. This read sequence cannot be reversed because the Status Register, UARTRSR, is updated only when a read occurs from the Data Register, UARTDR. However, the status information can also be obtained by reading the UARTDR Register.

Table 16-6 and Table 16-7 describe the UARTRSR/UARTECR Register for Write operations.

Table 16-6. UARTRSR/UARTECR Register (Write Operations)

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								EC							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	UART 0: 0xFFFC0000 + 0x004 UART 1: 0xFFFC1000 + 0x004 UART 2: 0xFFFC2000 + 0x004															

Table 16-7. UARTRSR/UARTECR Fields (Write Operations)

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	EC	**Error Clear** A Write to this register clears the Framing, Break, Parity/Address Received, and Overrun Errors. The data value is not important.

Table 16-8 and Table 16-9 describe the UARTRSR/UARTECR Register for Read operations.

Table 16-8. UARTRSR/UARTECR Register (Read Operations)

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												OE	BE	PEAR	FE
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	UART 0: 0xFFFC0000 + 0x004 UART 1: 0xFFFC1000 + 0x004 UART 2: 0xFFFC2000 + 0x004															

Table 16-9. UARTRSR/UARTECR Fields (Read Operations)

BITS	NAME	DESCRIPTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	OE	**Data Overrun Error** This bit signifies when a data overrun has occurred. 1 = Data is received and the FIFO is already full 0 = Cleared to 0 by a write to the UARTECR Register. The FIFO contents remain valid since no further data is written when the FIFO is full, only the contents of the shift register are overwritten. The CPU must now read the data in order to empty the FIFO.
2	BE	**Break Error** In FIFO Mode, this error is associated with the character at the top of the FIFO. When a break occurs, only one 0 character is loaded into the FIFO. The next character is only enabled after the receive data input goes to a 1 (Idle state) and the next valid Start bit is received. 1 = A break condition was detected, indicating that the received data input was held LOW for longer than a full-word transmission time (defined as Start, data, parity, and Stop bits) 0 = This bit is cleared to 0 after a write to the UARTECR Register
1	PEAR	**Parity Error/Address Received** See Table 16-5. This bit is cleared to 0 by a write to UARTECR.
0	FE	**Framing Error** This bit indicates a Framing Error has occurred. 1 = The received character did not have a valid Stop bit (valid Stop bit is 1) 0 = This bit is cleared to 0 by a write to the UARTECR Register

16.3.2.3 Flag Register (UARTFR)

UARTFR is the Flag Register. After System Reset, TXFF, RXFF, and BUSY are 0, and TXFE and RXFE are 1.

Table 16-10. UARTFR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								TXFE	RXFF	TXFF	RXFE	BUSY	///		
RESET	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	UART 0: 0xFFFC0000 + 0x018 UART 1: 0xFFFC1000 + 0x018 UART 2: 0xFFFC2000 + 0x018															

Table 16-11. UARTFR Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	TXFE	**Transmit FIFO Empty** The meaning of this bit depends on the state of the FIFO Enable bit (UARTLCR_H:FEN). See Section 16.3.2.7. For UARTLCR_H:FEN = 1 (FIFO enabled) 1 = Transmit FIFO is empty 0 = Transmit FIFO not empty For UARTLCR_H:FEN = 0 (FIFO disabled) 1 = Transmit Holding Register is empty 0 = Transmit Holding Register not empty
6	RXFF	**Receive FIFO Full** The meaning of this bit depends on the state of the FEN bit in the UARTLCR_H Register (see Section 16.3.2.7). FIFO disabled = This bit is set when the Receive Holding Register is full. FIFO enabled = RXFF bit is set when the receive FIFO is full.
5	TXFF	**Transmit FIFO Full** The meaning of this bit depends on the state of the FEN bit in the UARTLCR_H Register (see Section 16.3.2.7). FIFO disabled = This bit is set when the Transmit Holding Register is full. FIFO enabled = The TXFF bit is set when the transmit FIFO is full.
4	RXFE	**Receive FIFO Empty** The meaning of this bit depends on the state of the FEN bit in the UARTLCR_H Register (see Section 16.3.2.7). FIFO disabled = This bit is set when the Receive Holding Register is empty. FIFO enabled = The RXFE bit is set when the receive FIFO is empty
3	BUSY	**UART Busy** This bit is set as soon as the transmit FIFO becomes non-empty, regardless of whether the UART is enabled or not. 1 = UART is busy transmitting data. This bit remains set until the complete byte, including all Stop bits, has been sent from the shift register. 0 = Not busy
2:0	///	**Reserved** Unpredictable when read; write the reset value.

16.3.2.4 IrDA Low-Power Counter Register (UARTILPR)

Program the UARTILPR Register with a divisor value to generate the SIR Baud Clock signal. The UARTILPR Register is reset to 0 and must be reprogrammed with a non-zero divisor value for use. Programming a zero value will result in no SIR Baud Clock pulses being generated. The SIR Baud Clock is generated by dividing the UART Clock input signal by the Low Power Divisor Value written to this register.

The Low Power Divisor value is calculated as follows:

(frequency UART Clock ÷ frequency SIR Baud Clock) – 1,

where (frequency SIR Baud Clock) is nominally 1.8432 MHz.

The divisor value written to this register must be chosen so that

1.42 MHz < SIR Baud Clock < 2.12 MHz.

The divisor value must produce a low power pulse duration of 1.41 μs - 2.11 μs (three times the period of SIR Baud Clock).

The minimum frequency of IrLPBaud16 ensures that pulses less than one period of UARTCLK are rejected as random noise, but that pulses greater than two periods of UARTCLK are accepted as valid. Table 16-13 describes the bit fields in the UARTILPR Register.

Table 16-12. UARTILPR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								ILPDVSR							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x020 UART 1: 0xFFFC1000 + 0x020 UART 2: 0xFFFC2000 + 0x020															

Table 16-13. UARTILPR Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
7:0	ILPDVSR	**InfraRed Low Power Divisor** See text description for derivation of the value to be programmed.

16.3.2.5 Integer Baud Rate Divisor Register (UARTIBRD)

UARTIBRD is the integer portion of the baud rate divisor value. All of the bits in this register clear to 0 on System Reset.

Table 16-14. UARTIBRD Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	BAUDDIVINT															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x024 UART 1: 0xFFFC1000 + 0x024 UART 2: 0xFFFC2000 + 0x024															

Table 16-15. UARTIBRD Fields

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15:0	BAUDDIVINT	**Integer Baud Rate Divisor** This value is used with the Fractional Baud Rate Divisor to program the baud rate for the UART. For information about calculating this divisor value, see Section 16.3.2.6.1. For information about the Fractional Baud Rate Divisor, see Section 16.3.2.6. These bits are cleared to 0 on Reset.

16.3.2.6 Fractional Baud Rate Divisor Register (UARTFBRD)

UARTFBRD is the fractional portion of the baud rate divisor value.

Table 16-16. UARTFBRD Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///										FRAC					
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x028 UART 1: 0xFFFC1000 + 0x028 UART 2: 0xFFFC2000 + 0x028															

Table 16-17. UARTFBRD Fields

BIT	NAME	DESCRIPTION
31:6	///	**Reserved** Reading returns 0. Write the reset value.
5:0	FRAC	**Fractional Baud Rate Divisor** This value is used with the Integer Baud Rate Divisor to program the baud rate for the UART. See Section 16.3.2.6.1 and Section 16.3.2.5.

16.3.2.6.1 Calculating the Divisor Value

The following example shows how to program a divisor value. This example assumes that the required baud rate is 460,800 and the UARTCLK = 11.2896 MHz.

1. Baud Rate Divisor = $(11.2896 \times 10^6) \div (16 \times 460{,}800) = 1.53125$
2. Integer part, i = integer (1.53125) = 1
3. Fractional part, m = integer $((0.53125 \times 64) + 0.5) = 34$
4. Generated baud rate divider = 1 + 34/64 = 1.53125
5. Generated baud rate = $(11.2896 \times 10^6) \div (16 \times 1.53125) = 460{,}800$

The maximum error using a 6-bit UARTFBRD Register = 1/64 × 100 = 1.56%. This occurs when m = 1 and the error is cumulative over 64 clock ticks. Using an 11.2896 MHz crystal produces the standard baud rates with no error.

16.3.2.6.2 Typical Baud Rates and Their Corresponding Divisor

Table 16-18 shows typical baud rates and their corresponding divisor using an example UART clock frequency of 11.2896 MHz.

Table 16-18. Bit Rates and Their Corresponding Divisors

UART CLK (MHz)	BAUD RATE	INTEGER DIVISOR (UARTIBRD)	FRACTIONAL DIVISOR, m (UARTFBRD)
11.2896	921,600	0	49
11.2896	460,800	1	34
11.2896	230,400	3	4
11.2896	115,200	6	8
11.2896	76,800	9	12

16.3.2.7 Line Control Register (UARTLCR_H)

UARTLCR_H is the Line Control Register. This register is used to configure the UARTs. The contents of the UARTLCR_H Register are not updated until transmission or reception of the current character is complete. Table 16-21 is a truth table for the SPS, EPS, and PEN bits of the UARTLCR_H Register.

Table 16-19. UARTLCR_H Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///						9BIT	ADDTX	SPS	WLEN		FEN	STP2	EPS	PEN	BRK
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x02C UART 1: 0xFFFC1000 + 0x02C UART 2: 0xFFFC2000 + 0x02C															

Table 16-20. UARTLCR_H Fields

BIT	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
9	9BIT	**Nine-bit Mode Enable** Use this bit to enable Nine-bit Mode. 1 = Nine-bit Mode enabled; characters are tagged during transmission as address or data and checked during reception for address or data 0 = Nine-bit Mode disabled
8	ADDTX	**Transmit Address** This bit allows tagging characters in the UARTDR. Not used and ignored if Nine-bit MODE ENABLE = 0. During Nine-bit Mode (9BIT = 1): 1 = The next character written to UARTDR is tagged as an address. This bit is automatically cleared when UARTDR is written. 0 = The next character written to UARTDR is tagged as data.
7	SPS	**Stick Parity Select** Bits [7], [2], and [1] work together to set up the parity. See Table 16-21.
6:5	WLEN	**Word Length** Indicates the number of data bits transmitted or received in a frame. 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits

Table 16-20. UARTLCR_H Fields (Cont'd)

BIT	NAME	DESCRIPTION
4	FEN	**FIFO Enable Buffers** This bit not only enables and disables the FIFO buffers, but also controls the mode. When the FIFO is enabled, it is used to buffer receive and transmit data. When the FIFO is disabled, the UART is in Character Mode, using just one line in the FIFO to store a single character. 1 = Enables transmit and receive FIFO buffers (FIFO Mode). 0 = FIFOs are disabled (Character Mode); that is, the FIFOs become one-byte-deep holding registers.
3	STP2	**Frame Stop Bits** This bit sets the number of Stop bits. 1 = Two Stop bits are transmitted at the end of the frame. The receive logic does not check for two Stop bits being received. 0 = One Stop bit
2	EPS	**Even Parity Select** Bits [7], [2], and [1] work together to set up the parity. See Table 16-21.
1	PEN	**Parity Enable** Bits [7], [2], and [1] work together to set up the parity. See Table 16-21.
0	BRK	**SEND BREAK** This bit commands the UART to enter a Break condition. This bit must be asserted for at least one complete frame transmission time in order to generate a break condition. The transmit FIFO contents remain unaffected during a break condition. For normal use, this bit must be cleared to 0. 1 = A LOW level is continually output on the UARTTXD output, after completing transmission of the current character 0 = No Break

Table 16-21. Truth Table for 9BIT, SPS, EPS, and PEN bits

9BIT	SPS	EPS	PEN	RESULTANT PARITY BIT (TRANSMITTED OR CHECKED)
0	X	X	0	Not transmitted or checked
0	0	0	1	Odd parity
0	0	1	1	Even parity
0	1	0	1	1
0	1	1	1	0
1	X	X	X	For transmission, ADDRESS TRANSMIT is transmitted as the parity bit for the frame. If the received parity bit = 1, then the PARITY-ERROR/ADDRESS RECEIVED error is generated.

16.3.2.8 UART Control Register (UARTCR)

UARTCR is the UART Control Register. To enable transmission, bit [8] and bit [0] must be set. Similarly, to enable reception, bit [9] and bit [0] must be set.

Table 16-22. UARTCR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	CTSEN	RTSEN	///		RTS	///	RXE	TXE	LBE	///				SIRLP	SIREN	UARTEN
RESET	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x030 UART 1: 0xFFFC1000 + 0x030 UART 2: 0xFFFC2000 + 0x030															

Table 16-23. UARTCR Fields

BIT	NAME	DESCRIPTION
31:10	///	**Reserved** Reading returns 0. Write the reset value.
15	CTSEN	**CTS Hardware Flow Control Enable (CTSEN)** This bit enables CTS hardware flow control. Data is only transmitted when the nUARTCTS signal is asserted. (Only used with UART0) 1 = CTS hardware flow control enabled 0 = CTS hardware flow control disabled
14	RTSEN	**RTS Hardware Flow Control Enable (RTSEN)** This bit enables RTS hardware flow control. Data is only requested when there is space in the receive FIFO for it to be received. (Only used with UART0) 1 = RTS hardware flow control is enabled 0 = RTS hardware flow control disabled
12:13	///	**Reserved** Reading returns 0. Write the reset value.
11	RTS	**Request to Send (RTS)** This bit is the complement of the request to send (nUARTRTS) modem status output. (Only used with UART0) 1 = The output is 0 0 = The output is 1
10	///	**Reserved** Do not modify. When writing to this register, perform a read-modified-write operation to this bit.
9	RXE	**Receive Section** Enables the receive section of the UART. 1 = Receive section enabled. When the UART is disabled in the middle of reception, it completes the current character before stopping 0 = Receive section not enabled

Table 16-23. UARTCR Fields

BIT	NAME	DESCRIPTION
8	TXE	**Transmit Section** Enables the transmit section of the UART. 1 = Transmit section enabled. When the UART is disabled in the middle of transmission, it completes the current character before stopping 0 = Transmit section not enabled
7	LBE	**Loop Back Enable** Places the UART into Loopback Mode. 1 = Loopback is enabled 0 = Loopback is disabled
6:3	///	**Reserved** Do not modify. When writing to this register, perform a read-modified-write operation to this field.
2	SIRLP	**IrDA SIR Low Power Mode** This bit selects the IrDA SIR encoding mode. When this bit field is cleared to 0, low-level bits are transmitted as an active HIGH pulse with a width of 3/16th of the bit period. When this bit field is set to 1, low-level bits are transmitted with a shorter pulse width. Enabling this bit lowers power consumption, but may reduce the optical transmission distance. 1 = IrDA Low Power mode enabled 0 = IrDA Low Power mode disabled
1	SIREN	**SIR Enable** This bit can be programmed to enable or disable the SIR function. When the SIR ENDEC is enabled, data is transmitted and received on the UARTIRRX and UARTIRTX pins. This data is converted to the SIR format by the SIR ENDEC. 1 = SIR Enabled 0 = SIR Disabled
0	UARTEN	**UART Enable** Enables the UART. 1 = Enables the UART. When the UART is disabled in the middle of transmission or reception, it completes the current character before stopping. 0 = UART disabled

16.3.2.9 Interrupt FIFO Level Select Register (UARTIFLS)

UARTIFLS is the Interrupt FIFO Level Select Register. The UARTIFLS Register defines the FIFO level at which interrupts are generated to request service for the receive and transmit FIFOs. The interrupts are generated based on a transition through a level rather than being based on the level; that is, the design is such that the interrupts are generated when the fill level progresses through the trigger level. The bits are reset so that the trigger level is when the FIFOs are at the half-way mark.

Table 16-24. UARTIFLS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///										RXIFLSEL			TXIFLSEL		
RESET	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x034 UART 1: 0xFFFC1000 + 0x034 UART 2: 0xFFFC2000 + 0x034															

Table 16-25. UARTIFLS Fields

BIT	NAME	DESCRIPTION
31:6	///	**Reserved** Reading returns 0. Write the reset value.
5:3	RXIFLSEL	**Trigger Points for the Receive Interrupt** 000 = Receive FIFO becomes ≥ 1/8 full 001 = Receive FIFO becomes ≥ 1/4 full 010 = Receive FIFO becomes ≥ 1/2 full 011 = Receive FIFO becomes ≥ 3/4 full 100 = Receive FIFO becomes ≥ 7/8 full 101 = Receive FIFO becomes full 110:111 = Invalid
2:0	TXIFLSEL	**Trigger Points for the Transmit Interrupt** 000 = Transmit FIFO becomes ≤ 1/8 full 001 = Transmit FIFO becomes ≤ 1/4 full 010 = Transmit FIFO becomes ≤ 1/2 full 011 = Transmit FIFO becomes ≤ 3/4 full 100 = Transmit FIFO becomes ≤ 7/8 full 101 = Transmit FIFO becomes empty 110:111 = Invalid

16.3.2.10 Interrupt Mask Set/Clear Register (UARTIMSC)

UARTIMSC is the Interrupt Mask Register. On a read, this register returns the current value of the mask on the relevant interrupt. Writing 0 to the particular bit masks the interrupt. Writing 1 enables the corresponding interrupt. All the bits are cleared to 0 following a System Reset, which masks all interrupts by default.

The combined UARTx UARTINT, presented to the VIC, is the bitwise logical AND of this register and the Raw Interrupt register, UARTRIS.

Table 16-26. UARTIMSC Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///					OEIM	BEIM	PEARIM	FEIM	RTIM	TXIM	RXIM	///		CTS0IM	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RO	RO	RW	RO
ADDR	UART 0: 0xFFFC0000 + 0x038 UART 1: 0xFFFC1000 + 0x038 UART 2: 0xFFFC2000 + 0x038															

Table 16-27. UARTIMSC Fields

BITS	NAME	DESCRIPTION
31:11	///	**Reserved** Reading returns 0. Write the reset value.
10	OEIM	**Overrun Error Interrupt Mask** When read, returns the current mask for the OEIM interrupt. Write values: 1 = Enable the OEI interrupt 0 = Mask the OEI interrupt
9	BEIM	**Break Error Interrupt Mask** When read, returns the current mask for the BEIM interrupt. Write values: 1 = Enable the BEI interrupt 0 = Mask the BEI interrupt
8	PEARIM	**Parity Error/Address Received Interrupt Mask** When read, returns the current mask for the PEARIM interrupt. Write values: 1 = Enable the PEARI interrupt 0 = Mask the PEARI interrupt
7	FEIM	**Framing Error Interrupt Mask** When read, returns the current mask for the FEIM interrupt. Write values: 1 = Enable the FEI interrupt 0 = Mask the FEI interrupt

Table 16-27. UARTIMSC Fields (Cont'd)

BITS	NAME	DESCRIPTION
6	RTIM	**Receive Timeout Error Interrupt Mask** When read, returns the current mask for the RTIM interrupt. Write values: 1 = Enable the RTI interrupt 0 = Mask the RTI interrupt
5	TXIM	**Transmit Interrupt Mask** When read, returns the current mask for the TXIM interrupt. Write values: 1 = Enable the TXI interrupt 0 = Mask the TXI interrupt
4	RXIM	**Receive Interrupt Mask** When read, returns the current mask for the RXIM interrupt. Write values: 1 = Enable the RXI interrupt 0 = Mask the RXI interrupt
3:2	///	**Reserved** Reading returns 0. Write the reset value.
1	CTS0IM	**CTS0 Interrupt Mask (only for UART0)** When Read, returns the current mask for the CTS0 interrupt. Write values: 1 = Enable the CTS0 interrupt 0 = Mask the CTS0 interrupt
0	///	**Reserved** Reading returns 0. Write the reset value.

16.3.2.11 Raw Interrupt Status Register (UARTRIS)

UARTRIS is the Raw Interrupt Status Register. These values are the state of the interrupt prior to applying the mask specified in the UARTIMSC register. On a read, this register returns the current raw status value of the corresponding interrupt. A write has no effect.

Table 16-28. UARTRIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///					OERIS	BERIS	PEARIS	FERIS	RTRIS	TXRIS	RXRIS	///		CTS0IS	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	UART 0: 0xFFFC0000 + 0x03C UART 1: 0xFFFC1000 + 0x03C UART 2: 0xFFFC2000 + 0x03C															

Table 16-29. UARTRIS Fields

BITS	NAME	DESCRIPTION
31:11	///	**Reserved** Reading returns 0. Write the reset value.
10	OERIS	**Overrun Error Interrupt Status** Specifies the raw interrupt state of the UARTOEINTR interrupt. 1 = Interrupt pending 0 = No interrupt
9	BERIS	**Break Interrupt Status** Specifies the raw interrupt state of the UARTBEINTR interrupt. 1 = Interrupt pending 0 = No interrupt
8	PEARIS	**Parity Error/ Address Received Interrupt Status** Specifies the raw interrupt state of the UARTPEARINTR interrupt. 1 = Interrupt pending 0 = No interrupt
7	FERIS	**Framing Error Interrupt Status** Specifies the raw interrupt state of the UARTFEINTR interrupt. 1 = Interrupt pending 0 = No interrupt
6	RTRIS	**Receive Timeout Interrupt Status** Specifies the raw interrupt state of the UARTRTINTR interrupt. 1 = Interrupt pending 0 = No interrupt

Table 16-29. UARTRIS Fields

BITS	NAME	DESCRIPTION
5	TXRIS	**Transmit Interrupt Status** Specifies the raw interrupt state of the UARTTXINTR interrupt. 1 = Interrupt pending 0 = No interrupt
4	RXRIS	**Receive Interrupt Status** Specifies the raw interrupt state of the UARTRXINTR interrupt. 1 = Interrupt pending 0 = No interrupt
3:2	///	**Reserved** Do not modify.
1	CTS0IS	**CTS0 Interrupt Status (only for UART0)** Specifies the raw interrupt state of the UARTCTS0 interrupt. 1 = Interrupt pending 0 = No interrupt
0	///	**Reserved** Reading returns 0. Write the reset value.

16.3.2.12 Masked Interrupt Status Register (UARTMIS)

UARTMIS is the Masked Interrupt Status Register. On a read, this register returns the current masked status value of the corresponding interrupt. A write has no effect.

Table 16-30. UARTMIS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///					OEMIS	BEMIS	PEARMIS	FEMIS	RTMIS	TXMIS	RXMIS	///		CTS0MIS	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	UART 0: 0xFFFC0000 + 0x040 UART 1: 0xFFFC1000 + 0x040 UART 2: 0xFFFC2000 + 0x040															

Table 16-31. UARTMIS Fields

BIT	NAME	DESCRIPTION
31:11	///	**Reserved** Reading returns 0. Write the reset value.
10	OEMIS	**Overrun Error Masked Interrupt Status** Specifies the masked interrupt state of the UARTOEINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
9	BEMIS	**Break Error Masked Interrupt Status** Specifies the masked interrupt state of the UARTBEINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
8	PEARMIS	**Parity Error/Address Received Masked Interrupt Status** Specifies the masked interrupt state of the UARTPEARINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
7	FEMIS	**Framing Error Masked Interrupt Status** Specifies the masked interrupt state of the UARTFEINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
6	RTMIS	**Receive Timeout Masked Interrupt Status** Specifies the masked interrupt state of the UARTRTINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
5	TXMIS	**Transmit Timeout Masked Interrupt Status** Specifies the masked interrupt state of the UARTTXINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked

Table 16-31. UARTMIS Fields (Cont'd)

BIT	NAME	DESCRIPTION
4	RXMIS	**Receive Masked Interrupt Status** Specifies the masked interrupt state of the UARTRXINTR interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
3:2	///	**Reserved** Reading returns 0. Write the reset value.
1	CTS0MIS	**CTS0 Masked Interrupt Status (only for UART0)** Specifies the masked interrupt state of the CTS0 interrupt. 1 = Interrupt pending 0 = No interrupt, or interrupt masked
0	///	**Reserved** Reading returns 0. Write the reset value.

16.3.2.13 Interrupt Clear Register (UARTICR)

UARTICR is the Interrupt Clear Register. The active bits used in this register are Write Only. On a write of 1, the corresponding interrupt is cleared. A write of 0 has no effect.

Table 16-32. UARTICR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///					OEIC	BEIC	PEARIC	FEIC	RTIC	TXIC	RXIC	///		CTS0IC	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	WO	WO	WO	WO	WO	WO	WO	RO	RO	WO	RO
ADDR	UART 0: 0xFFFC0000 + 0x044 UART 1: 0xFFFC1000 + 0x044 UART 2: 0xFFFC2000 + 0x044															

Table 16-33. UARTICR Fields

BIT	NAME	DESCRIPTION
31:15	///	**Reserved** Reading returns 0. Write the reset value.
10	OEIC	**Overrun Error Interrupt Clear** 1 = Clears the interrupt 0 = No effect
9	BEIC	**Break Error Interrupt Clear** 1 = Clears the interrupt 0 = No effect
8	PEARIC	**Parity Error/Address Received Interrupt Clear** 1 = Clears the interrupt 0 = No effect
7	FEIC	**Framing Error Interrupt Clear** 1 = Clears the interrupt 0 = No effect
6	RTIC	**Receive Timeout Interrupt Clear** 1 = Clears the interrupt 0 = No effect
5	TXIC	**Transmit Interrupt Clear** 1 = Clears the interrupt 0 = No effect
4	RXICR	**Receive Interrupt Clear** 1 = Clears the interrupt 0 = No effect
3:2	///	**Reserved** Reading returns 0. Write the reset value.
1	CTS0IC	**CTS0 Interrupt Clear (only for UART0)** 1 = Clears the interrupt 0 = No effect
0	///	**Reserved** Reading returns 0. Write the reset value.

16.3.2.14 UART0 DMA Control Register (DMACTRL)

UART0 DMACTRL is the UART0 DMA Control Register. It allows control of certain UART DMA functions.

The RXDMAEN, and TXDMAEN bits are not automatically cleared for standard Stream 0 through 3 DMA operations, respectively. These bits should be explicitly cleared by software as soon as possible following DMA completion.

On initiating a DMA operation the DMAC:CTRL:ENABLE bit should be set before either of the above mentioned bits are set.

Table 16-34. DMACTRL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													DMAOE	TXDMAEN	RXDMAEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
ADDR	UART 0: 0xFFFC0000 + 0x048															

Table 16-35. DMACTRL Fields

BIT	NAME	DESCRIPTION
15:3	///	**Reserved** Reading returns 0. Write the reset value.
2	DMAOE	**DMA on Error** 1 = Disables the DMA receive request output when the UART Error Interrupt is asserted 0 = Does not disables the DMA receive request output when the UART Error Interrupt is asserted
1	TXDMAEN	**Transmit DMA Enable** 1 = Enables the DMA for the transmit FIFO 0 = Disables the DMA for the transmit FIFO
0	RXDMAEN	**Receive DMA Enable** 1 = Enables the DMA for the receive FIFO 0 = Disables the DMA for the receive FIFO

Chapter 17
Universal Serial Bus Device

The USB Device is compatible with the USB 1.1 and 2.0 Full Speed specification and compatible with both OpenHCI and Intel UHCI standards. This USB Device supports USB-standard Full-Speed (12 Mbit/s) operation, and SUSPEND and RESUME signalling. Four Endpoints (EP[3:0]) are supported which allow Bulk/Interrupt or Isochronous transfer. Each of the Endpoints has a separate in and out FIFO (except EP0 which shares a FIFO between IN and OUT). The FIFOs allow DMA access.

Figure 17-1 shows the USB Device block diagram. The individual blocks in the diagram are described in the subsequent sections.

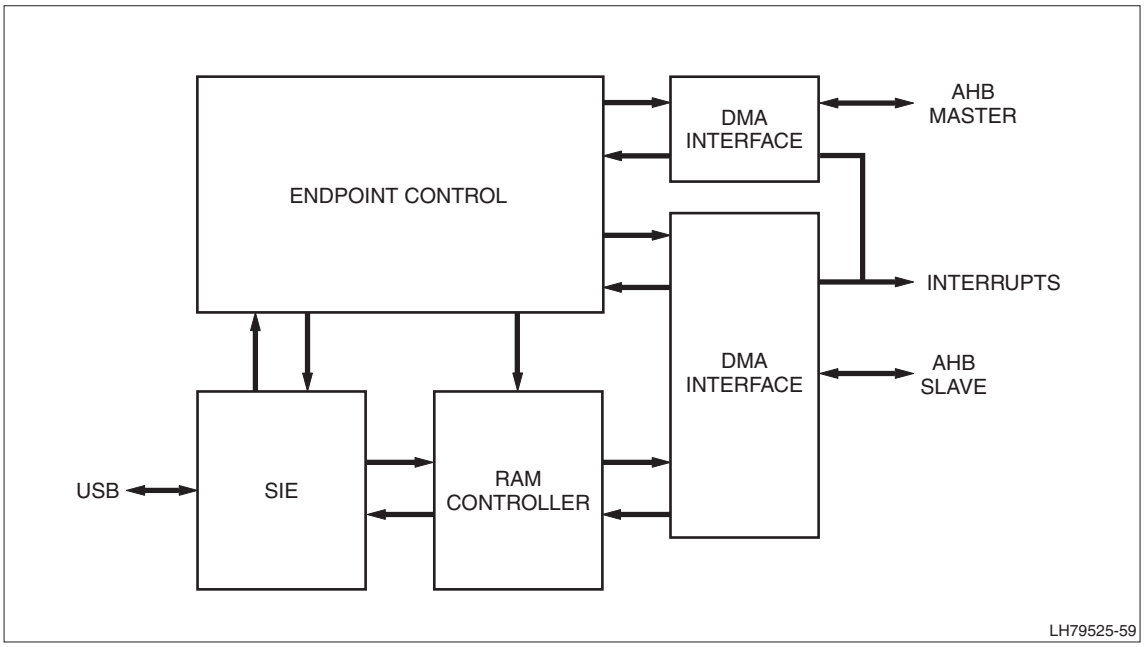


Figure 17-1. USB Block Diagram

17.1 Theory of Operation

The LH79524 and LH79525 implement a USB Device only. All USB communications are managed by one or more external USB Hosts. The USB Device is identical in both parts, so all descriptions apply to the LH79524 and LH79525.

17.1.1 Endpoints

Communications take place between the Host and Device via data 'pipes'. Multiple communication pipes can exist between a Host and the Client, with each pipe terminating at the USB Device in an 'endpoint'.

The USB Device has four endpoints, Endpoint 0 (EP0) through Endpoint 3 (EP3). Each endpoint has a FIFO to facilitate communications. Figure 17-2 is a graphical representation of these endpoints, and Table 17-1 describes the endpoints and their function. Note that the direction type associated with the endpoints is from the perspective of the Host. For example, an IN endpoint terminates a data pipe transferring data *from* the LH79524/LH79525 *to* the Host.

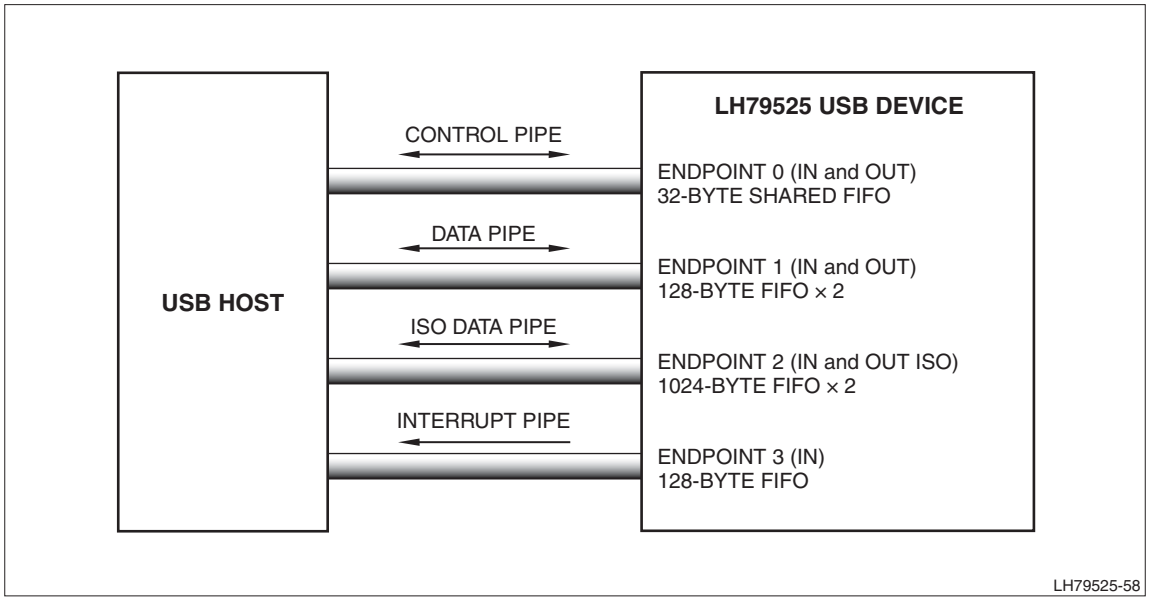


Figure 17-2. USB Communication Endpoints

Table 17-1. Endpoint Function

ENDPOINT	TYPE	FUNCTION
EP0	IN/OUT	Control endpoint. EP0 is always enabled when power is applied. The USB Host uses EP0 for initial configuration and for control.
EP1	IN/OUT	EP1 handles Bulk IN and OUT transfers. Transfers of large data blocks from the USB Device into the USB Host take place through the pipe terminating in EP1.
EP2	IN/OUT	EP2 handles Isochronous IN and OUT transfers. Transfers of large data blocks from the USB Host into the USB Device take place through the pipe terminating in EP2.
EP3	IN	EP3 handles Interrupt IN transfers. The EP3 pipe allows the USB Device to interrupt the USB Host.

17.1.1.1 Isochronous Endpoints

For Isochronous Endpoints, the host generates IN tokens based on the polling interval. The MCU responds to the IN token with data packet. The polling interval is set in the software driver.

Most of the standard drivers generate single IN token per polling period. Some custom drivers may attempt to execute more than one transfer per polling period. This will result in an error. Thus, drivers must be written that do not attempt more than one transfer per polling period.

17.1.2 FIFOs

Each data pipe endpoint has a FIFO to assemble USB serial data into parallel data to be used by the DMA controller. The FIFOs and maximum packet size for each endpoint are described in Table 17-2.

Table 17-2. Endpoint FIFO Characteristics

ENDPOINT	FIFO SIZE	MAXIMUM PACKET SIZE
EP0	32 Bytes	32 Bytes
EP1 In	128 Bytes	64 Bytes
EP1 Out	128 Bytes	64 Bytes
EP2 In	1024 Bytes	1024 Bytes
EP2 Out	1024 Bytes	1024 Bytes
EP3 In	128 Bytes	64 Bytes

17.1.3 Serial Interface Engine (SIE)

The SIE handles the direct USB interface. The Serial Interface functions include:

- Decoding and encoding
- Cyclic redundancy check (CRC) generation and checking
- Bit stuffing
- Endpoint address decoding for USB packets
- Interface signals for an external USB Transceiver.

The Serial Interface incorporates differential USB transceivers implementing a USB-standard Full Speed interface, allowing communication at up to 12 Mbits/s.

17.1.3.1 OUT_PKT_RDY Interrupt Operation for Endpoint 0

The USB Host, via the protocol layer, sends a zero-length packet to acknowledge receipt of data during control transactions. The USB Controller acknowledges this packet, but doesn't generate an OUT_PKT_RDY Interrupt. This reduces communication overhead.

Since the next packet from the USB Controller is a Setup Packet, communication will transpire with no problems. However, if the driver is written to use the OUT_PKT_RDY Interrupt, it will not function. Drivers must be written without the use of the OUT_PKT_RDY Interrupt.

17.1.4 DMA Interface

The USB Device includes a six channel DMA Controller with a 64-byte buffer. This section describes the DMA operation and gives programming examples.

17.1.4.1 DMA Modes

The DMA controller supports two modes of operation. The operating mode of each channel can be programmed independently.

- DMA Mode 0: DMA Mode 0 can be used with any endpoint, whether it uses Control, Bulk, Isochronous, or Interrupt transactions. In this mode, the DMA controller can be programmed to only load/unload one packet, so processor intervention is required for each packet transferred over the USB.
- DMA Mode 1: DMA Mode 1 can only be used with endpoints that use Bulk transactions. In this mode, the DMA controller can be programmed to load/unload a complete Bulk transfer (which can be many packets). Once set up, the DMA controller will load and unload all packets of the transfer, interrupting the processor when the entire transfer has completed.

17.1.4.2 Setting up DMA

Before programming the DMA controller, check the OUT_PKT_RDY bit. If it is set to 1, use DMA Mode 0; otherwise program DMA in Mode 1. To assure proper operation, software should implement a loop as follows:

1. Check the OUT_PKT_RDY bit.
2. If it is 1, program DMA in Mode 0 to transfer the first packet and then exit.
3. If OUT_PKT_RDY bit is 0 in step1, program DMA in mode1.
4. Since software takes several instructions to program DMA in mode1, a packet can arrive in between, causing OUT_PKT_RDY to become set. Therefore, check whether OUT_PKT_RDY became set before returning. If set then check whether the transfer started by comparing DMA ADDRx with the previously programmed values. No change in address implies transfer didn't start. If transfer didn't start go to step 2. If transfer started or OUT_PKT_RDY is not set, return.

17.1.4.3 DMA Bus Cycles

The DMA controller uses incrementing bursts of unspecified length on the AHB. It starts a new burst when first granted bus mastership (whether at the start of a USB packet or when regaining the bus after losing it part way through a packet), and when the AHB address starts a new 1KB block.

When the DMA controller is instructed to load a packet, it requests bus mastership of the AHB. When granted, the DMA controller reads the AHB until either the entire packet has been read or the DMA buffer becomes full. It then releases the AHB and begins transferring the data just read to the selected FIFO. This process is repeated until the entire packet has been read and loaded into the FIFO.

When the DMA controller is instructed to unload a packet, it reads data from the selected FIFO into the DMA buffer until either the entire packet has been read or the DMA buffer becomes full. It then requests bus mastership on the AHB and, when granted, performs AHB writes until the DMA buffer has been unloaded. This process is repeated until the entire packet has been unloaded from the FIFO and transferred on the AHB.

If AHB slave access to any of the registers occurs while the DMA controller is either loading or unloading the FIFO, the DMA controller will stop the transfer while the AHB slave access is being made, then continue the transfer after the slave access has been completed.

As long as the start address (written to the ADDRx register) is word aligned, all the transfers for a packet will be word transfers (32-bits), with possible half-word and/or byte transfers added at the end to handle any residue. However if the start address is merely half-word aligned, the DMA controller will use half-word transfers for the duration of the packet, with a possible byte transfer at the end. If the start address is an odd byte address, the DMA controller will use byte transfers for the duration of the packet.

17.1.4.4 Bus Errors

If a bus error occurs while the DMA controller is accessing memory on the AHB, the DMA controller immediately terminates the DMA transfer and interrupts the processor with the CNTLx:BUS_ERR bit set to 1.

17.1.5 DMA Operation

DMA access to the Endpoint FIFOs requires both the DMA controller and the endpoint to be programmed for the selected DMA Mode. Details are given in the following sections. (It will be helpful to refer to the register descriptions in Section 17.2 before reading these sections.)

17.1.5.1 DMA Mode 0: OUT Endpoints

For operation in DMA Mode 0, these steps describe programming an OUT endpoint:

1. Program the proper interrupt enable bit in the OUT Interrupt Enable (OIE) register to 1 to enable that interrupt.
2. Program the INDEX register to the EP number.
3. Then program the OUTCSR2:USB_DMA_EN bit to 0. This disables DMA bulk transfers for Mode 0 operation.
4. When a packet has been received by the USB Device, it sends an interrupt to the VIC. Software should then program the DMA registers with:
   - ADDRx: Memory address to store packet
   - COUNTx: Size of packet (determined by reading the OUTCOUNTx register)
   - CNTLx: 0x0009 (see Section 17.2.3.14 for the CNTLx register description)

The DMA Controller then requests bus mastership and transfers the packet to memory. When it completes the transfer, an interrupt is asserted to the VIC.

5. Software must then program the OUTCSR1:OUT_PKT_RDY bit to 0, indicating that there is no packet waiting for transfer.

17.1.5.2 DMA Mode 0: IN Endpoints

For operation in DMA Mode 0, these steps describe programming an IN endpoint:

1. Program the proper interrupt enable bit in the IN Interrupt Enable (IIE) register to 1 to enable that interrupt.
2. Program the INDEX register to the EP number.
3. Program INCSR2:USB_DMA_EN bit to 0 to disable DMA bulk transfers for Mode 0.

4. When the FIFO becomes available in the USB Device, an interrupt is sent to the VIC. Software should then program the DMA registers with:
   – ADDRx: Memory address of packet to send
   – COUNTx: Size of packet to be sent
   – CNTLx: 0x000B

The DMA controller then requests bus mastership and transfers the packet to the Endpoint FIFO. When it completes the transfer, an interrupt is asserted to the VIC.

5. Software must program the INCSR1:IN_PKT_RDY bit to 0, indicating that the FIFO is available for packet data.

17.1.5.3 DMA Mode 1: OUT Endpoints

For operation in DMA Mode 0, these steps describe programming an OUT endpoint:

1. Program the proper interrupt enable bit in the OIE register to 1 to enable that interrupt.
2. Program the INDEX register to the EP number.
3. Then program the OUTCSR2:AUTO_CLR, OUTCSR2:USB_DMA_EN, and OUTCSR2:DMA_MODE bits to 1. This enables DMA bulk transfers and Mode 1 operation, plus the automatic OUT_PKT_RDY clearing function.
4. Program the DMA registers with:
   – ADDRx: Memory address to store packet
   – COUNTx: Maximum size of the data buffer
   – CNTLx: Program the DMAEN, DMA_MODE, and INTEN bits to 1, and DIRECTION to 0. Program the packet size corresponding to OUTMAXP into bits [14:8].

When a packet is received by the USB Device, the DMA controller requests bus mastership and transfers the packet to memory. The USB Device automatically clears the OUT_PKT_RDY bit in the appropriate OUTCSR1 register. This process continues automatically until the USB Device receives a 'short packet' (one of less than the maximum packet size for the endpoint), signifying the end of the transfer. This 'short packet' will not be transferred by the DMA controller. Instead the USB Device asserts an interrupt to the VIC. Software can then read the OUTCOUNTx register to see the size of the 'short packet' and either unload it manually or reprogram the DMA controller in Mode 0 to unload the packet.

The DMA controller ADDRx register automatically increments as the packets are unloaded so software can compare the current value of ADDRx with the start address of the memory buffer to determine the size of the transfer. If the size of the transfer exceeds the data buffer size, the DMA controller will stop unloading the FIFO and interrupt the processor.

17.1.5.4 DMA Mode 1: IN Endpoints

For operation in DMA Mode 1, these steps describe programming an IN endpoint:

1. Program the proper interrupt enable bit in the IN Interrupt Enable (IIE) register to 1 to enable that interrupt.
2. Program the INDEX register to the EP number.
3. Then program the INCSR2:AUTO_SET and INCSR2:USB_DMA_EN bit to 1. This enables DMA bulk transfers for Mode 1 operation and the automatic IN_PKT_RDY clearing function.
4. Program the DMA registers with:
   - ADDRx: Memory address of block to send
   - COUNTx: Size of data block to be sent
   - CNTLx: Program the DMAEN, DMA_MODE, INTEN, and DIRECTION bits to 1. Program the packet size corresponding to INMAXP into bits [14:8].

When the FIFO becomes available in the USB Device, the DMA controller requests bus mastership and transfers a packet to the FIFO. The USB Device automatically sets the IN_PKT_RDY bit to 1 in the appropriate INCSR1 register. This process continues until the entire data block has been transferred to the USB Device. The DMA controller then asserts an interrupt to the VIC. If the last packet to be loaded was less than the maximum packet size for the endpoint, the IN_PKT_RDY bit will not have be set. Software must set the IN_PKT_RDY bit to allow the last 'short packet' to be sent. If the last packet to be loaded was of the maximum packet size, software should still set the IN_PKT_RDY bit to send a null packet signifying the end of the transfer.

17.1.6 Remote Wakeup

The USB Device is capable of being remotely awakened by software following an internal or external stimulus. To enable this capability, the PMR:ENABLE_SUSPEND bit must be programmed to 1. When the USB Device enters Suspend, it programs the PMR:SUSPEND_MODE to 1, which signals software of that fact.

Once in the Suspend mode, software must program the PMR:UC_RESUME bit to 1 to exit. (If the clock to the USB Device has been stopped, it will need to be restarted before this write can occur.)

The software should leave this bit set for approximately 10 ms (minimum of 2 ms, a maximum of 15 ms) then reset it to 0. By this time the hub should have taken over driving Resume signaling on the USB. Note that no Resume Interrupt will be generated when software initiates a remote wakeup.

17.2 Register Reference

This section provides the USB Device register memory mapping and bit fields.

17.2.1 Memory Map

The base address for the USB Device is 0xFFFF5000. Table 17-3 summarizes the USB Device registers.

Table 17-3. USB Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	FAR	Function Address Register
0x004	PMR	Power Management Register
0x008	IIR	Interrupt Register for Endpoint 0, 1, 2, and 3.
0x00C	///	Reserved — Do not access
0x010	OIR	Interrupt register for OUT Endpoint 1 and 2.
0x014	///	Reserved — Do not access
0x018	UIR	Interrupt register for common USB interrupts.
0x01C	IIE	Interrupt enable register for IIR.
0x020	///	Reserved — Do not access
0x024	OIE	Interrupt enable register for OIR.
0x028	///	Reserved — Do not access
0x02C	UIE	Interrupt enable register for UIR.
0x030	FRAME1	Frame number bits 0 to 7.
0x034	FRAME2	Frame number bits 8 to 10.
0x038	INDEX	Index register for selecting the endpoint when accessing the status and control registers.
0x03C	///	Reserved — Do not access
0x040	INMAXP	Maximum packet size for IN endpoint. (INDEX register set to select Endpoints 1, 2, or 3 only)
0x044	CSR0	Control Status register for Endpoint 0. (INDEX register set to select Endpoint 0)
	INCSR1	Control Status register 1 for IN endpoint. (INDEX register set to select Endpoints 1, 2, and 3)
0x048	INCSR2	Control Status register 2 for IN endpoint. (INDEX register set to select Endpoints 1, 2, and 3 only)
0x04C	OUTMAXP	Maximum packet size for OUT endpoint. (INDEX register set to select Endpoints 1 and 2 only)
0x050	OUTCSR1	Control Status register 1 for OUT endpoint. (INDEX register set to select Endpoints 1 and 2 only)
0x054	OUTCSR2	Control Status register 2 for OUT endpoint. (INDEX register set to select Endpoints 1 and 2 only)

Table 17-3. USB Register Summary (Cont'd)

ADDRESS OFFSET	NAME	DESCRIPTION
0x058	OUTCOUNT0	Number of received bytes in Endpoint 0 FIFO. (INDEX register set to select Endpoint 0)
	OUTCOUNT1	Number of bytes in OUT endpoint FIFO (lower byte). (INDEX register set to select Endpoints 1 – 3)
0x05C	OUTCOUNT2	Number of bytes in OUT endpoint FIFO (upper byte). (INDEX register set to select Endpoints 1 – 3 only)
0x060 to 0x07F	///	Reserved — Do not access
0x080	EP0FIFO	FIFO for Endpoint 0.
0x090	EP1FIFO	FIFO for Endpoint 1.
0x0A0	EP2FIFO	FIFO for Endpoint 2.
0x0B0	EP3FIFO	FIFO for Endpoint 3.
0x0C0 to 0x1FF	///	Reserved — Do not access
0x200	INTR	Indicates pending DMA interrupts
0x204	CNTL1	DMA Channel 1 Control
0x208	ADDR1	DMA Channel 1 AHB Memory Address (32 bits)
0x20C	COUNT1	DMA Channel 1 Byte Count (32 bits)
0x210	///	Reserved — Do not access
0x214	CNTL2	DMA Channel 2 Control
0x218	ADDR2	DMA Channel 2 AHB Memory Address (32 bits)
0x21C	COUNT2	DMA Channel 2 Byte Count (32 bits)
0x220	///	Reserved — Do not access
0x224	CNTL3	DMA Channel 3 Control
0x228	ADDR3	DMA Channel 3 AHB Memory Address (32 bits)
0x22C	COUNT3	DMA Channel 3 Byte Count (32 bits)
0x230	///	Reserved — Do not access
0x234	CNTL4	DMA Channel 4 Control
0x238	ADDR4	DMA Channel 4 AHB Memory Address (32 bits)
0x23C	COUNT4	DMA Channel 4 Byte Count (32 bits)
0x240	///	Reserved — Do not access
0x244	CNTL5	DMA Channel 5 Control
0x248	ADDR5	DMA Channel 5 AHB Memory Address (32 bits)
0x24C	COUNT5	DMA Channel 5 Byte Count (32 bits)
0x250	///	Reserved — Do not access
0x254	CNTL6	DMA Channel 6 Control
0x258	ADDR6	DMA Channel 6 AHB Memory Address (32 bits)
0x25C	COUNT6	DMA Channel 6 Byte Count (32 bits)

17.2.2 Register Definitions

17.2.2.1 Function Address Register (FAR)

FAR is a register that should be written with the function's 7-bit address (received through a SET_ADDRESS descriptor). It is then used for decoding the function address in subsequent token packets.

Table 17-4. FAR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								ADDR_UPDATE	FUNCTION_ADDR						
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x000															

Table 17-5. FAR Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	ADDR_UPDATE	**Address Update** Software must program this bit to 1 to inform the USB Host that the FUNCTION_ADDR field in this register has been updated. The USB clears this bit. 1 = The FUNCTION_ADDR field has been updated 0 = The FUNCTION_ADDR field has not been updated
6:0	FUNCTION_ADDR	**Function Address** The CPU writes the USB function address to this field.

17.2.2.2 Power Management Register (PMR)

This register is used for SUSPEND, RESUME, and RESET signalling, and for monitoring USB Bus Reset status.

Table 17-6. PMR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								SOF_UPDATE	///			USB_RESET	UC_RESUME	SUSPEND_MODE	ENABLE_SUSPEND
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RO	RO	RO	RO	RW	RO	RW
ADDR	0xFFFF5000 + 0x004															

Table 17-7. PMR Fields

BITS	NAME	FUNCTION
32:8	///	**Reserved** Reading returns 0. Write the reset value.
7	SOF_UPDATE	**Start of Frame Update** This bit allows software to instruct the USB Device to wait for an SOF token from the time IN_PKT_RDY is set before sending the packet. If an IN token is received before an SOF token, a zero length data packet is sent. This bit is only used by endpoints performing Isochronous transfers. 1 = Wait for SOF token before sending the pending packet 0 = Do not wait for the SOF token before sending packets
6:4	///	**Reserved** Reading returns 0. Write the reset value.
3	USB_RESET	**USB RESET** The USB block programs this bit to 1 when RESET signalling is received from the Host. This bit remains 1 as long as RESET persists on the USB bus. 1 = A RESET signal is asserted on the USB bus 0 = No RESET signal is asserted
2	UC_RESUME	**UC RESUME** Software programs a 1 to this bit for not less than 10 ms, nor more than 15 ms to initiate RESUME signalling. 1 = Initiate RESUME signalling (must be 10 ms to 15 ms) 0 = Normal operation
1	SUSPEND_MODE	**SUSPEND Mode** The USB block programs this bit to 1 when the Host sends SUSPEND signalling to the MCU. 1 = USB in SUSPEND mode 0 = USB not in SUSPEND mode
0	ENABLE_SUSPEND	**SUSPEND Enable** The software programs this bit to 1 to enable the SUSPEND mode. If this bit is zero, the device will not enter SUSPEND mode. 1 = Enable SUSPEND mode 0 = Disable SUSPEND mode (Default)

17.2.2.3 Interrupt Register for Endpoint 0, 1, 2, and 3 (IIR)

IIR is a read-only register that indicates which of the interrupts for IN Endpoints 1, 2, and 3 are currently active. It also indicates whether the Endpoint 0 (the Control Endpoint) interrupt is currently active. Upon interrupt, software should read each of the three interrupt registers (IIR, OIR, and UIR), which clears the interrupt bit. The UIR must be the last register read and cleared.

Table 17-8. IIR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												EP3IN	EP2IN	EP1IN	EP0IN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x008															

Table 17-9. IIR Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	EP3IN	**Endpoint3 IN Interrupt** The EP3 interrupt is generated for Interrupt IN transfers. This bit is programmed to 1 by the USB when: IN_PKT_RDY is cleared to 0 by the USB; The FIFO is flushed by the USB; and USB has issued a STALL response IN token, indicated by SENT_STALL = 1. Software clears this interrupt by reading this register. 1 = Interrupt IN transfer pending 0 = Interrupt cleared or the above conditions are not met
2	EP2IN	**Endpoint2 IN Interrupt** The EP2 interrupt is generated for Isochronous Interrupt IN transfers. The USB block programs this bit to 1 when IN_PKT_RDY bit is cleared to 0 by the USB Host, the FIFO is flushed by the USB Host, and the USB Host has issued a STALL response IN token, as indicated by SENT_STALL = 1. Software clears this interrupt by reading this register. 1 = Isochronous Interrupt IN transfer pending 0 = Interrupt cleared or the above conditions are not met
1	EP1IN	**Endpoint1 IN Interrupt** The EP1 interrupt is generated for BULK IN transfers. The USB block programs this bit to 1 when IN_PKT_RDY bit is cleared to 0 by the USB Host, the FIFO is flushed by the USB Host, and the USB Host has issued a STALL response IN token, as indicated by SENT_STALL = 1. Software clears this interrupt by reading this register. 1 = BULK IN transfer pending and the above three conditions are met 0 = Interrupt cleared or the above conditions are not met
0	EP0IN	**Endpoint 0 Interrupt** The EP0 interrupt is generated for Control transfers. The EP0 interrupt is programmed to 1 by the USB block when: OUT_PKT_RDY is set to 1 by the USB Host; IN_PKT_RDY is cleared to 0 by the USB Host; SENT_STALL is set to 1 by the USB Host; SETUP_END is set to 1 by the USB Host; and DATA_END is cleared to 0 by the USB Host (Indicates end of control transfer). Software clears this interrupt by reading this register. 1 = EP0 interrupt set 0 = EP0 interrupt cleared or the above conditions are not met

17.2.2.4 Interrupt Register for OUT Endpoint 1 and 2 (OIR)

The OUT Interrupt register (OIR) acts as an interrupt status register for the OUT endpoint EP1 and EP2. Upon interrupt, software should read each of the three interrupt registers (IIR, OIR, and UIR), which clears the interrupt bit. The UIR must be the last register read and cleared.

Table 17-10. OIR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													EP2OUT	EP1OUT	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x010															

Table 17-11. OIR Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	EP2OUT	**EP2 Out Interrupt** This interrupt is generated for Isochronous OUT transfers. The USB block programs this bit to 1 when: OUT_PKT_RDY and SENT_STALL are set to 1 by the USB Host. Software clears this interrupt by reading this register. 1 = Isochronous OUT transfer is ready 0 = Interrupt cleared or the above conditions are not met
1	EP1OUT	**EP1 Out Interrupt** This interrupt is generated for BULK OUT transfers. The USB block programs this bit to 1 when: OUT_PKT_RDY and SENT_STALL are set to 1 by the USB Host. Software clears this interrupt by reading this register. 1 = BULK OUT transfer is ready 0 = Interrupt cleared or the above conditions are not met
0	///	**Reserved** Reading returns 0. Write the reset value.

17.2.2.5 Interrupt Register for common USB interrupts (UIR)

UIR is a read-only register that indicates which USB interrupts are currently active. All active interrupts will be cleared when this register is read.

Table 17-12. UIR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												SOF	URINT	RESINT	SUSINT
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x018															

Table 17-13. UIR Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	SOF	**SOF Interrupt** The USB block programs this bit to 1 at the start of each frame. 1 = Start of Frame detected 0 = Interrupt cleared
2	URINT	**USB RESET Interrupt** The USB block programs this bit to 1 when it receives RESET signalling from the USB Host. Software clears this interrupt by reading this register. 1 = USB RESET Interrupt set 0 = Interrupt cleared
1	RESINT	**RESUME Interrupt** The USB block programs this bit to 1 when it receives RESUME signalling while in SUSPEND mode from the USB Host. If the RESUME is due to a USB RESET, the CPU is first interrupted with a RESUME interrupt. Once the clocks resume and the SUSPEND condition persists for 3 ms, USB RESET Interrupt will be asserted. Software clears this interrupt by reading this register. 1 = RESUME Interrupt set 0 = Interrupt cleared
0	SUSINT	**SUSPEND Interrupt** The USB block programs this bit to 1 when it receives SUSPEND signaling from the USB Host. This bit is set whenever there is no activity for 3 ms on the bus. Thus, if the CPU does not stop the clock after the first SUSPEND Interrupt, it will continue to be interrupted every 3 ms as long as there is no activity on the USB bus. This interrupt is disabled by default. Software clears this interrupt by reading this register. 1 = SUSPEND Interrupt set 0 = Interrupt cleared

17.2.2.6 IN Interrupt Enable Register (IIE)

IIE provides interrupt enable bits for the interrupts in IIR. Following reset, all interrupts are enabled.

Table 17-14. IIE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												EP3INEN	EP2INEN	EP1INEN	EP0INEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x01C															

Table 17-15. IIE Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3	EP3INEN	**Endpoint 3 IN Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
2	EP2INEN	**Endpoint 2 IN Interrupt Enable** 0 = interrupt disabled 1 = interrupt enabled
1	EP1INEN	**Endpoint 1 IN Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
0	EP0EN	**Endpoint 0 IN Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled

17.2.2.7 OUT Interrupt Enable Register (OIE)

OIE provides interrupt enable bits for the interrupts in OIR. Following reset, all interrupts are enabled.

Table 17-16. OIE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													EP2OUTEN	EP1OUTEN	///
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW
ADDR	0xFFFF5000 + 0x024															

Table 17-17. OIE Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2	EP2OUTEN	**Endpoint 2 OUT Interrupt Enable.** 0 = Interrupt disabled 1 = Interrupt enabled
1	EP1OUTEN	**Endpoint 1 OUT Interrupt Enable.** 0 = Interrupt disabled 1 = Interrupt enabled
0	///	**Reserved** Reading returns 0. Write the reset value.

17.2.2.8 Interrupt Enable Register (UIE)

UIE provides interrupt enable bits for the interrupts in UIR. Following reset, only the USB RESET and the RESUME interrupts are enabled.

Table 17-18. UIE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												SOFINTEN	URINTEN	RESINTEN	SUSINTEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x02C															

Table 17-19. UIE Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
3	SOFINTEN	**SOF Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
2	URINTEN	**USB RESET Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
1	RESINTEN	**RESUME Interrupt Enable** 1 = Interrupt enabled 0 = Interrupt disabled
0	SUSINTEN	**SUSPEND Interrupt Enable** Software programs this bit to 1 to enable an Interrupt when it receives SUSPEND signalling. This bit is set whenever there is no activity for 3 ms on the bus. Thus, if the CPU does not stop the clock after the first SUSPEND interrupt, it will continue to be interrupted every 3 ms as long as there is no activity on the USB bus. Following reset, this interrupt is disabled. 1 = Interrupt enabled 0 = Interrupt disabled

17.2.2.9 Frame Number Registers (FRAMEx)

The FRAMEx registers store the current USB bus frame number. The frame number comprises 11 bits. FRAME1 holds the lower eight bits and FRAME2 holds the upper three bits.

Table 17-20. FRAME1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								FRAME1							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x030															

Table 17-21. FRAME2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///													FRAME2		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x034															

Table 17-22. FRAME1 Fields

BIT	FIELD	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	FRAME1	**Frame 1** Least significant 8 bits of USB Frame Number.

Table 17-23. FRAME2 Fields

BIT	FIELD	DESCRIPTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2:0	FRAME2	**Frame 2** Most significant 3 bits of USB Frame Number

17.2.3 Indexed Registers

The next group of registers in the USB Device are Indexed. Each IN endpoint and each OUT Endpoint have their own set of control/status registers. Only one set of IN control and status registers and one set of OUT control and status registers appear in the memory map at any one time. Before accessing an endpoint's control/status registers, the endpoint number must be written to the INDEX register so that the correct control/status registers appear in the memory map.

For example, to access INCSR1 (offset 0x44) for Endpoint 1, first write 0x1 to the INDEX register, then read or write INCSR1. To access INCSR1 for Endpoint 3, first write 0x3 to the INDEX register, then read or write INCSR1.

17.2.3.1 Index Register (INDEX)

The INDEX register determines which endpoint control/status registers are accessed at address offsets 0x040 to 0x05C.

Table 17-24. INDEX Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///												INDEX			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	R	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x038															

Table 17-25. INDEX Fields

BITS	NAME	FUNCTION
31:4	///	**Reserved** Reading returns 0. Write the reset value.
3:0	INDEX	**End Point Index** This field specifies, by INDEX offset, which endpoint for the particular register will be accessed by the next software read or write. The Index values are: 0b0011 = EP3 0b0010 = EP2 0b0001 = EP1 0b0000 = EP0

17.2.3.2 IN Maximum Packet Size Register (INMAXP)

INMAXP defines the maximum packet size for transactions through the currently-selected IN endpoint in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1,023 (the maximum size for an Isochronous packet transferred in a Full-speed transaction) rather than 1,024. When setting this value, note the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt, and Isochronous transactions in Full-speed operations.

There is an INMAXP register for each IN endpoint, except Endpoint 0. Each register is accessed via the INDEX register. For example, to access EP1 INMAXP register, write 0x1 to the INDEX register and then write the data required to address INMAXP (0x40).

The value written to this register should match the wMaxPacketSize field of the Standard Endpoint Descriptor for the associated endpoint (see Universal Serial Bus Specification Revision 2.0, Chapter 9). A mismatch may cause unexpected results. If a value greater than the IN FIFO (see Table 17-2) size for the endpoint is written to this register, the value will be automatically changed to the IN FIFO size. If the value written to this register is less than or equal to half the IN FIFO size, two IN packets can be buffered. The register is reset to 0. If this register is changed after packets have been sent from the endpoint, the endpoint IN FIFO should be flushed (using INCSR1:FIFO_FLUSH) after writing the new value to the INMAXP register.

Table 17-26. INMAXP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								INMAXP							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x040 (with the INDEX register set to IN endpoint 1, 2, or 3)															

Table 17-27. INMAXP Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	INMAXP	**IN Maximum Packet Size** Maximum Packet Size/transaction

17.2.3.3 Control Status Register for EP 0 (CSR0)

CSR0 provides control and status bits for Endpoint 0. It's important to be aware that the STALL function behaves differently for EP 0 than the other endpoints.

Endpoint 0 must STALL the host only once so that subsequent commands are possible. It is possible that extra STALLs may be sent to the host if the SendStall is not cleared. After multiple STALL events, some hosts may deem the device unknown and stop any communication attempt.

If the software wants to abort the current transfer, because it cannot process the command or has some other internal error, it should set the SendStall bit (Bit 5). The device will then send a STALL packet to the host, set the SentStall bit (Bit 2, CSR0) and generate an Endpoint 0 interrupt. If the SendStall bit is not cleared before exiting the interrupt service routing for the Endpoint 0 interrupt, extra stalls could be sent to the host.

To ensure correct operation, the Endpoint 0 interrupt service routine must clear the SendStall bit (Bit 5) within this register for Endpoint 0 (CSR0) at the appropriate time.

Table 17-28. CSR0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								CLR_SETUP_END	CLR_OUT	SEND_STALL	SETUP_END	DATA_END	SENT_STALL	IN_PKT_RDY	OUT_PKT_RDY
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RO	RW	RW	RW	RO
ADDR	0xFFFF5000 + 0x044 (with the INDEX register set to 0)															

Table 17-29. CSR0 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	CLR_SETUP_END	**Clear SETUP_END Bit** Software programs a 1 to this bit to clear SETUP_END (bit 4). 1 = Clear the SETUP_END bit to 0 0 = No effect
6	CLR_OUT	**Clear OUT_PKT_RDY Bit** Software programs a 1 to this bit to clear OUT_PKT_RDY (bit 1). 1 = Clear the OUTPACKETRDY bit to 0 0 = No effect

Table 17-29. CSR0 Fields (Cont'd)

BITS	NAME	FUNCTION
5	SEND_STALL	**Send STALL Handshake** Software writes a 1 to this bit at the same time it programs a 0 to OUT_PKT_RDY(bit 0) when it decodes an invalid token. The USB Host issues a STALL handshake to the current control transfer. Software must write a 0 to end the STALL condition. 1 = Issue STALL Handshake 0 = End STALL condition
4	SETUP_END	**Setup Ends** This is a Read Only bit. Software programs this bit to 1 when a control transfer ends, before DATA_END (bit 3) is set. Software programs this bit to 0, by writing a 1 to the CLR_SETUP_END (bit 7) bit. When the USB Host programs this bit to 1, an interrupt is generated to the CPU. When such a condition occurs, the USB Host flushes the FIFO, and invalidates CPU access to the FIFO. When CPU access to the FIFO is invalidated, this bit is programmed to 0. 1 = Control transfer ended 0 = No control transfer end
3	DATA_END	**Data End** Software programs this bit to 1: • After loading the last packet of data into the FIFO, at the same time IN_PKT_RDY is set • While it clears OUT_PKT_RDY after unloading the last packet of data. For a zero-length data phase, when it clears OUT_PKT_RDY and sets IN_PKT_RDY. 1 = Last packet loaded to FIFO 0 = Last packet unloaded from FIFO
2	SENT_STALL	**Sent Stall Handshake** The USB device programs this bit to 1 if the USB Host ends a control transaction due to a protocol violation. An interrupt is generated when this bit is set. Software must clear this bit by writing a 0. 1 = Protocol violation 0 = Normal operation
1	IN_PKT_RDY	**IN Packet Ready** Software programs this bit to 1 after writing a packet of data into ENDPOINT 0 FIFO. The USB block programs this bit to 0 when the USB Host signals that the packet has been successfully received at the Host. An interrupt is generated when the USB Host clears this bit, so software can load the next packet. For a zero-length data phase, software programs IN_PKT_RDY and DATA_END (bit 3) to 1 at the same time. 1 = Data packet written to ENDPOINT 0 FIFO 0 = Data packet successfully sent to USB host
0	OUT_PKT_RDY	**OUT Packet Ready** This is a Read Only bit. The USB programs this bit to 1 once valid data from the packet is written to the FIFO. An interrupt is generated when the USB sets this bit. Software programs this bit to 0 writing a 1 to the CLR_OUT bit (bit 6). 1 = Valid data from packet has been written to the OUT FIFO 0 = No pending data from packet

17.2.3.4 Control Status Register 1 for IN EP 1, 2, and 3 (INCSR1)

The INCSR1 register maintains the control and status bits for IN endpoints. Software should only access this register for an IN endpoint after the endpoint has been configured via INCSR2. The INDEX register must be used to write and read INCSR1.

Table 17-30. INCSR1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									CLRTOG	SENT_STALL	SEND_STALL	FIFO_FLUSH	UNDERRUN	FIFO_NE	IN_PKT_RDY
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x044 (with the INDEX register set to IN endpoint 1, 2, or 3)															

Table 17-31. INCSR1 Fields

BITS	NAME	FUNCTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6	CLRTOG	**Clear Data Toggle** The Serial Interface Engine (SIE) toggles the Data PID sequence identifier for USB transactions with multiple data packets. In the case of an error condition that requires the USB transaction to be re-synchronized, this bit must be programmed to 1 by software to reset the data toggle so that the SIE will transmit a DATA0 packet identifier on the succeeding transfer. This bit is automatically cleared to 0 when the USB Device reads it. 1 = Data Toggle bit cleared 0 = No effect on Data Toggle
5	SENT_STALL	**STALL Send Acknowledge** The USB block programs this bit to 1 when a STALL handshake is issued to an IN token by the USB Host, in response to software programming the SEND_STALL bit to 1. When the USB issues a STALL handshake, IN_PKT_RDY is cleared to 0. Clear this bit by writing a 0 to it. 1 = STALL handshake issued by USB to an in IN token in response to software setting the SEND_STALL bit 0 = Normal operation
4	SEND_STALL	**Send STALL Handshake to USB** Software must program this bit to 1 to issue a STALL handshake to the next IN token. The USB reads the bit and issues a STALL handshake. Software must program the bit to 0 to end the STALL condition. 1 = Issue a STALL handshake to the USB Host 0 = End the STALL condition

Table 17-31. INCSR1 Fields (Cont'd)

BITS	NAME	FUNCTION
3	FIFO_FLUSH	**FIFO Flush Request** Software programs this bit to 1 if it intends to flush the IN FIFO. This bit is programmed to 0 by the USB after the FIFO is flushed (IN_PKT_RDY must be read as a 1 before the USB can program this bit to 0). The CPU is interrupted when this happens. If a token is in progress, the USB waits until the transmission is complete before the FIFO is flushed. If two packets are loaded into the FIFO, only the topmost packet (one that was intended to be sent by the Host) is flushed, and the corresponding IN_PKT_RDY bit for that packet is cleared. 1 = FIFO flush requested 0 = FIFO flush completed
2	UNDERRUN	**Underruns** In Isochronous mode, this bit is set (1) when a zero length data packet is sent after receiving an IN token with the IN_PKT_RDY bit not set. In Bulk/Interrupt mode, this bit is set (1) when a NAK is returned in response to an IN token.
1	FIFO_NE	**FIFO Not Empty** This bit indicates there is at least one data packet in the FIFO. 1 = Either 2 packets are in the IN FIFO and MAXP is equal to, or less than half of the IN FIFO size; or 1 packet is in the IN FIFO and MAXP is less than or equal to the IN FIFO size 0 = 1 packet in the IN FIFO
0	IN_PKT_RDY	**IN Packet Ready** After writing a packet of data into the FIFO, software must program this bit to 1. The USB programs this bit to 0 once the packet has been successfully sent to the Host. An interrupt is generated when the USB clears this bit informing the core that the next packet can be loaded. While this bit is 1, software cannot write to the FIFO. If the SEND_STALL bit is programmed by software to a 1, this bit cannot be programmed to 1. 1 = IN FIFO has unsent data 0 = FIFO available for next IN packet

17.2.3.5 Control Status Register 2 for IN EP 1, 2, and 3 (INCSR2)

The INCSR2 register allows software to configure USB access and the function of the IN_PKT_RDY bit. Software should configure endpoints via INCSR2 before reading the INCSR1 register.

Table 17-32. INCSR2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								AUTO_SET	ISO	///	USB_DMA_EN	FRC_DATA_TOG	///		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RO	RO	RO
ADDR	0xFFFF5000 + 0x048 (with the INDEX register set to IN endpoint 1, 2, or 3)															

Table 17-33. INCSR2 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	AUTO_SET	**Auto Set IN_PKT_RDY Bit** Auto set the IN_PKT_RDY bit. 1 = IN_PKT_RDY is automatically set to 1, without intervention from software, each time MAXP data is written. If software writes less than MAXP data, the IN_PKT_RDY bit must be programmed to 1 by software. When two packets are in the IN FIFO, IN_PKT_RDY will also be automatically set after the first packet has been sent, if the second packet is the maximum packet size. 0 = Software must explicitly control the IN_PKT_RDY bit
6	ISO	**Isochronous Mode Enable** Use this bit to enable Isochronous transfers. 1 = Enable the IN endpoint for isochronous transfers 0 = Enable the IN endpoint for bulk or interrupt transfers
5	///	**Reserved** Reading returns 0. Write the reset value.
4	USB_DMA_EN	**USB DMA Enable** This bit enables DMA Bulk transfers for the IN endpoint. 1 = FIFO is accessed via the DMA 0 = FIFO is accessed via direct Reads
3	FRC_DATA_TOG	**Force Data Toggle** This bit can be used to force the endpoint IN data toggle to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This can be used by interrupt IN endpoints that are used to communicate rate feedback for isochronous endpoints. 1 = Force data toggle 0 = Normal operation
2:0	///	**Reserved** Reading returns 0. Write the reset value.

17.2.3.6 OUT Maximum Packet Size Register EP 1 and 2 (OUTMAXP)

OUTMAXP is programmed with the maximum packet size for transactions through the currently-selected OUT endpoint — in units of 8 bytes, except that a value of 128 sets the maximum packet size to 1023 (the maximum size for an isochronous packet) rather than 1024. In setting this value, note the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transactions in Full-speed operations.

The value written to this register should match the wMaxPacketSize field of the Standard Endpoint Descriptor for the associated endpoint (see Universal Serial Bus Specification Revision 2.0, Chapter 9). A mismatch could cause unexpected results.

The total amount of data represented by the value written to this register must not exceed the FIFO size for the OUT endpoint, and should not exceed half the FIFO size if double-buffering is required. If a value greater than the OUT FIFO size (see Table 17-2) for the endpoint is written to this register, the value will be automatically changed to the OUT FIFO size. If the value written to this register is less than or equal to half the OUT FIFO size, two OUT packets can be buffered.

Table 17-34. OUTMAXP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								OUTMAXP							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFF5000 + 0x04C (with the INDEX register set to OUT endpoint 1 or 2)															

Table 17-35. OUTMAXP Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	OUTMAXP	**OUT Maximum Packet Size** Maximum Packet Size/transaction

17.2.3.7 Control Status Register 1 for OUT EP1 and EP2 (OUTSCSR1)

OUTCSR1 provides control and status bits for transfers through the currently-selected OUT endpoint.

Table 17-36. OUTCSR1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								CLRTOG	SENT_STALL	SEND_STALL	FIFO_FLUSH	DATAERROR	OVERRUN	FIFOFull	OUT_PKT_RDY
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RO	RW	RO	RW
ADDR	0xFFFF5000 + 0x050 (with the INDEX register set to 1 or 2)															

Table 17-37. OUTCSR1 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	CL_DATATOG	**Clear Data Toggle Sequence Bit** The Serial Interface Engine (SIE) tracks the Data PID sequence toggle received for USB transactions with multiple data packets. An error condition that requires the USB transaction to be re-synchronized, this bit should be programmed to 1 to reset the data toggle so that the SIE expects a DATA0 packet identifier on the next transfer. Software writes a 1 to this bit to clear the data toggle bit. The USB block programs this bit to 0 when a read is received from the USB Host. 1 = The data toggle sequence bit is reset to DATA0 0 = No effect
6	SENT_STALL	**Stall Handshake Sent** The USB block sets this bit to 1 when an OUT token is ended with a STALL handshake from the USB Host. The USB block issues a stall handshake if the Host sends more than MAXP data for the OUT token. Software clears this bit by writing a 0. 1 = OUT token ended with a STALL handshake 0 = No STALL handshake received
5	SEND_STALL	**Send Stall Handshake** Software programs a 1 to this bit to issue a STALL handshake to the USB Host. 1 = Issue STALL handshake to USB Host 0 = End STALL condition

Table 17-37. OUTCSR1 Fields (Cont'd)

BITS	NAME	FUNCTION
4	FIFO_FLUSH	**Flush OUT FIFO** Software programs this bit to 1 to flush the FIFO. This bit can be programmed to 1 only when OUT_PKT_RDY is 1. The packet due to be unloaded by software will be flushed. 1 = Flush OUT FIFO 0 = Do not flush FIFO
3	DATAERROR	**Data Error** This bit indicates when OUT_PKT_RDY is set if the data packet has a CRC or bit-stuff error. This bit is automatically cleared when OUT_PKT_RDY is cleared. The bit is only valid in isochronous mode. 1 = OUT_PKT_RDY is set and the data packet has a CRC or bit-stuff error 0 = No error
2	OVERRUN	**Data Overrun** This bit indicates that an OUT packet cannot be loaded into the OUT FIFO. The bit is only valid in isochronous mode. 1 = An OUT packet cannot be loaded into the OUT FIFO 0 = No error
1	FIFO_FULL	**FIFO FULL** This bit indicates no more packets can be accepted. 1 = 2 packets are in the IN FIFO, so the FIFO is full 0 = FIFO is not full
0	OUT_PKT_RDY	**OUT Packet Ready** The USB block programs this bit to 1 once the USB Host has loaded a packet of data into the OUT FIFO. After software reads the entire packet from FIFO this bit must be programmed to 0 by software. An interrupt is generated when this bit is set, notifying the core that a packet is ready to read. 1 = Data packet ready in OUT FIFO 0 = No packet is ready in the OUT FIFO

17.2.3.8 Control Status Register 2 for OUT EP1 and EP 2 (OUTCSR2)

OUTCSR2 provides further control bits for transfers through the currently-selected OUT endpoint.

Table 17-38. OUTCSR2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								AUTO_CLR	ISO	USB_DMA_EN	DMA_MODE	///			
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	R	R	R	R
ADDR	0xFFFF5000 + 0x054 (with the INDEX register set to 1 or 2)															

Table 17-39. OUTCSR2 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	AUTO_CLR	**Auto Clear** This bit allows OUT_PKT_RDY to be automatically cleared if it equals OUTMAXP. If the packet is smaller than OUTMAXP, software must manually clear OUT_PKT_RDY. 1 = OUT_PKT_RDY is automatically programmed to 0, without any intervention from software, each time a complete packet is read from OUT FIFO 0 = Software must explicitly clear the OUT_PKT_RDY bit after reading each packet
6	ISO	**Isochronous Enable** Use this bit to enable the OUT endpoint for isochronous transfers or to enable the OUT endpoint for bulk or interrupt transfers. 1 = Enable the OUT endpoint for isochronous transfers 0 = Enable the OUT endpoint for bulk or interrupt transfers
5	USB_DMA_EN	**USB DMA Enable** Use this bit to enable DMA bulk transfers for the OUT endpoint. 1 = The OUT FIFO is accessed via the DMA 0 = The OUT FIFO is accessed via direct Reads
4	DMA_MODE	**DMA Operation Mode** There are two modes of DMA operation. 1 = A DMA request (but no interrupt) is generated for OUT packets of size OUTMAXP bytes, and an interrupt is generated (but no DMA request) for all other size packets 0 = A DMA request and interrupt is generated for all OUT packets
3:0	///	**Reserved** Reading returns 0. Write the reset value.

17.2.3.9 Count 0 Register (OUTCOUNT0)

OUTCOUNT0 is a read-only register that indicates the number of received data bytes in the Endpoint 0 FIFO. The value returned is valid while CSR0:OUT_PKT_RDY is 1.

Table 17-40. OUTCOUNT0 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///									OUTCOUNT0						
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x058 (with the INDEX register set to 0)															

Table 17-41. OUTCOUNT0 Fields

BITS	NAME	FUNCTION
31:7	///	**Reserved** Reading returns 0. Write the reset value.
6:0	OUTCOUNT0	**Count of OUT EP0 Packet Bytes** Number of bytes in the packet ready to be unloaded by software.

17.2.3.10 Count 1 Register (OUTCOUNT1)

The OUTCOUNT1 is a read-only register that holds the lower 8 bits of the number of received data bytes in the packet in the FIFO associated with the currently-selected OUT endpoint. The value returned is valid while OUTCSR1:OUT_PKT_RDY is 1.

Table 17-42. OUTCOUNT1 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								OUTCOUNT1							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x058 (with the INDEX register set to 1 or 2)															

Table 17-43. OUTCOUNT1 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	OUTCOUNT1	**Count of OUT EP1 or EP2 Packet Bytes** This contains the lower 8 bits of the number of bytes in the packet ready to be unloaded by software.

17.2.3.11 Out Count 2 Register (OUTCOUNT2)

OUTCOUNT2 is a read-only register that holds the upper 3 bits of the number of received data bytes in the packet in the FIFO associated with the currently-selected OUT endpoint. The value returned is valid while OUT_PKT_RDY (OUTCSR1.D0) is set.

Table 17-44. OUTCOUNT2 Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								///					OUTCOUNT2		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x05C (with the INDEX register set to 1 or 2)															

Table 17-45. OUTCOUNT2 Fields

BITS	NAME	FUNCTION
31:3	///	**Reserved** Reading returns 0. Write the reset value.
2:0	OUTCOUNT2	**Count of OUT EP1 or EP2 Packet Bytes** This contains the upper 3 bits of the number of bytes in the packet ready to be unloaded by software.

17.2.3.12 FIFOs for Endpoints 0-3 (FIFOx)

These 4 addresses provide CPU access to the FIFOs for each endpoint. Writing to these addresses loads data into the IN FIFO for the corresponding endpoint. Reading from these addresses unloads data from the OUT FIFO for the corresponding endpoint.

Table 17-46. FIFO Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								FIFO							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	FIFO0 = 0xFFFF5000 + 0x080 (Endpoint 0) FIFO1 = 0xFFFF5000 + 0x084 (Endpoint 1) FIFO2 = 0xFFFF5000 + 0x088 (Endpoint 2) FIFO3 = 0xFFFF5000 + 0x08C (Endpoint 3)															

Table 17-47. COUNT1 Fields

BITS	NAME	FUNCTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7:0	FIFO	**FIFO Accesses** FIFO access for each endpoint

17.2.3.13 Pending DMA Interrupts Register (INTR)

This register indicates the status of pending DMA interrupts.

Table 17-48. INTR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///										INTR6	INTR5	INTR4	INTR3	INTR2	INTR1
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFF5000 + 0x200															

Table 17-49. INTR Fields

BITS	NAME	FUNCTION
31:6	///	**Reserved** Reading returns 0. Write the reset value.
5	INTR6	**Channel 6 Interrupt Status** 1 = Interrupt pending from DMA Channel 6 0 = No pending interrupt
4	INTR5	**Channel 5 Interrupt Status** 1 = Interrupt pending from DMA Channel 5 0 = No pending interrupt
3	INTR4	**Channel 4 Interrupt Status** 1 = Interrupt pending from DMA Channel 4 0 = No pending interrupt
2	INTR3	**Channel 3 Interrupt Status** 1 = Interrupt pending from DMA Channel 3 0 = No pending interrupt
1	INTR2	**Channel 2 Interrupt Status** 1 = Interrupt pending from DMA Channel 2 0 = No pending interrupt
0	INTR1	**Channel 1 Interrupt Status** 1 = Interrupt pending from DMA Channel 1 0 = No pending interrupt

17.2.3.14 DMA Channel x Control Register (CNTLx)

This register allows configuring various functions for DMA Channels 1 through 6.

Table 17-50. CNTLx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	BUS_ERR	MAX							///		ENDPOINT		INTEN	MODE	DIRECTION	EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO	RW	RW	RW	RW	RW	RW
ADDR	Channel 1 = 0xFFFF5000 + 0x204 Channel 2 = 0xFFFF5000 + 0x214 Channel 3 = 0xFFFF5000 + 0x224 Channel 4 = 0xFFFF5000 + 0x234 Channel 5 = 0xFFFF5000 + 0x244 Channel 6 = 0xFFFF5000 + 0x254															

Table 17-51. CNTLx Fields

BITS	NAME	FUNCTION
31:16	///	**Reserved** Reading returns 0. Write the reset value.
15	BUS_ERR	**Bus Error** If a bus error occurs while DMA is accessing memory on the AHB, the DMA controller immediately terminates the DMA transfer and interrupts the processor by setting this bit. 1 = Bus error occurred 0 = No bus error
14:8	MAX	**Max Packet Size** Program with the maximum packet size, in units of 8 bytes (required for Mode 1 only).
7:6	///	**Reserved** Reading returns 0. Write the reset value.
5:4	ENDPOINT	**Endpoint Number** Program the endpoint number into this field (0-3).
3	INTEN	**Interrupt Enable** 1 = Enable interrupt 0 = Disable interrupt
2	DMA_MODE	**DMA Operation Mode** See full description in Table 17-39. 1 = DMA Mode 1 0 = DMA Mode 0
1	DIRECTION	**Data Direction** 1 = IN endpoint 0 = OUT endpoint
0	DMAEN	**DMA Enable** 1 = Enable DMA 0 = Disable DMA

17.2.3.15 DMA Channel x AHB Memory Address Register

Program this register with the AHB memory addresses for each of the six DMA channels.

Table 17-52. ADDRx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	ADDR1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	ADDR1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Channel 1 = 0xFFFF5000 + 0x208 Channel 2 = 0xFFFF5000 + 0x218 Channel 3 = 0xFFFF5000 + 0x228 Channel 4 = 0xFFFF5000 + 0x238 Channel 5 = 0xFFFF5000 + 0x248 Channel 6 = 0xFFFF5000 + 0x258															

Table 17-53. ADDRx Fields

BITS	NAME	FUNCTION
31:0	ADDRx	**Memory Address for Channel x** DMA Channel x AHB Memory Address

17.2.3.16 DMA Channel x Byte Count Register (COUNTx)

Program this register with the byte count for each DMA channel. See Section 17.1.4 for details.

Table 17-54. COUNTx Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	COUNT1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	COUNT1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	Channel 1 = 0xFFFF5000 + 0x20C Channel 2 = 0xFFFF5000 + 0x21C Channel 3 = 0xFFFF5000 + 0x22C Channel 4 = 0xFFFF5000 + 0x23C Channel 5 = 0xFFFF5000 + 0x24C Channel 6 = 0xFFFF5000 + 0x25C															

Table 17-55. COUNTx Fields

BITS	NAME	FUNCTION
31:0	COUNTx	**Byte Count** DMA Channel x Byte Count

Chapter 18
Vectored Interrupt Controller

The LH79524 and LH97525 incorporate a Vectored Interrupt Controller (VIC). A vectored interrupt has improved latency as it provides direct information about where service routines are located and eliminates levels of software arbitration needed with a simpler interrupt controller. Throughout this chapter all descriptions apply to both the LH79524 and LH97525.

18.1 Theory of Operation

The VIC provides hardware for initial prioritization and processing of up to 32 interrupts. Of these 32 interrupts, 23 are routed from internal sources (such as the DMA controller, the Watchdog Timer, etc.); 8 are from external interrupts; and 1 is a ‘spare’ and can be used as a software interrupt-. Up to 16 interrupts can be assigned as vectored interrupts. The VIC is programmed by application software, as are other functional blocks, via a set of registers. All 32 interrupt source lines can be enabled, disabled, and cleared individually, and individual interrupt status may be determined.

All internal and external interrupts are routed to the VIC, where interrupt priority is determined by hardware. The CPU services the interrupt as either a vectored interrupt or a default-vectored interrupt. A vectored interrupt results in a low-latency invocation of the service routine for that particular interrupt. A default-vectored interrupt requires the CPU to perform additional processing to determine which interrupt source caused the interrupt.

Any of the 32 lines can be assigned to any of the 16 interrupt vectors. Any line not explicitly assigned to an interrupt vector is processed as a default-vectored interrupt. At reset, all 32 lines are set to be default-vectored interrupts.

Each interrupt line can be explicitly assigned as either an IRQ interrupt type (default) or an FIQ interrupt type. Vectored-interrupt servicing is only available for IRQ interrupts. Although more than one interrupt source can be designated as FIQ, only one source normally is designated to take advantage of the low latency of FIQ exception handling for a specific need.

When an interrupt is being processed, other incoming, lower priority interrupt requests are masked. Upon completion of in ISR, the mask is cleared, and the next lower priority interrupt can be serviced.

On reset, the VIC is configured to pass all interrupts through to the CPU IRQ input as default-vectored IRQ interrupts. In the reset configuration, the VIC Status Registers can be used in a conventional way to service interrupts using the CPU IRQ Exception Vector at address 0x18. Users must configure the VIC to use the vectored interrupt feature. Following reset, all interrupts are disabled.

18.1.1 VIC Interrupt Listing

Table 18-1 lists the 32 interrupt source lines for the VIC and their permanent position assignment. For a detailed description of each interrupt, see the chapter for the peripheral that generates the interrupt.

Table 18-1. Interrupt Assignments

INTERRUPT NUMBER	DESCRIPTION	INTERRUPT SOURCE	AVAILABLE MODE*			
			STANDBY	SLEEP	STOP1	STOP1
0	WDT	Watchdog Timer				
1	Not Used	Available as a software interrupt				
2	COMMRX	Interrupt for debugging				
3	COMMTX	Interrupt for debugging				
4	Timer0 Combined	Timer0	•			
5	Timer1 Combined	Timer1	•			
6	Timer2 Combined	Timer2	•			
7	External Interrupt 0	External General Purpose interrupt	•	•	•	•
8	External Interrupt 1	External General Purpose interrupt	•	•	•	•
9	External Interrupt 2	External General Purpose interrupt	•	•	•	•
10	External Interrupt 3	External General Purpose interrupt	•	•	•	•
11	External Interrupt 4	External General Purpose interrupt	•	•	•	•
12	External Interrupt 5	External General Purpose interrupt	•	•	•	•
13	External Interrupt 6	External General Purpose interrupt	•	•	•	•
14	External Interrupt 7	External General Purpose interrupt	•	•	•	•
15	RTC Interrupt	Real Time Clock	•	•	•	•
16	TSCIRQ (combined)	Analog-to-Digital Converter	•			
17	BROWNOUTINTR	Brown Out Detector	•			
18	PENIRQ	Analog-to-Digital Converter Pen interrupt	•			
19	CLCD Interrupt	CLCD Controller	•			
20	DMA Stream 0	DMA Controller	•			
21	DMA Stream 1	DMA Controller	•			
22	DMA Stream 2	DMA Controller	•			
23	DMA Stream 3	DMA Controller	•			
24	SSPI2SINTR	Synchronous Serial Port	•			
25	Ethernet Interrupt	Ethernet MAC	•			
26	USB Interrupt	USB Device	•			
27	UART0 UARTINTR	UART0	•			
28	UART1 UARTINTR	UART1	•			
29	UART2 UARTINTR	UART2	•			
30	USB DMA Interrupt	USB DMA Interrupt	•			
31	I^2C Interrupt	I^2C Controller	•			

NOTE: *All interrupts are available in RUN Mode. If an interrupt not marked with a bullet wakes the MCU unintentionally, the offending interrupt should be disabled in the VIC.

18.1.2 Vectored Interrupts

Each interrupt source line must be identified as either an IRQ type or an FIQ type using the Interrupt Select Register (INTSELECT). FIQ interrupts are non-vectored. Once the VIC causes the FIQ interrupt to be asserted to the core, the FIQ interrupt handler is entered directly by loading the instruction at 0x1C independently of the VIC.

For default-vectored interrupts, set the Default Vector Address Register (DEFVECTADDR) to the entry address of the ISR which is to handle all default-vectored interrupts.

Vectored interrupts are set up by:

- Program the Vector Address Register (VECTADDRx where 'x' is 0-15) with the entry address of the ISR which is to handle each vectored interrupt. The easiest scheme is to program VECTADDRx vectors 0-15 in increasing address order.
- Program the VECTCTRLx:INTSOURCE field to the interrupt source for that specific vector. Then, enable that interrupt source as a vectored interrupt using the VECTCTRLx:E field in that register.

For example, to assign the Real Time Clock Alarm interrupt (interrupt 15) to address 0x12345678 using vector 10, program the VECTADDR10 register to address 0x12345678 (the location of the ISR for the RTC Alarm), then program VECTCTRL10 to 0x0000000F (RTC Alarm = interrupt number 15). Then, program the VECTCTRL10:E bit to 1 to enable the vector.

After all interrupt vector addresses and associations have been programmed, enable the interrupts to be active, whether vectored or default-vectored, using the Interrupt Enable Register (INTENABLE).

18.1.3 External Interrupts

All external interrupts are conditioned by the RCPC module before being presented to the VIC. External interrupt conditioning can be configured to one of four triggers using the RCPC Interrupt Configuration Register (see the RCPC Chapter):

- LOW-level trigger
- HIGH-level trigger
- Falling-edge trigger
- Rising-edge trigger.

On reset, all external interrupt triggers are LOW-level triggers. Therefore, make sure that all external interrupt input signals are HIGH at reset. External edge-triggered interrupts must be cleared using the RCPC Interrupt Clear Register. If the external interrupt is configured as a level-trigger interrupt, the external interrupt must be cleared, reset, or disabled at its source (external to the MCU).

18.1.4 Clearing Interrupts

The general procedure for clearing an interrupts is:

1. The interrupt must be cleared at its source, regardless of whether the interrupt source is external, internal, or software generated.
2. The interrupt must be cleared within the VIC by writing any value to the VECTADDRx register. Writing a value of 0 is recommended. This action signals the hardware vector address and priority logic that it can assert a new interrupt and its associated address.

18.1.5 Priority

The VIC can assert an FIQ interrupt and an IRQ interrupt simultaneously. When this occurs, the CPU gives the FIQ priority over the IRQ interrupt. Priority arbitration for simultaneously invoked IRQ interrupts is performed in the VIC hardware.

The priority of IRQ interrupt sources is:

- Vectored interrupts have priority over default-vectored interrupts
- Vectored interrupt priority is from lowest-number to highest-number
- Within the VIC, all default-vectored interrupts have the same priority, which is the lowest priority.

18.1.6 External Level-Sensitive Interrupts

When external interrupts are configured as level-sensitive, the ISR must ensure that there is sufficient time between the external interrupts being cleared and the interrupt at the VIC being cleared. Otherwise, the source of an external interrupt can still be asserted, causing the VIC to enter the ISR a second time. Because the VIC samples the line after the clear, it generates another interrupt to the ARM core if the line is recognized to be still active. To avoid this situation, clear the source of the interrupt as early as practical in the ISR. Doing so ensures a maximum delay between clearing the external interrupt and clearing the interrupt at the VIC.

An interrupt line shared by multiple open-collector devices in a wired-OR configuration with a pull-up resistor can cause multiple interrupts if there is insufficient delay between the time that the source of the external interrupt is cleared and the time that the interrupt at the VIC is cleared. This situation is due to the relatively slow rise time of the interrupt signal when being pulled to its inactive state by the pull-up resistor. The larger the resistor and load capacitance on the interrupt line, the slower the rise time and the greater the delay required.

18.1.7 Software Guidelines

User software that makes changes to the VIC IRQSTATUS, FIQSTATUS, or RAWINTR registers should not immediately issue a read to these registers. Instead, at least one Idle cycle must separate the write and read operations. The Idle cycle(s) is necessary because the VIC is a zero-wait-state peripheral that requires two clocks for the write operation to update internal registers. The pipelining of the AHB, along with the VIC not inserting wait states, means that a read access immediately following a write returns the previous register values.

18.2 Register Reference

This section provides the VIC register memory mapping and bit fields.

18.2.1 Memory Map

Table shows the mapping of the VIC registers. The base address for the VIC is 0xFFFFF000.

Table 18-2. VIC Register Summary

ADDRESS OFFSET	NAME	DESCRIPTION
0x000	IRQSTATUS	IRQ Status Register
0x004	FIQSTATUS	FIQ Status Register
0x008	RAWINTR	Raw Interrupt Status Register
0x00C	INTSELECT	Interrupt Select Register
0x010	INTENABLE	Interrupt Enable Register
0x014	INTENCLEAR	Interrupt Enable Clear Register
0x018	SOFTINT	Software Interrupt Register
0x01C	SOFTINT_CLEAR	Software Interrupt Clear Register
0x020	///	Reserved — Do not access
0x030	VECTADDR	Vector Address Register
0x034	DEFVECTADDR	Default Vector Address Register
0x100	VECTADDR0	Vector Address 0 Register
0x104	VECTADDR1	Vector Address 1 Register
0x108	VECTADDR2	Vector Address 2 Register
0x10C	VECTADDR3	Vector Address 3 Register
0x110	VECTADDR4	Vector Address 4 Register
0x114	VECTADDR5	Vector Address 5 Register
0x118	VECTADD6	Vector Address 6 Register
0x11C	VECTADDR7	Vector Address 7 Register
0x120	VECTADDR8	Vector Address 8 Register
0x124	VECTADDR9	Vector Address 9 Register
0x128	VECTADDR10	Vector Address 10 Register
0x12C	VECTADDR11	Vector Address 11 Register
0x130	VECTADDR12	Vector Address 12 Register
0x134	VECTADDR13	Vector Address 13 Register
0x138	VECTADDR14	Vector Address 14 Register
0x13C	VECTADDR15	Vector Address 15 Register
0x200	VECTCTRL0	Vector Control 0 Register
0x204	VECTCTRL1	Vector Control 1 Register
0x208	VECTCTRL2	Vector Control 2 Register
0x20C	VECTCTRL3	Vector Control 3 Register
0x210	VECTCTRL4	Vector Control 4 Register
0x214	VECTCTRL5	Vector Control 5 Register
0x218	VECTCTRL6	Vector Control 6 Register
0x21C	VECTCTRL7	Vector Control 7 Register

Table 18-2. VIC Register Summary (Cont'd)

ADDRESS OFFSET	NAME	DESCRIPTION
0x220	VECTCTRL8	Vector Control 8 Register
0x224	VECTCTRL9	Vector Control 9 Register
0x228	VECTCTR10	Vector Control 10 Register
0x22C	VECTCTRL11	Vector Control 11 Register
0x230	VECTCTRL12	Vector Control 12 Register
0x234	VECTCTRL13	Vector Control 13 Register
0x238	VECTCTRL14	Vector Control 14 Register
0x23C	VECTCTRL15	Vector Control 15 Register
0x240 - 0x308	///	Reserved — Do not access
0x30C	ITOP	Interrupt Test Output Register
0x310	///	Reserved — Do not access

18.2.2 Register Descriptions

This section describes the bit fields, reset values, and uses of the registers. For simplicity, all of the following register tables indicate the default base addresses.

18.2.2.1 IRQ Status Register (IRQSTATUS)

This Read Only register provides the status of all interrupts [31:0] after IRQ masking. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-3. IRQSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	IRQStatus															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	IRQStatus															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFFF000 + 0x000															

Table 18-4. IRQSTATUS Fields

BITS	NAME	DESCRIPTION
31:0	IRQStatus	**Interrupt Status After Masking** Shows the status of the interrupts after masking by the INTENABLE and INTSELECT Registers. For each bit: 1 = Interrupt is active and generates an IRQ exception to the ARM7 core 0 = Interrupt is not active

18.2.2.2 FIQ Status Register (FIQSTATUS)

This Read Only register provides the status of the interrupts after FIQ masking. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-5. FIQSTATUS Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	FIQStatus															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	FIQStatus															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFFF000 + 0x004															

Table 18-6. FIQSTATUS Fields

BITS	NAME	DESCRIPTION
31:0	FIQStatus	**Interrupt Status After Masking** Shows the status of the interrupts after masking by the IntEnable and IntSelect Registers. For each bit: 1 = Interrupt is active and generates an FIQ exception to the ARM7 core 0 = Interrupt is not active

18.2.2.3 Raw Interrupt Status Register (RAWINTR)

This Read Only register provides the status of the source interrupts (and software interrupts) to the VIC. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-7. RAWINTR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	RawInterrupt															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RawInterrupt															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFFF000 + 0x008															

Table 18-8. RAWINTR Fields

BITS	NAME	DESCRIPTION
31:0	RawInterrupt	**Raw Interrupt Status** Shows the status of the interrupts before masking by the Interrupt Enable Registers. For each bit: 1 = Appropriate interrupt request is active before masking 0 = Appropriate interrupt request is not active before masking

18.2.2.4 Interrupt Select Register (INTSELECT)

This register selects whether the corresponding interrupt source generates an FIQ or an IRQ interrupt. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-9. INTSELECT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	IntSelect															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	IntSelect															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFFF000 + 0x00C															

Table 18-10. INTSELECT Fields

BITS	NAME	DESCRIPTION
31:0	IntSelect	**Interrupt Type** Selects the type of interrupt for the interrupt request. For each bit: 1 = FIQ interrupt 0 = IRQ interrupt

18.2.2.5 Interrupt Enable Register (INTENABLE)

The bits in this register allow software to individually enable and disable interrupts. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-11. INTENABLE Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	IntEnable															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	IntEnable															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFFF000 + 0x010															

Table 18-12. INTENABLE Fields

BITS	NAME	DESCRIPTION
31:0	IntEnable	**Interrupt Enable** Following a System Reset, all interrupts are disabled. Read, for each bit: 1 = Interrupt is enabled, allowing interrupt request to the core CPU 0 = Interrupt is disabled Write, for each bit: 1 = Enable the corresponding interrupt 0 = Has no effect

18.2.2.6 Interrupt Enable Clear Register (INTENCLEAR)

This register clears the individual bits in the INTENABLE Register. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-13. INTENCLEAR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	IntEnable Clear															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	IntEnable Clear															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFFF000 + 0x014															

Table 18-14. INTENCLEAR Fields

BITS	NAME	DESCRIPTION
31:0	IntEnable Clear	**Clear IntEnable Bit** Clears bits in the INTENABLE Register. For each bit: 1 = Clears the corresponding bit in the IntEnable Register 0 = Has no effect

18.2.2.7 Software Interrupt Register (SOFTINT)

SoftInt is the Software Interrupt Register. This register generates software interrupts. Bits [31:0] correspond to the interrupt number in Table 18-1. Note that interrupt number 1 is the only interrupt souce not associated with a physical hardware interrrupt and is therefore reserved for software interrupts. However, software interrupts can be generated using any of the 32 bit locations in the SOFTINT register. For example, as software interrupt can be generated for interrupt number 0 to test the interrupt processing of the hardware WDT interrupt.

Table 18-15. SOFTINT Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SoftInt															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SoftInt															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFFF000 + 0x018															

Table 18-16. SOFTINT Fields

BITS	NAME	DESCRIPTION
31:0	SoftInt	**Generate Software Interrupt** Setting a bit generates a software interrupt for the specific source interrupt before interrupt masking. For each bit: 1 = Software Interrupt asserted 0 = Has no effect

18.2.2.8 Software Interrupt Clear Register (SOFTINTCLEAR)

This Write Only register clears the corresponding bit (and the interrupt assertion) in the SOFTINT Register. Bits [31:0] correspond to the interrupt number in Table 18-1.

Table 18-17. SOFTINTCLEAR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	SoftInt Clear															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	SoftInt Clear															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFFF000 + 0x01C															

Table 18-18. SOFTINTCLEAR FIelds

BITS	NAME	DESCRIPTION
31:0	SoftInt Clear	**Clear SoftInt Register Bits** Clears bits in the SOFTINT Register. For each bit: 1 = Clears the corresponding bit in the SoftInt Register 0 = Has no effect

18.2.2.9 Vector Address Register (VECTADDR)

The Vector Address Register contains the ISR address of the currently active interrupt. Reading this register provides the address of the ISR, and indicates to the priority hardware that the interrupt is being serviced. Writing to this register indicates to the priority hardware that the interrupt has been serviced.

The ISR can read the VECTADDR Register:

- When an IRQ interrupt is generated at the end of the ISR.
- When the VECTADDR Register is written to.
- To update the priority hardware.

Reading or writing to the register at other times can cause incorrect operation.

Table 18-19. VECTADDR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	VectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	VectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFFF000 + 0x030															

Table 18-20. VECTADDR Fields

BITS	NAME	DESCRIPTION
31:0	VectorAddr	**ISR Address** Reading returns the address of the currently active ISR. Writing clears the interrupt.

18.2.2.10 Default Vector Address Register (DEFVECTADDR)

This register contains the default ISR address. This address is used for non-vectored IRQs.

Table 18-21. DEFVECTADDR Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	Default VectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	Default VectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	0xFFFFF000 + 0x034															

Table 18-22. DEFVECTADDR Fields

BITS	NAME	DESCRIPTION
31:0	Default VectorAddr	**Default ISR Handler Address** Contains the address of the default ISR handler.

18.2.2.11 Vector Address Registers (VECTADDRx)

There are 16 Vector Address Registers, designated VectAddr0 through VectAddr15. Each register contains the ISR vector addresses for that particular vectored IRQ interrupt.

Table 18-23. VECTADDRx Registers

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	VICVectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	VICVectorAddr															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
ADDR	VECTADDR0: 0xFFFFF000 + 0x100 VECTADDR1: 0xFFFFF000 + 0x104 VECTADDR2: 0xFFFFF000 + 0x108 VECTADDR3: 0xFFFFF000 + 0x10C VECTADDR4: 0xFFFFF000 + 0x110 VECTADDR5: 0xFFFFF000 + 0x114 VECTADDR6: 0XFFFFF000 + 0x118 VECTADDR7: 0xFFFFF000 + 0x11C VECTADDR8: 0xFFFFF000 + 0x120 VECTADDR9: 0xFFFFF000 + 0x124 VECTADDR10: 0xFFFFF000 + 0x128 VECTADDR11: 0xFFFFF000 + 0x12C VECTADDR12: 0xFFFFF000 + 0x130 VECTADDR13: 0xFFFFF000 + 0x134 VECTADDR14: 0xFFFFF000 + 0x138 VECTADDR15: 0xFFFFF000 + 0x13C															

Table 18-24. VECTADDRx Fields

BITS	NAME	DESCRIPTION
31:0	VectorAddr	**ISR Vector Addresses** Contains ISR vector addresses.

18.2.2.12 Vector Control Registers (VECTCTRLx)

There are 16 Vector Control Registers, designated VECTCTRL0 through VECTCTRL15. Software uses these registers to assign the desired interrupt to the desired interrupt vector. The interrupt number from Table 18-1 (in hexadecimal) is programmed to the IntSource bits, and the 'E' bit is set to enable that vector.

Vectored interrupts are only generated if the interrupt is enabled in the INTENABLE Register, and the interrupt is programmed to generate an IRQ interrupt in the INTSELECT Register. This prevents multiple interrupts from being generated by a single request if the controller is incorrectly programmed.

Table 18-25. VECTCTRLx Registers

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///										E	IntSource				
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW
ADDR	VECTCTRL0: 0xFFFFF000 + 0x200 VECTCTRL1: 0xFFFFF000 + 0x204 VECTCTRL2: 0xFFFFF000 + 0x208 VECTCTRL3: 0xFFFFF000 + 0x20C VECTCTRL4: 0xFFFFF000 + 0x210 VECTCTRL5: 0xFFFFF000 + 0x214 VECTCTRL6: 0xFFFFF000 + 0x218 VECTCTRL7: 0xFFFFF000 + 0x21C VECTCTRL8: 0xFFFFF000 + 0x220 VECTCTRL9: 0xFFFFF000 + 0x224 VECTCTRL10: 0xFFFFF000 + 0x228 VECTCTRL11: 0xFFFFF000 + 0x22C VECTCTRL12: 0xFFFFF000 + 0x230 VECTCTRL13: 0xFFFFF000 + 0x234 VECTCTRL14: 0xFFFFF000 + 0x238 VECTCTRL15: 0xFFFFF000 + 0x23C															

Table 18-26. VECTCTRLx Fields

BITS	NAME	DESCRIPTION
31:6	///	**Reserved** Reading returns 0. Write the reset value.
5	E	**Vectored Interrupt Enable** Enables the vector interrupt. This bit is cleared on System Reset.
4:0	IntSource	**Interrupt Source Selection** Selects the interrupt source from any of the 32 interrupt sources.

18.2.2.13 Interrupt Test Output Register (ITOP)

Reading the ITOP register returns the status of the IRQ and FIQ interrupt request outputs from the VIC to the ARM exception-handling circuitry.

Table 18-27. ITOP Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								VF	VI	///					
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RW	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFFF000 + 0x30C															

Table 18-28. ITOP Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading returns 0. Write the reset value.
7	VI	**VIC IRQ Output Status** 1 = an IRQ interrupt request to the ARM core is asserted. 0 = an IRQ interrupt request to the ARM core is not asserted.
6	VF	**VIC FIQ Output Status** 1 = an FIQ interrupt request to the ARM core is asserted. 0 = an FIQ interrupt request to the ARM core is not asserted.
5:0	///	**Reserved** Reading returns 0. Write the reset value.

Chapter 19
Watchdog Timer

19.1 Theory of Operation

The Watchdog Timer (WDT) can be used to ensure that the system does not hang in an invalid state. In a normally operating system, the WDT is reset periodically by software. If an event causes software to fail to reset the WDT during a programmed interval, the WDT can cause an interrupt or a system reset. This reset **does not** reset the System PLL nor the USB PLL. The WDT is programmed with a 32-bit timing value in the Control register (CTL:TOP) and decrements that value on each HCLK cycle. Upon underflow, (timing out) the WDT causes either:

- A flag to be set in the Reset, Clock, and Power Controller (RESETSTATUS:WDTO), triggering a system reset, or
- An interrupt is sent to the Vectored Interrupt Controller (VIC).

Note that the interrupt is only recognized in the Active Mode.

Three conditions cause the TOP value to be loaded into the counter:

- After a system reset
- After a counter reset (programming 0x1984 into the RST register)
- After the counter counts down to 0.

When first enabled or when reset, the WDT begins counting from the programmed timing value. The WDT is enabled by programming the CTL:EN bit to 1. The WDT block diagram is shown in Figure 19-1. All descriptions apply to both the LH79524 and LH79525.

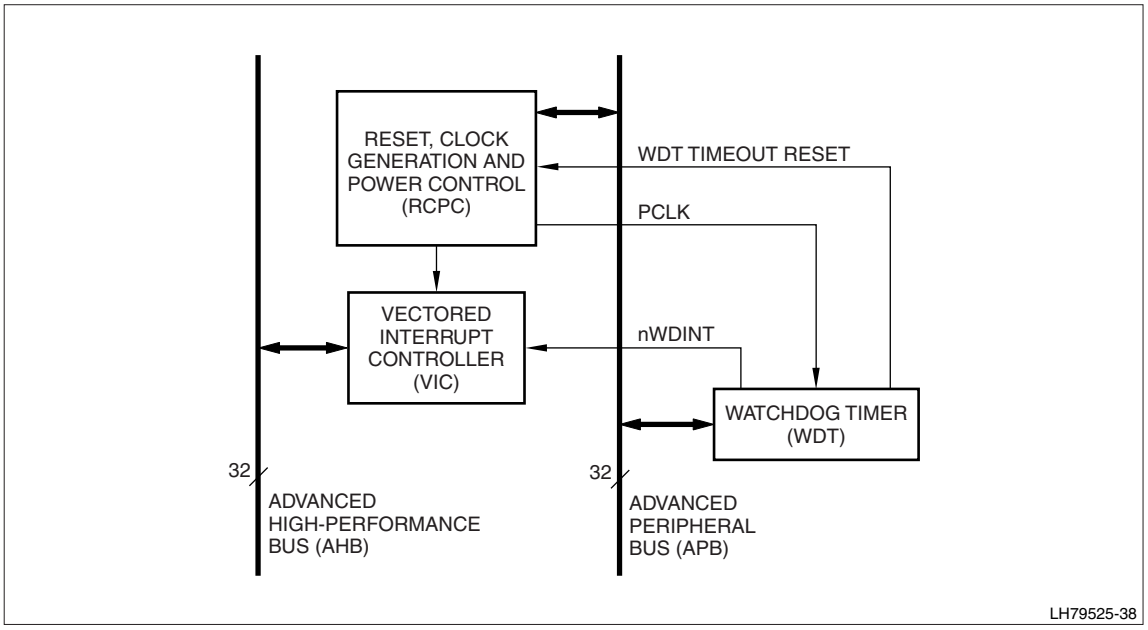


Figure 19-1. Watchdog Timer Block Diagram

19.1.1 WDT Operation Details

The WDT is enabled and disabled by programming the CTL:EN bit to 1. To reset the WDT, program the Reset register (RST) with the value 0x1984. To prevent the WDT from being inadvertently disabled, the Enable function can be locked by setting the CTL Freeze field (CTL:FRZ).

CAUTION

Once set, FRZ can only be cleared by a system reset. Ensure that the WDT will always be serviced by software before freezing the Enable bit.

To configure the initial value, program the CTL Timeout Period field (CTL:TOP). The value in TOP specifies one of 16 time-out periods, ranging from 2^{16} through 2^{31} HCLK cycles. When the WDT is enabled or reset, the value in CTL:TOP is loaded into the Count registers (COUNT[3:0]) and the WDT starts decrementing.

COUNT[3:0] are a set of registers operating as a cascaded counter, reporting the current WDT decrementing value:

- COUNT3 contains bits 31 through 24 of the current value.
- COUNT2 contains bits 23 through 16 of the current value.
- COUNT1 contains bits 15 through 8 of the current value.
- COUNT0 contains bits 7 through 0 of the current value.

When all of COUNT[3:0] are 0, the WDT has timed out. Software can set WDT operation to cause a system reset, or an interrupt followed by a system reset:

- To cause a system reset after one WDT timeout, program the CTL Interrupt First bit (CTL:IF) to 0.
- To generate an interrupt after one WDT timeout, and a reset only if the interrupt is not serviced, program CTL:IF to 1. The first timeout sends an interrupt to the Vectored Interrupt Controller (VIC). This interrupt is also reported in the Status register INT field (STATUS:INT). Unless software services this interrupt and resets the WDT, a second (and subsequent) timeout causes a reset. Interrupts are generated until a counter reset is performed, even if the counter has been disabled after the first interrupt was generated.
  - The VIC allows programming the type of interrupt (IRQ or FIQ) generated by the WDT. Bit 0 in the VIC's INTSELECT register can be programmed to 1 to generate an FIQ, or 0 to generate an IRQ. After reset, this bit is programmed to 0, for an IRQ.

19.2 Register Reference

This section describes the location and programming of the WDT registers.

19.2.1 Memory Map

Register offsets in Table 19-1 are relative to the Timer base address 0xFFFC3000.

Table 19-1. Watchdog Timer Memory Map

ADDRESS OFFSET	NAME	DESCRIPTION
0x00	CTL	Watchdog Control Register
0x04	RST	Watchdog Counter Reset
0x08	STATUS	Watchdog Status Register
0x0C	COUNT0	Current Count bits [7:0]
0x10	COUNT1	Current Count bits [15:8]
0x14	COUNT2	Current Count bits [23:16]
0x18	COUNT3	Current Count bits [31:24]

19.2.2 Register Descriptions

19.2.2.1 Control Register (CTL)

The WDT control register, described in Table 19-2 and Table 19-3 enables and disables the WDT, and specifies the timeout period and interrupt response.

Table 19-2. CTL Register

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								TOP				FRZ	///	IF	EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RW	RW	RO	RW	RW	RW
ADDR	0xFFFE3000 + 0x00															

Table 19-3. CTL Fields

BITS	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7:4	TOP	**Timeout Period** Program this field to select one of 16 possible values to load into the WDT to determine the timeout, in HCLK cycles. The loaded value is $2^{(TOP+16)}$. For example: • TOP = 0x0 results in a timeout period of 2^{16} HCLK cycles • TOP = 0xF results in a timeout period of 2^{31} HCLK cycles When a timeout period is programmed, the new value takes effect after a counter reset command or after the count reaches 0.
3	FRZ	**Freeze** Set this bit while the watchdog is enabled, to prevent clearing EN. Only a System Reset can clear FRZ. 1 = When WDT is enabled, the EN bit is frozen and cannot be cleared (set to 0) 0 = WDT function is not frozen. (FRZ *cannot* be cleared by writing a 0 to this bit; a 0 is only valid when this bit is read. FRZ is only cleared with a System Reset)
2	///	**Reserved** Reading this bit returns invalid data. Write the reset value.
1	IF	**Interrupt First** Program this bit to specify whether the first WDT timeout generates an interrupt or a system reset: 1 = The first timeout generates an interrupt and restarts the WDT. If this interrupt is not cleared by software or by a reset, the second timeout generates a system reset. A system reset clears this interrupt. 0 = Each timeout generates a system reset
0	EN	**Enable** Program this bit to enable or disable the WDT: 1 = Enables the WDT. The counter decrements. Timeouts generate interrupts or system resets, depending on the setting of the IF field. To prevent interrupts or system resets, the WDT *must* be periodically reset. 0 = Disables the WDT. The counter does not decrement. Because no timeouts occur, the WDT generates no interrupts or system resets.

19.2.2.2 Counter Reset Register (RST)

Write this register to reset the WDT, preventing a timeout.

Table 19-4. RST Description

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	RST															
RESET	undefined															
TYPE	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
ADDR	0xFFFE3000 + 0x04															

Table 19-5. RST Field

BIT	NAME	DESCRIPTION
31:16	///	**Reserved** Reading this field returns 0. Write the reset value.
15:0	RST	**Reset** Write 0x1984 to this register to reset the WDT and commence counting down. If the first timeout interrupt is asserted, this write deasserts the interrupt.

19.2.2.3 Status Register (STATUS)

This register, described in Table 19-6 and Table 19-7, provides the status of the WDT interrupts, and allows programming whether a system reset is generated upon the first timeout, or if an interrupt is generated on the first timeout, followed by a system reset if that interrupt is not serviced before a second timeout occurs.

Table 19-6. STATUS Description

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								INT	///	IF		///			
RESET	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	0xFFFE3000 + 0x08															

Table 19-7. STATUS Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7	INT	**INT** This bit reports the WDT timeout interrupt status: 1 = An interrupt has occurred and has been sent to the VIC 0 = No interrupt has occurred
6	///	**Reserved** Reading this bit returns 1. Write the reset value.
5	///	**Reserved** Reading this bit returns invalid data. Write the reset value.
4	IF	**Interrupt First** This bit reports whether the WDT is programmed to assert an interrupt or a reset on the first timeout. This bit duplicates the value of CTL:IF. 1 = The first timeout generates an interrupt and restarts the WDT. If this interrupt is not cleared by software or by a reset, the second timeout generates a system reset. A reset clears this interrupt. 0 = Each timeout generates a system reset
3:0	///	**Reserved** Reading this field returns 0. Write the reset value.

19.2.2.4 Current Watchdog Count Registers (COUNT[3:0])

The COUNTx registers, described in Table 19-8 and Table 19-9, are a set of registers operating as a cascaded counter, reporting the current WDT decrementing value:

- COUNT3 contains bits 31 through 24 of the current value
- COUNT2 contains bits 23 through 16 of the current value
- COUNT1 contains bits 15 through 8 of the current value
- COUNT0 contains bits 7 through 0 of the current value.

When all of COUNT[3:0] are 0x00, the WDT has timed out.

Table 19-8. COUNTx Description

BIT	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIELD	///															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
BIT	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIELD	///								COUNT							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
TYPE	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
ADDR	COUNT0 = 0xFFFE3000 + 0x0C COUNT1 = 0xFFFE3000 + 0x10 COUNT2 = 0xFFFE3000 + 0x14 COUNT3 = 0xFFFE3000 + 0x18															

Table 19-9. COUNTx Fields

BIT	NAME	DESCRIPTION
31:8	///	**Reserved** Reading this field returns 0. Write the reset value.
7:0	COUNT	**Current Count** This byte corresponds to the bit position of the entire 32-bit WDT count, as described (e.g. for COUNT3, this is the most-significant byte).

Chapter 20
Glossary

802.3
See IEEE 802.3.

AC
Audio Codec

AC97
AC97 Codec interface following Intel's AC97 specification.

AHB
Advanced High-Performance Bus. Defined in the AMBA specification, the AHB connects the high-performance blocks. In the LH79524/LH79525, the AHB connects the ARM720 core, and a number of peripherals. The AHB connects to the APB via the APB Bridge. The AHB supports burst mode data transfers and split transactions, and all timing is referenced to a single clock edge.

ALI
Advanced LCD Interface. Allows direct connection to the Row and Column Driver chips in newer superthin panels that do not incorporate a separate timing ASIC.

AMBA
Advanced Microprocessor Bus Architecture. This architecture is an open standard for an on-chip bus connecting the blocks of an MCU.

APB
Advanced Peripheral Bus. Defined in the AMBA specification, the APB connects the lower-performance peripheral blocks. In the LH79524/LH79525, the APB connects a number of peripherals that do not require the speed or bandwidth of the AHB. The APB connects to the AHB via the APB Bridge.

APB Bridge
Connection between the AHB and APB.

ARM720T Core
A microprocessor based on the ARM720T 32-bit RISC CPU, connecting to AMBA compliant interfaces. For more information, see the ARM Ltd. website: http://www.nxp.com/redirect/arm.com.

AUI (Attachment Unit Interface)
Attachment Unit Interface. A 15-pin shielded, twisted pair Ethernet cable used (optionally) to connect between network devices and an MAU. See also 'MAU'.

Back-Off Time

In Ethernet communications, this is a random period of time that a device attempting communication waits following a collision before re-attempting communication.

Big-endian

The most significant part of the data is stored at the lowest storage address or transmitted or received first. See Endianness.

Block

A Block is the on-chip circuitry to implement a peripheral, memory circuit, or processor.

Byte

An 8-bit data element. Bytes in this User's Guide are shown with the most significant bit on the left (or top) and the least significant bit on the right (or bottom). Also see Half Word, Nibble, and Word.

Byte Lane

A data path that is one byte wide.

Checksum

A checksum is generated by adding all data elements of a packet together, and throwing away the carry. The same as done at the receiving end. If the checksums are different, a transmission error occurred. Generally, the receiver asks the transmitter to resend the previous data packet.

Chip

A packaged integrated circuit device.

CLCDC

The on-chip Color Liquid Crystal Display Controller.

Core

See ARM720T Core.

CPSR

Current Program Status Register. In ARM architecture, it stores the condition code bits.

CSMA/CD

Carrier Sense Multiple Access/Collision Detection. This is a network access method in which devices that are ready to transmit data first check the channel for a carrier. If no carrier is sensed, a device can transmit. If two devices do transmit at once, a collision occurs and each computer backs off and waits a random amount of time before attempting to retransmit. This is the access method used by Ethernet. See also MAC and Back Off Time.

DMA

Direct Memory Access. The LH79524/LH79525 includes an on-chip DMA Controller.

ECC

Error Correction Code. This code, appended to a packet or page of data, allows not only detection of errors in the data stream, but correction of the errors as well.

ED

Endpoint Descriptor. A memory structure that describes the information necessary for the USB Host Controller to communicate (via Transfer Descriptors) with a USB Client Endpoint. An ED includes a Transfer Descriptor pointer.

Embedded SRAM

In the LH79524/LH79525, 16KB of on-chip SRAM. The LCD controller has access to an internal frame buffer in embedded SRAM and an extension buffer in SDRAM for dual panel or large displays. The core and DMA controller share the main system bus, providing access to all external memory devices and the embedded SRAM frame buffer.

ENDEC

Encoder and Decoder

Endianness

Describes the bit, byte, or word sequence of data communication or storage, associating the most significant or least significant end of a data sequence with the lowest address or with the beginning of reception or transmission. See Big-endian and Little-endian.

Endpoint Address

The combination of a client Device Address and an Endpoint Number on the USB.

EOF

End Of Frame. The end of a USB-defined frame.

EMC

External Memory Controller. In the LH79524/LH79525, the EMC is an AHB slave block, providing an interface between the AHB and external memory-mapped devices.

FIQ

Fast Interrupt request. FIQs are assigned in the VIC. FIQ interrupts are higher priority than an IRQ. See IRQ.

Frame (USB)

A frame begins with a Start of Frame (SOF) token and is 1.0 ms +/- 0.25% in length.

GPIO

General Purpose Input and Output

Half Word

In the 32-bit LH79524/LH79525, a 16-bit data element structured as an ordered pair of bytes. Half words in this User's Guide are shown with the most significant byte on the left (or top) and the least significant byte on the right (or bottom). Also see Byte, Nibble, and Word.

HCCA

Host Controller Communication Area (USB)

I^2C

Bidirectional, two wire serial bus providing a communication link between integrated circuits.

IEEE 802.3

The IEEE specification defining Ethernet. The full specification is available at: http://standards.ieee.org/getieee802/

Interrupt Controller

A block that provides a uniform way of enabling, disabling, and examining the status of interrupt sources. The LH79524/LH79525 uses a Vectored Interrupt Controller (VIC).

IRQ

Interrupt Request. The hardware responds to an IRQ by saving some registers and moving execution to the address provided by the VIC for interrupt handling software. IRQ interrupts are lower priority than FIQ. See FIQ.

Isochronous Data

A continuous stream of data delivered at a steady rate.

IrDA

A serial, half-duplex optical communications protocol sponsored by the InfraRed Data Association.

Jabber

An Ethernet node continuously sending data. A jabbering station is one whose circuitry or logic has failed, and which has locked up a network channel with incessant transmission.

Little-endian

The least significant part of the data is stored at the lowest storage address or transmitted or received first. The LH79520 uses little-endian byte ordering for storage. See Endianness.

LSB; LSb

Least significant byte (LSB) or least significant bit (LSb) of an ordered sequence.

LSW

Least significant word of an ordered sequence.

Maskable

Can be enabled or disabled. See Non-maskable.

MAC

Media Access Control, a layer of the IEEE 802.3 specification that controls access to the communication channel, and provides framing and error detection. See also CSMA/CD.

MAU (Medium Attachment Unit)

Medium Attachment Unit. The component of an Ethernet interface which provides the actual connection to the Ethernet cable.

MCU

Microcontroller. A single-chip microprocessor system directly supporting peripherals used by an embedded-design product.

Merging Write Buffer

A merging write buffer compacts writes of all widths (byte, half-word, and word) into quad-word bursts which can be efficiently transferred to SDRAM.

MIB

Management Information Base. A database on the network that tracks, records, and corrects performance for each device on the network. See also RMON.

MII (Media Independent Interface)

Media Independent Interface. Standard developed for Fast Ethernet in IEEE 802.3u specification. The Fast Ethernet equivalent to the AUI in 10 Mbits/s Ethernet, allowing different types of Fast Ethernet media to be connected to a Fast Ethernet device via a common interface.

MMC

MultiMediaCard.

MSB; MSb

Most significant byte (MSB) or most significant bit (MSb) of an ordered sequence.

MSW

Most significant word of an ordered sequence.

Nibble

In the LH79524/LH79525, a 4-bit data element. Nibbles in this User's Guide are shown with the most significant bit on the left (or top) and the least significant bit on the right (or bottom). Also see Byte, Half Word, and Word.

Non-maskable

Cannot be disabled. See Maskable.

Non-Volatile Memory

A memory technology that retains its contents when power is removed. Examples are ROM and Flash. Also see Volatile Memory.

PHY (Physical Layer Device)

Physical Layer Device. The name used for a transceiver in Fast Ethernet systems, which is connected to an electronic device.

Pixel

Picture Element. The smallest controllable unit of a matrix LCD display.

RO

Read Only. Values written to RO fields cannot be read back. Writing RO fields can cause unpredictable LH79524/LH79525 operation.

RCPC

Reset, Clock, and Power Controller. This block controls clock generation, reset function and power allocation for low-power modes.

RMON

Remote Monitoring. A network management tool to gather information at a single point.

RTC

Real Time Clock

RW

Read or Write. RW bits or fields can be read from or written to.

SIR

Serial InfraRed

SOF

Start of Frame. The beginning of a USB-defined frame. SOF is the first transaction in each frame. The SOF allows endpoints to identify the start of frame and synchronize internal endpoint clocks to the host.

SSP

Synchronous Serial Port

SWI

Software Interrupt. An interrupt that can be defined by software.

TD

The USB Transfer Descriptor. A TD is a memory structure that describes information necessary for the USB Host Controller to transfer a block of data to or from a client Endpoint.

UART
Universal Asynchronous Receiver and Transmitter

USB
Universal Serial Bus

USB Hub
A device that provides additional connections to the USB.

VBP
Vertical Back Porch. The quantity of inactive lines at the start of a frame, after the vertical synchronization period.

Vectored Interrupt Controller
A Vectored Interrupt Controller (VIC) allows hardware to automatically assign addresses (vectors) for system interrupts, increasing interrupt servicing speed. The LH79524/LH79525 provides up to 16 vectored interrupts, and also allows prioritizing interrupts as well as assigning interrupts as IRQs or FIQs.

VFP
Vertical Front Porch. The quantity of inactive lines at the end of a frame, before the vertical synchronization period.

VLAN
Virtual LAN. A logical, not physical, group of devices, defined by software. VLANs allow network administrators to resegment their networks without physically rearranging the devices or network connections. Workstations on different LANs can be connected using VLAN tagging, where 'tags' are assigned to the frames transmitted by devices.

Volatile Memory
A general term for any memory technology that loses its contents when power is removed. Examples are RAM, SRAM, and SDRAM. See Non-Volatile Memory.

VSW
Vertical Synchronization (Pulse) Width. The quantity of horizontal synchronization lines fed to an LCD panel during its reset time from the bottom of the frame to the top of the next frame.

WDT
Watchdog Timer

WO
Write Only. Write Only fields should not be read as the data is invalid.

Word
In the 32-bit LH79524/LH79525, a 32-bit data element structured as an ordered sequence. Words in this User's Guide are shown with the most significant byte on the left (or top) and the least significant byte on the right (or bottom). Also see Byte, Half Word, and Nibble.

ANNEX A: Disclaimers (11)

1. t001dis100.fm: General (DS, AN, UM, errata)

General — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

2. t001dis101.fm: Right to make changes (DS, AN, UM, errata)

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

3. t001dis102.fm: Suitability for use (DS, AN, UM, errata)

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of a NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

4. t001dis103.fm: Applications (DS, AN, UM, errata)

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

5. t001dis104.fm: Limiting values (DS)

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) may cause permanent damage to the device. Limiting values are stress ratings only and operation of the device at these or any other conditions above those given in the Characteristics sections of this document is not implied. Exposure to limiting values for extended periods may affect device reliability.

6. t001dis105.fm: Terms and conditions of sale (DS)

Terms and conditions of sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, including those pertaining to warranty, intellectual property rights infringement and limitation of liability, unless explicitly otherwise agreed to in writing by NXP Semiconductors. In case of any inconsistency or conflict between information in this document and such terms and conditions, the latter will prevail.

7. t001dis106.fm: No offer to sell or license (DS)

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

8. t001dis107.fm: Hazardous voltage (DS, AN, UM, errata; if applicable)

Hazardous voltage — Although basic supply voltages of the product may be much lower, circuit voltages up to 60 V may appear when operating this product, depending on settings and application. Customers incorporating or otherwise using these products in applications where such high voltages may appear during operation, assembly, test etc. of such application, do so at their own risk. Customers agree to fully indemnify NXP Semiconductors for any damages resulting from or in connection with such high voltages. Furthermore, customers are drawn to safety standards (IEC 950, EN 60 950, CENELEC, ISO, etc.) and other (legal) requirements applying to such high voltages.

9. t001dis108.2.fm: Bare die (DS; if applicable)

Bare die (if applicable) — Products indicated as Bare Die are subject to separate specifications and are not tested in accordance with standard testing procedures. Product warranties and guarantees as stated in this document are not applicable to Bare Die Products unless such warranties and guarantees are explicitly stated in a valid separate agreement entered into by NXP Semiconductors and customer.

10. t001dis109.fm: AEC unqualified products (DS, AN, UM, errata; if applicable)

AEC unqualified products — This product has not been qualified to the appropriate Automotive Electronics Council (AEC) standard Q100 or Q101 and should not be used in automotive critical applications, including but not limited to applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is for the customer’s own risk.

11. t001dis110.fm: Suitability for use in automotive applications only (DS, AN, UM, errata; if applicable)

Suitability for use in automotive applications only — This NXP Semiconductors product has been developed for use in automotive applications only. The product is not designed, authorized or warranted to be suitable for any other use, including medical, military, aircraft, space or life support equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer’s own risk.