MIPS
PR31700
32-bit RISC
microprocessor

MIPS PR31700

32-bit RISC microprocessor

User Manual

Version 0.3

Preliminary specification 1998 Sep 30

Let's make things better.

Philips
Semiconductors

PHILIPS

Contents PR31700 V0.3

Revision History

Version	Date	Changes from Last Version Page
0.1	9/15/97	Original Issue (Draft)
0.2	12/03/97	• Modify the clock module block diagram 75
		• Fix the clock matrix table... 77
		• Rewrite the paragraph on page 99 and 100.................................... 99, 100
		• Correct the description of DIVMOD 145
		• Insert the missing equation of CP rate... 219
0.3	7/14/98	Complete document reformat

Chapter 1 Overview

PR31700 V0.3

This section provides an overview of the features and functions of the PR31700 Processor.

1.1 Overview

The PR31700 Processor is the single-chip, low-cost, integrated embedded processor consists of MIPS R3000 core and system support logic to interface with varieties of devices. Figure 1-1 shows a system block diagram using the PR31700 Processor.

The PR31700 is available today and has a core runs at a doubled speed of 74 MHz. The new PR31700 described in this manual is pin to pin compatible and functionally identical to the PR31500, but enhancement is made in power consumption during idle mode of operation and added the modification to support 8 bit PC-card access. Please refer appendix in the back of the manual for more detail modification and enhancement.

The PR31700 consists of a MIPS R3000 RISC CPU with 4 KBytes of instruction cache memory and 1 KByte of data cache memory, plus integrated functions for interfacing to numerous system components and external I/O modules. The R3000 RISC CPU is also augmented with a multiple/ accumulate module to allow integrated DSP functions, such as a software modem for high-performance standard data and fax protocols. The PR31700 also contains multiple DMA channels and a high-performance and flexible Bus Interface Unit (BIU) for providing an efficient means for transferring data between external system memory, cache memory, the CPU core, and external I/O modules. The types of external memory devices supported include dynamic random access memory (DRAM), synchronous dynamic random access memory (SDRAM), static random access memory (SRAM), Flash memory, read-only memory (ROM), and expansion cards (PCMCIA). The PR31700 also contains a System Interface Module (SIM) containing integrated functions for interfacing to numerous external I/O modules such as liquid crystal displays (LCD's), the UCB1200 ASIC (which handles most of the analog functions of the system, including sound and telecom codecs and touchscreen ADC), ISDN/high-speed serial, infrared, wireless peripherals, etc. Lastly, the PR31700 contains support for implementation of power management for the entire device, whereby various PR31700 internal modules and external subsystems can be individually (under software control) powered up and down. The PR31700 contains the following overall features:

- high level of integration on a single chip, optimized for low cost, small board space, low pin count, low power, and high performance
  - Plastic 208-pin quad flat pack (LQFP) package
  - 32-bit R3000 RISC CPU, cache memory, multiply-accumulate module, multi-channel DMA controller, bus interface unit and memory controller, power management, and other peripheral subsystems all on a single integrated chip
  - minimal number of inter-chip connections
  - 74 MHz version available
- low power consumption
  - maximum peak current running at 73.728 MHz = 100 to 120 mA (estimated)
    - standby current = 10 A (typ.)
  - entire PR31700 operation is 3.3V
  - real-time clock based on 32.768 KHz reference
  - CPU clock stop state for low standby current
  - power-down modes for individual internal peripheral modules

Figure 1-2 shows a block diagram of the PR31700 . The key functions and features for each PR31700 module are briefly described in following sections.

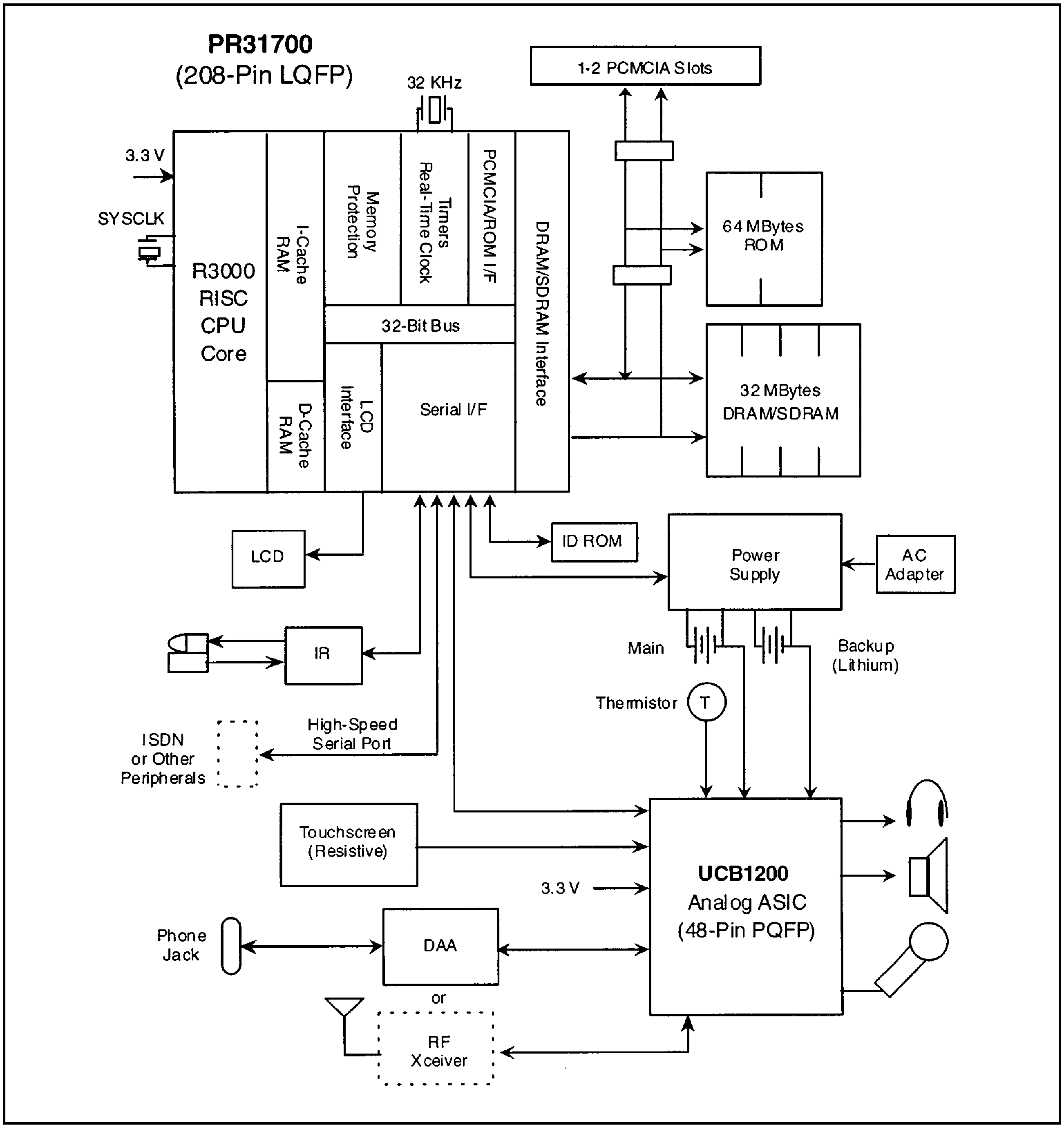


Figure 1-1. System Block Diagram

CPU Module

- R3000 RISC central processing unit core
  - full 32-bit operation (registers, instructions, addresses, etc.)
  - 32 general purpose 32-bit registers; 32-bit program counter
  - MIPS RISC Instruction Set Architecture (ISA) supported
  - 32 x 64-bit wide entries, on-chip Translation Look-aside Buffers (TLBs) provide the virtual-to-physical address translation that is essential in implementing a powerful operating system
- on-chip cache
  - 4 KByte direct-mapped instruction cache (I-cache)
    - physical address tag and valid bit per cache line
    - programmable burst size
    - instruction streaming mode supported
  - 1 KByte two-way associative data cache (D-cache)
    - physical address tag and valid bit per cache line
    - programmable burst size
    - write-through
  - cache address snoop mode supported for DMA
  - four-stage write buffer
- programmable memory protection
  - separate read and write protection control for kernel and user space
  - 8 total protectable regions available, each individually programmable, using breakpoint address, mask, control, and status registers
  - causes address exception on illegal reads or writes
- high-speed multiplier/accumulator
  - on-chip hardware multiplier
  - supports 16x16 or 32x32 multiplier operations, with 64-bit accumulator
  - existing multiply instructions are enhanced and new multiply and add instructions are added to R3000 instruction set to improve the performance of DSP applications
- CPU interface
  - handles data bus, address bus, and control interface between CPU core and the rest of the PR31700 logic

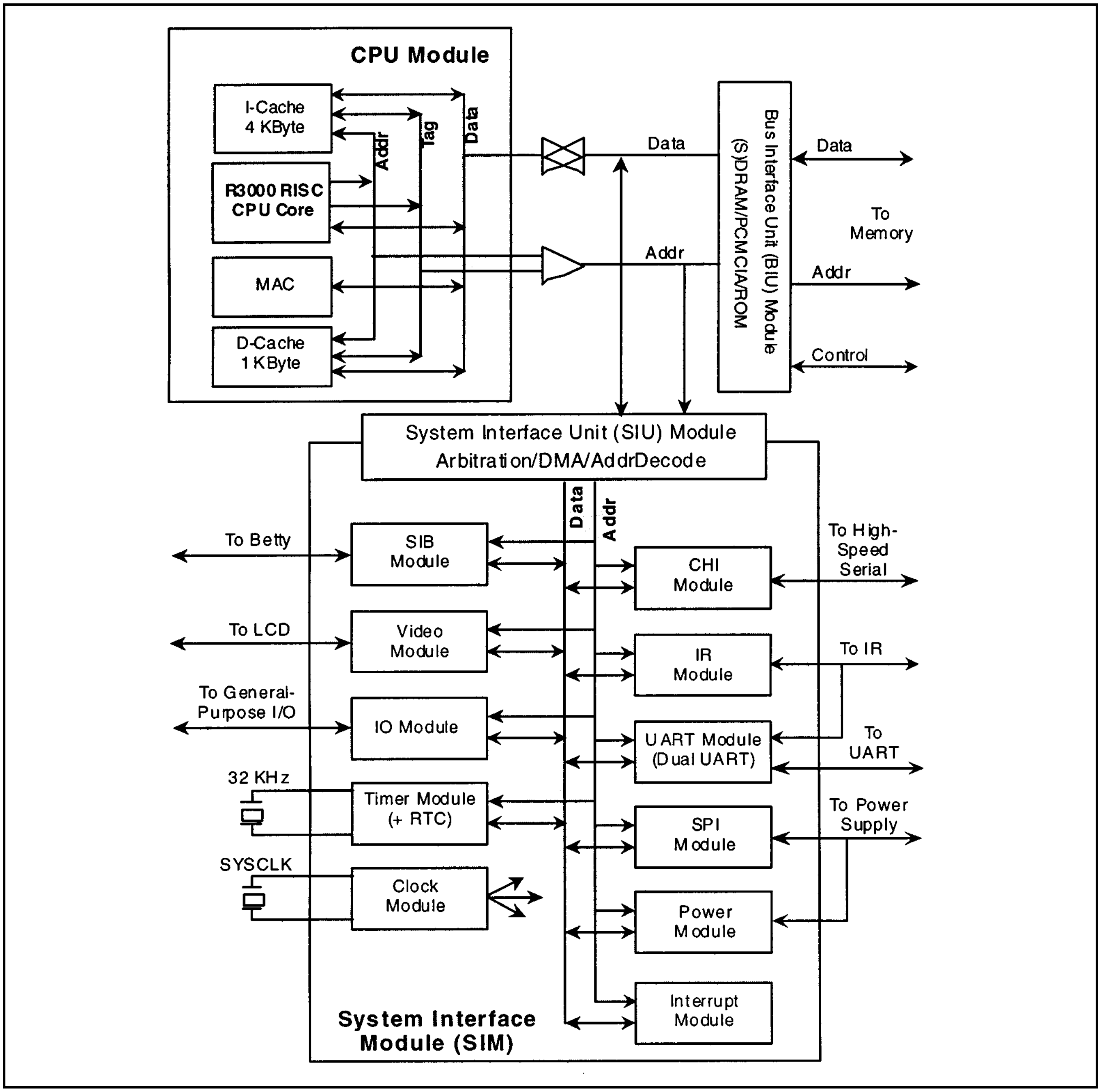


Figure 1-2. PR31700 Processor Block Diagram

BIU Module

- system memory and PR31700 Bus Interface Unit (BIU)
  - supports up to 2 banks of physical memory
  - supports self-refreshing DRAM and SDRAM programmable parameters for each bank of DRAM or SDRAM (row/column address configuration, refresh, burst modes, etc.)
- programmable chip select memory access
  - 4 programmable (size, wait states, burst mode control) memory device and general purpose chip selects
    - available for system ROM, SRAM, Flash
    - available for external port expansion registers
  - 4 programmable (wait states, burst mode control) general purpose chip selects
    - PR31700 provides the chip select and card detect signals
    - supports card insertion/removal timeouts
- supports up to 2 identical full PCMCIA ports
  - PR31700 and UCB1200 provide the control signals and accepts the status signals which conform to the PCMCIA version 2.01 standard
  - appropriate connector keying and level-shifting buffers required for 3.3V versus 5V PCMCIA interface implementations
  - support 16-bit and 8-bit PC-Card access.

SIU Module

- multi-channel 32-bit DMA controller and System Interface Unit (SIU)
- independent DMA channels for video, SIB to/from UCB1200 audio/telecom codecs,
- high-speed serial port, IR UART, and general purpose UART
- address decoding for submodules within System Interface Module (SIM)

Clock Module

- PR31700 supports system-wide single crystal configuration, besides the 32 KHz RTC XTAL (reduces cost, power, and board space)
- common crystal rate divided to generate clock for CPU, video, sound, telecom, UARTs, etc.
- independent enabling or disabling of individual clocks under software control, for power management

CHI Module

- high-speed serial Concentration Highway Interface (CHI) contains logic for interfacing to external full-duplex serial time-division-multiplexed (TDM) communication peripherals
- supports ISDN line interface chips and other PCM/TDM serial devices
- CHI interface is programmable (number of channels, frame rate, bit rate, etc.) to provide support for a variety of formats
- supports data rates up to 4.096 Mbps independent DMA support for CHI receive and transmit

Interrupt Module

- contains logic for individually enabling, reading, and clearing all PR31700 interrupt sources
- interrupts generated from internal PR31700 modules or from edge transitions on external signal pins

IO Module

- contains support for reading and writing the 7 bi-directional general purpose IO pins and the 32 bi-directional multi-function IO pins
- each IO port can generate a separate positive and negative edge interrupt
- independently configurable IO ports allow PR31700 to support a flexible and wide range of system applications and configurations

IR Module

- IR consumer mode
  - allows control of consumer electronic devices such as stereos, TVs, VCRs, etc.
  - programmable pulse parameters
  - external analog LED circuitry
- IrDA communication mode
  - allows communication with other IrDA devices such as FAX machines, copiers, printers, etc.
  - supported by UART module within the PR31700
  - external analog receiver preamp and LED circuitry
  - data rate = up to 115 Kbps at 1 meter
- IR FSK communication mode
  - supported by UART module within the PR31700
  - external analog IR chip(s) perform frequency modulation to generate the desired IR communication mode protocol
  - data rate = up to 36000 bps at 3 meters
- carrier detect state machine
  - periodically enables IR receiver to check if a valid carrier is present

Power Module

- power-down modes for individual internal peripheral modules
- serial (SPI port) power supply control interface supported
- power management state machine has 3 states: ACTIVE, IDLE (DOZING) and SLEEP

SIB Module

- The PR31700 contains holding and shift registers to support the serial interface to the UCB1200 ASIC and/or other optional codec devices
- synchronous, frame-based protocol
- The PR31700 always master source of clock and frame frequency and phase; programmable clock frequency
- each SIB frame consists of 128 clock cycles, further divided into 2 subframes or words of 64 bits each (supports up to 2 devices simultaneously)

- independent DMA support for audio receive and transmit, telecom receive and transmit
- supports 8-bit or 16-bit mono telecom formats
- supports 8-bit or 16-bit mono audio formats
- independently programmable audio and telecom sample rates
- CPU read/write registers for subframe control and status

SPI Module

- provides interface to SPI peripherals and devices
- full-duplex, synchronous serial data transfers (data in, data out, and clock signals)
- The PR31700 supplies dedicated chip select and interrupt for an SPI interface serial power supply
- 8-bit or 16-bit data word lengths for the SPI interface
- programmable SPI baud rate

Timer Module

- Real Time Clock (RTC) and Timer
- 40-bit counter (30.517 sec granularity); maximum uninterrupted time = 388.36 days
- 40-bit alarm register (30.517 sec granularity)
- 16-bit periodic timer (0.868 sec granularity); maximum timeout = 56.8 msec
- interrupts on alarm, timer, and prior to RTC roll-over

UART Module

- 2 independent full-duplex UARTs
- programmable baud rate generator
- UART-A port used for general purpose serial control interface
- UART-B port used for serial control interface to external IR module
- UART-A and UART-B DMA support for receive and transmit

Video Module

- bit-mapped graphics
- supports monochrome, grey scale, or color modes
- time-based dithering algorithm for grey scale and color modes
- supports multiple screen sizes
- supports split and non-split displays
- variable size and relocatable video buffer
- DMA support for fetching image data from video buffer

1.2 References

(1) Philips Semiconductors PR31700 User Manual, Revision 1.0

This document contains a preliminary description of the PR31700 Processor. This document includes pin descriptions, address memory map, theory of operation and block diagrams for each submodule, register assignments and functions, and interface timing diagrams and protocols.

(2) Philips Semiconductors UCB1100/1200 Advanced modem/audio analog front-end, 1997 Jun. 12

This document contains an objective specification of the mixed signal telecomm/audio codec. This document includes pin descriptions, block diagrams, register assignments and functions, and interface timing diagrams.

Chapter 2
Pin Descriptions

This section describes the pins of the PR31700 Processor.

2.1 Overview

The PR31700 processor contains 208 pins consisting of input, output, bi-directional, and power and ground pins. These pins are used to support various functions. The following sections will describe the function of each pin including any special power-down considerations for each pin.

2.2 Pins

The PR31700 Processor contains 208 total pins, consisting of 135 signal pins, 5 spare pins, 34 power pins, and 34 ground pins. Of the 133 signal pins, 32 of them are multi-function and can be independently programmed either as IO ports or for an alternate standard/normal function. As an IO port, any of these pins can be programmed as an input or output port, with the capability of generating a separate positive and negative edge interrupt. See Section 2.3 for a summary of the multi-function IO ports versus their standard functions.

2.2.1 Memory Pins

D(31:0): INPUT/OUTPUT

These pins are the data bus for the system. 8-bit SDRAMs should be connected to bits 7:0 and 16-bit SDRAMs and DRAMs should be connected to bits 15:0. All other 16-bit ports should be connected to bits 31:16. Of course, 32-bit ports should be connected to bits 31:0. These pins are normally outputs and only become inputs during reads, thus no resistors are required since the bus will only float for a short period of time during bus turn-around.

A(12:0): OUTPUT

These pins are the address bus for the system. The address lines are multiplexed and can be connected directly to SDRAM and DRAM devices. To generate the full 26-bit address for static devices, an external latch must be used to latch the signals using the ALE signal. For static devices, address bits 25:13 are provided by the external latch and address bits 12:0 (directly connected from the PR31700's address bus) are held afterward by the PR31700 processor for the remainder of the address bus cycle.

ALE OUTPUT

This pin is used as the address latch enable to latch A(12:0) using an external latch, for generating the upper address bits 25:13.

/RD: OUTPUT

This pin is used as the read signal for static devices. This signal is asserted for reads from /MCS3-0, /CS3-0, /CARD2CS and /CARD1CS for memory and attribute space, and for reads from PR31700 processor accesses if SHOWPR31700 is enabled (for debugging purposes).

/WE: OUTPUT

This pin is used as the write signal for the system. This signal is asserted for writes to /MCS3-0, /CS3-0, /CARD2CS and /CARD1CS for memory and attribute space, and for writes to DRAM and SDRAM.

/CAS0 (/WE0): OUTPUT

This pin is used as the CAS signal for SDRAMs, the CAS signal for D(7:0) for DRAMs, and the write enable signal for D(7:0) for static devices.

/CAS1 (/WE1): OUTPUT

This pin is used as the CAS signal for D(15:8) for DRAMs and the write enable signal for D(15:8) for static devices.

/CAS2 (/WE2): OUTPUT

This pin is used as the CAS signal for D(23:16) for DRAMs and the write enable signal for D(23:16) for static devices.

/CAS3 (/WE3): OUTPUT

This pin is used as the CAS signal for D(31:24) for DRAMs and the write enable signal for D(31:24) for static devices.

/RAS0: OUTPUT
This pin is used as the RAS signal for SDRAMs and the RAS signal for Bank0 DRAMs.

/RAS1 (/DCS1): OUTPUT
This pin is used as the chip select signal for Bank1 SDRAMs and the RAS signal for Bank1 DRAMs.

/DCS0: OUTPUT
This pin is used as the chip select signal for Bank0 SDRAMs.

DCKE: OUTPUT
This pin is used as the clock enable for SDRAMs.

DCLKIN: INPUT
This pin must be tied externally to the DCLKOUT signal and is used to match skew for the data input when reading from SDRAM and DRAM devices.

DCLKOUT: OUTPUT
This pin is the (nominal) 73.728 MHz clock for the SDRAMs.

DQMH: OUTPUT
This pin is the upper data mask for a 16-bit SDRAM configuration.

DQML: OUTPUT
This pin is the lower data mask for a 16-bit SDRAM or 8-bit SDRAM configuration.

/CS3-0: OUTPUT
These pins are the Chip Select 3 through 0 signals. They can be configured to support either 32-bit or 16-bit ports.

/MCS3-0: OUTPUT
These pins are the auxiliary Chip Select 3 through 0 signals. They only support 16-bit ports.

/CARD2CSH,L: OUTPUT
These pins are the Chip Select signals for PCMCIA card slot 2.

/CARD1CSH,L: OUTPUT
These pins are the Chip Select signals for PCMCIA card slot 1.

/CARDREG: OUTPUT
This pin is the /REG signal for the PCMCIA cards.

/CARDIORD: OUTPUT
This pin is the /IORD signal for the PCMCIA IO cards.

/CARDIOWR: OUTPUT
This pin is the /IOWR signal for the PCMCIA IO cards.

/CARDDIR: OUTPUT
This pin is used to provide the direction control for bi-directional data buffers used for the PCMCIA slot(s). This signal will assert whenever /CARD2CSH or /CARD2CSL or /CARD1CSH or /CARD1CSL is asserted and a read transaction is taking place.

/CARD2WAIT: INPUT
This pin is the card wait signal from PCMCIA card slot 2.

/CARD1WAIT: INPUT
This pin is the card wait signal from PCMCIA card slot 1.

2.2.2 Bus Arbitration Pins

/DREQ: **INPUT**

This pin is used to request external arbitration. If the TESTSIU signal is high and the TESTSIU function has been enabled, then once /DGRNT is asserted, external logic can initiate reads or writes to PR31700 processor registers by driving the appropriate input signals. If the TESTSIU signal is low or the TESTSIU function has not been enabled, then PR31700 memory transactions are halted and certain memory signals will be tri-stated when /DGRNT is asserted in order to allow an external master to access memory.

/DGRNT: **OUTPUT**

This pin is asserted in response to /DREQ to inform the external test logic or bus master that it can now begin to drive signals.

2.2.3 Clock Pins

SYSCLKIN: **INPUT**

This pin should be connected along with SYSCLKOUT to an external crystal which is the main PR31700 clock source.

SYSCLKOUT: **OUTPUT**

This pin should be connected along with SYSCLKIN to an external crystal which is the main PR31700 clock source.

C32KIN: **INPUT**

This pin along with C32KOUT should be connected to a 32.768 KHz crystal.

C32KOUT: **OUTPUT**

This pin along with C32KIN should be connected to a 32.768 KHz crystal.

BC32K: **OUTPUT**

This pin is a buffered output of the 32.768 KHz clock.

2.2.4 CHI Pins

CHIFS: **INPUT/OUTPUT**

This pin is the CHI frame synchronization signal. This pin is available for use in one of two modes. As an output, this pin allows the PR31700 to be the master CHI sync source. As an input, this pin allows an external peripheral to be the master CHI sync source and the PR31700 CHI module will slave to this external sync.

CHICLK: **INPUT/OUTPUT**

This pin is the CHI clock signal. This pin is available for use in one of two modes. As an output, this pin allows the PR31700 to be the master CHI clock source. As an input, this pin allows an external peripheral to be the master CHI clock source and the PR31700 CHI module will slave to this external clock.

CHIDOUT: **OUTPUT**

This pin is the CHI serial data output signal.

CHIDIN: **INPUT**

This pin is the CHI serial data input signal.

2.2.5 IO Pins

IO(6:0): **INPUT/OUTPUT**

These pins are general purpose input/output ports. Each port can be independently programmed as an input or output port. Each port can generate a separate positive and negative edge interrupt. Each port can also be independently programmed to use a 16 to 24 msec debouncer.

MFIO(1:0): **INPUT/OUTPUT**

These pins are multi-function input/output ports. Each port can be independently programmed as an input or output port, or can be programmed for multi-function use to support vendor-dependent test signals (for debugging purposes only). Each port can generate a separate positive and negative edge interrupt. Note that 30 other multi-function pins are available for usage as multi-function input/output ports. These pins are named after their respective standard/normal function and are not listed here.

2.2.6 Reset Pins

/CPURES: INPUT

This pin is used to reset the CPU core. This pin should be connected to a switch for initiating a reset in the event that a software problem might hang the CPU core. The pin should also be pulled up to VSTANDBY through an external pull-up resistor.

/PON: INPUT

This pin serves as the Power On Reset signal for PR31700. This signal must remain low when VSTANDBY is asserted until VSTANDBY is stable. Once VSTANDBY is asserted, this signal should never go low unless all power is lost in the system.

2.2.7 Power Supply Pins

ONBUTN: INPUT

This pin is used as the On Button for the system. Asserting this signal will cause PWRCS to set to indicate to the System Power Supply to turn power on to the system. PWRCS will not assert if the PWROK signal is low.

PWRCS: OUTPUT

This pin is used as the chip select for the System Power Supply. When the system is off, the assertion of this signal will cause the System Power Supply to turn VCCDRAM and VCC3 on to power up the system. The Power Supply will latch SPI commands on the falling edge of PWRCS.

PWROK: INPUT

This pin provides a status from the System Power Supply that there is a good source of power in the system. This signal typically will be asserted if there is a Battery Charger supplying current or if the Main Battery is good and the Battery Door is closed. If PWROK is low when the system is powered off, PWRCS will not assert as a result of the user pressing the ONBUTN or an interrupt attempting to wake up the system. If the device is on when the PWROK signal goes low, the software will immediately shut down the system since power is about to be lost. When PWROK goes low, there must be ample warning so that the software can shut down the system before power is actually lost.

PWRINT: INPUT

This pin is used by the System Power Supply to alert the software that some status has changed in the System Power Supply and the software should read the status from the System Power Supply to find out what has changed. These will be low priority events, unlike the PWROK status, which is a high priority emergency case.

VCC3: INPUT

This pin provides the status of the power supply for the ROM, UCB1200, system buffers, and other transient components in the system. This signal will be asserted by the System Power Supply when PWRCS is asserted, and will always be turned off when the system is powered down.

2.2.8 SIB Pins

SIBDIN: INPUT

This pin contains the input data shifted from the UCB1200 and/or external codec device.

SIBDOUT: OUTPUT

This pin contains the output data shifted to the UCB1200 and/or external codec device.

SIBSCLK: OUTPUT

This pin is the serial clock sent to the UCB1200 and/or external codec device. The programmable SIBSCLK rate is derived by dividing down from SIBMCLK.

SIBSYNC: OUTPUT

This pin is the frame synchronization signal sent to the UCB1200 and/or external codec device. This frame sync is asserted for one clock cycle immediately before each frame starts and all devices connected to the SIB monitor SIBSYNC to determine when they should transmit or receive data.

SIBIRQ: INPUT

This pin is a general purpose input port used for the SIB interrupt source from the UCB1200. This interrupt source can be configured to generate an interrupt on either a positive and/or negative edge.

SIBMCLK: **INPUT/OUTPUT**

This pin is the master clock source for the SIB logic. This pin is available for use in one of two modes. First, SIBMCLK can be configured as a high-rate output master clock source required by certain external codec devices. In this mode all SIB clocks are synchronously slaved to the main PR31700 system clock CLK2X. Conversely, SIBMCLK can be configured as an input slave clock source. In this mode, all SIB clocks are derived from an external SIBMCLK oscillator source, which is asynchronous with respect to CLK2X. Also, for this mode, SIBMCLK can still be optionally used as a high-rate master clock source required by certain external codec devices.

2.2.9 SPI Pins

SPICLK: **OUTPUT**

This pin is used to clock data in and out of the SPI slave device.

SPIOUT: **OUTPUT**

This pin contains the data that is shifted into the SPI slave device.

SPIIN: **INPUT**

This pin contains the data that is shifted out of the SPI slave device.

2.2.10 UART and IR Pins

TXD: **OUTPUT**

This pin is the UART transmit signal from the UARTA module.

RXD: **INPUT**

This pin is the UART receive signal to the UARTA module.

IROUT: **OUTPUT**

This pin is the UART transmit signal from the UARTB module or the Consumer IR output signal if Consumer IR mode is enabled.

IRIN: **INPUT**

This pin is the UART receive signal to the UARTB module.

RXPWR: **OUTPUT**

This pin is the receiver power output control signal to the external communication IR analog circuitry.

CARDET: **INPUT**

This pin is the carrier detect input signal from the external communication IR analog circuitry.

2.2.11 Video Pins

FRAME: **OUTPUT**

This pin is the frame synchronization pulse signal between the Video Module and the LCD, and is used by the LCD to return it's pointers to the top of the display. The Video Module asserts FRAME after all the lines of the LCD have been shifted and transferred, producing a full frame of display.

DF: **OUTPUT**

This pin is the AC signal for the LCD. Since LCD plasma tends to deteriorate whenever subjected to a DC voltage, the DF signal is used by the LCD to alternate the polarity of the row and column voltages used to turn the pixels on and off. The DF signal can be configured to toggle on every frame or can be configured to toggle every programmable number of LOAD signals.

LOAD: **OUTPUT**

This pin is the line synchronization pulse signal between the Video Module and the LCD, and is used by the LCD to transfer the contents of it's horizontal line shift register to the LCD panel for display. The Video Module asserts LOAD after an entire horizontal line of data has been shifted into the LCD.

CP: **OUTPUT**

This pin is the clock signal for the LCD. Data is pushed by the Video Module on the rising edge of CP and sampled by the LCD on the falling edge of CP.

VDAT(3:0): OUTPUT

These pins are the data for the LCD. These signals are directly connected to the LCD for 4-bit non-split displays. For 4-bit split and 8-bit non-split displays, an external register is required to demultiplex the 4-bit data into the desired 8 parallel data lines needed for the LCD.

DISPON: OUTPUT

This pin is the display-on enable signal for the LCD.

VIDDONE: OUTPUT

This pin will provide a synchronization pulse external to the PR31700 that can be used to externally synchronize events to periods when the video is not shifting.

2.2.12 Endianess Pin

/LBENDIAN: INPUT

This pin is used to select the endianess of the PR31700. The pin should be tied to GND for the little endian and VCC for the big endian mode of operation.

2.2.13 Test Pins

TESTSIU: INPUT

The TESTSIU pin is used for two functions in the PR31700. The first function is to define if the Boot ROM is 16 or 32 bits wide. The second function is to enable or disable the Test Bypass Mode. This second mode allows external logic to initiate read or write transactions to the PR31700 registers. The TESTSIU mode is enabled by toggling this signal after the device has powered up. Once the function is enabled, if the TESTSIU pin is high when the bus is arbitrated (using /DREQ and /DGRNT), then external logic can initiate read and write transactions to the PR31700 registers. This pin is used for debugging purposes only. (refer to section 3.4 Test Modes for more detail for both function)

TESTCPU: INPUT

This pin allows numerous internal CPU core signals to be brought to the external PR31700 pins, in place of the normal signals assigned to these pins. The CPU core signals assigned to their respective pins during TESTCPU mode are vendor-dependent. The TESTCPU mode is enabled by asserting this TESTCPU signal, and this function is provided for generating test vectors for the CPU core. This pin is used for debugging purposes only and should be tied to GND for normal operation.

TESTIN: INPUT

This pin is reserved for vendor-dependent use and used for debugging purposes only. This pin should be tied to GND for enabling internal Phase Lock Loop (PLL) for normal operation.

2.2.14 Spare Pins

NC5-1: NO CONNECT

These pins are reserved for future use and should be left unconnected.

RSRV1: RESERVED

This pin is reserved for future use and should be connected to ground.

2.2.15 Power Supply Pins

VDD (34 each): +3.3V

These pins are the power pins for the PR31700 and should be connected to the digital +3.3V power supply VSTANDBY.

VSS (34 each): GND

These pins are the ground pins for the PR31700 and should be connected to digital ground.

NOTE: For some vendor-dependent implementations of PR31700, pin 131 may be used for a filter capacitor for the SYSCLK oscillator (capacitor connected between pin 131 and digital ground).

2.3 Pin Usage Information

This section contains tables summarizing various aspects of the pin usage for the PR31700. Table 2-1 lists the standard/normal versus multi-function usage for each the PR31700 pin, if applicable. Those signal names shown in parentheses are test signals for debugging purposes only. The column showing the multi-function select signal and reset state indicates the internal control signal used to select the multi-function mode, as well as the default configuration of each multi-function pin during reset. The "Bus Arb State" column shows which pins are tri-stated whenever the /DGRNT signal is asserted in response to a /DREQ (external bus arbitration request).

Table 2-1. PR31700 Standard and Multi-Function Pin Usage

PR31700 pinout	standard function (I = input, O = output)	multi-function	multi-function select (reset state: 1 = multi-function mode selected; 0 = standard mode selected;	Bus Arb State
d[31:0]	d[31:0]	x	x	Z
a[12:0]	a[12:0]	x	x	Z
ale	ale (O)	x	x	Z
/rd	/rd (O)	x	x	Z
/we	/we (O)	x	x	Z
/cas0	(/we0) (O)	x	x	Z
/cas1	(/we1) (O)	x	x	Z
/cas2	(/we2) (O)	x	x	Z
/cas3	(/we3) (O)	x	x	Z
/ras0	/ras0 (O)	x	x	Z
/ras1	(/dcs1) (O)	x	x	Z
/dcs0	/dcs0 (O)	x	x	Z
dcke	dcke (O)	x	x	Z
dclkin	dclkin (I)	x	x	
dclkout	dclkout (O)	x	x	Z
dqmh	dqmh (O)	x	x	Z
dqml	dqml (O)	x	x	Z
/dreq	/dreq (I)	mio[27]	miosel[27] (0)	
/dgrnt	/dgrnt (O)	mio[26]	miosel[26] (0)	
sysclkin	sysclkin (I)	x	x	
sysclkout	sysclkout (O)	x	x	
c32kin	c32kin (I)	x	x	

Table 2-1. PR31700 Standard and Multi-Function Pin Usage (Continued)

PR31700 pinout	standard function (I = input, O = output)	multi-function	multi-function select (reset state: 1 = multi-function mode selected; 0 = standard mode selected;	Bus Arb State
c32kout	c32kout (O)	x	x	
bc32k	bc32k (O)	mio[25]	miosel[25] (1)	
vdat[3]	vdat[3] (O)	(berr)	IrqTest (0)	
vdat[2]	vdat[2] (O)	(cpu_stall)	IrqTest (0)	
vdat[1]	vdat[1] (O)	(irqhigh)	IrqTest (0)	
vdat[0]	vdat[0] (O)	(irqlow)	IrqTest (0)	
cp	cp (O)	x	x	
load	load (O)	x	x	
df	df (O)	x	x	
frame	frame (O)	x	x	
dispon	dispon (O)	x	x	
viddone	viddone (O)	x	x	
pwrcs	pwrcs (O)	x	x	
pwrint	pwrint (I)	x	x	
pwrok	pwrok (I)	x	x	
onbutn	onbutn (I)	x	x	
/cpures	/cpures (I)	x	x	
/pon	/pon (O)	x	x	
txd	txd (O)	mio[24]	miosel[24] (0)	
rxd	rxd (O)	mio[23]	miosel[23] (0)	
/cs0	/cs0 (O)	x	x	Z
/cs1	/cs1 (O)	mio[22]	miosel[22] (0)	
/cs2	/cs2 (O)	mio[21]	miosel[21] (0)	
/cs3	/cs3 (O)	mio[20]	miosel[20] (0)	
/mcs0	/mcs0 (O)	mio[19]	miosel[19] (1)	
/mcs1	/mcs1 (O)	mio[18]	miosel[18] (1)	

Table 2-1. PR31700 Standard and Multi-Function Pin Usage (Continued)

PR31700 pinout	standard function (I = input, O = output)	multi-function	multi-function select (reset state: 1 = multi-function mode selected; 0 = standard mode selected;	Bus Arb State
/mcs2	/mcs2 (O)	mio[17]	miosel[17] (1)	
/mcs3	/mcs3 (O)	mio[16]	miosel[16] (1)	
chifs	chifs (I/O)	mio[31]	miosel[31] (1)	
chiclk	chiclk (I/O)	mio[30]	miosel[30] (1)	
chidout	chidout (I/O)	mio[29]	miosel[29] (1)	
chidin	chidin (I/O)	mio[28]	miosel[28] (1)	
vcc3	vcc3 (I)	x	x	
io6	io6 (I/O)	x	x	
io5	io5 (I/O)	x	x	
io4	io4 (I/O)	x	x	
io3	io3 (I/O)	x	x	
io2	io2 (I/O)	x	x	
io1	io1 (I/O)	x	x	
io0	io0 (I/O)	x	x	
spiclk	spiclk (O)	mio[15]	miosel[15] (0)	
spiout	spiout (O)	mio[14]	miosel[14] (0)	
spiin	spiin (I)	mio[13]	miosel[13] (0)	
sibsync	sibsync (O)	x	x	
sibdout	sibdout (O)	x	x	
sibdin	sibdin (I)	x	x	
sibmclk	sibmclk (I/O)	mio[12]	miosel[12] (0)	
sibsclk	sibsclk (O)	x	x	
sibirq	sibirq (I)	x	x	
rxpwr	rxpwr (O)	x	x	
cardet	cardet (I/O)	x	x	
irout	irout (O)	x	x	

Table 2-1. PR31700 Standard and Multi-Function Pin Usage (Continued)

PR31700 pinout	standard function (I = input, O = output)	multi-function	multi-function select (reset state: 1 = multi-function mode selected; 0 = standard mode selected;	Bus Arb State
irin	irin	x	x	x
testsiu	testsiu	x	x	x
testcpu	testcpu	x	x	x
testin	testin	x	x	x
/cardreg	/cardreg (O) (showdino /CS)	mio[11]	miosel[11] (1)	
/cardiowr	/cardiowr	mio[10]	miosel[10] (1)	
/cardiord	/cardiord	mio[9]	miosel[9] (1)	
/card1csl	/card1csl (O)	mio[8]	miosel[8] (1)	
/card1csh	/card1csh (O)	mio[7]	miosel[7] (1)	
/card2csl	/card2csl (O)	mio[6]	miosel[6] (1)	
/card2csh	/card2csh (O)	mio[5]	miosel[5] (1)	
/card1wait	/card1wait (I)	mio[4]	miosel[4] (1)	
/card2wait	/card2wait (I)	mio[3]	miosel[3] (1)	
/carddir	/carddir (O)	mio[2]	miosel[2] (1)	
mfio[1]	(master)	mio[1]	miosel[1] (1)	
mfio[0]	(cpu_data_cyc)	mio[0]	miosel[0] (1)	
nc[6:1]	spare	x	x	
Reseved[0]	spare	x	x	
vdd - 34 each	+3.3V	x	x	
vss - 34 each	GND	x	x	

Table 2-2 lists various power-down states and conditions for each PR31700 pin. The "Power-Down Control" column shows the conditions which trigger a power-down for each respective pin. This column also shows the reset state for each of these conditions.

The "/PON state" column defines the state of each pin at power-on reset (/PON). This condition is defined as initial power up of the PR31700, whereby the PR31700 is initialized and the PR31700 pins are reset to the state shown in the table. This state is entered after power is applied for the very first time (VSTANDBY is turned on but VCC3 is still turned off).

The "1st-time power-up state" column defines the state of each pin after power-up mode (RUNNING STATE) is executed for the first time. This mode is defined as VCC3 applied to the entire system and is initiated by the user pressing the ONBUTN while in the power-on reset (/PON) state. Note that the defined state of various pins for 1st-time power-up may depend on the configuration of external devices attached to these pins. After 1st-time power-up, the software could change the state of various pins to be different from those shown in the table. Thereafter, subsequent transitions from SLEEP STATE to RUNNING STATE might result in different states for these pins. The "power-down state" column defines the state of each pin during power-down mode (SLEEP STATE). This mode is defined as VCC3 turned off to the entire system, except for the PR31700 (RTC and interrupts alive) and any persistent memory.

Table 2-2. PR31700 Power-Down Pin Usage

PR31700 pinout	Power-Down Control powerdown = /(vccon&vcc3) (reset state)	/PON state	1st time power-up state	power-down state
d[31:0]	mempowerdown (1)	low	low	low
a[12:0]	mempowerdown (1)	low	low	low
ale	x	low	low	low
/rd	powerdown	low	hi	low
/we	mempowerdown (1)	low	low	low
/cas0 (/we0)	mempowerdown (1)	low	low	low
/cas1 (/we1)	mempowerdown (1)	low	low	low
/cas2 (/we2)	mempowerdown (1)	low	low	low
/cas3 (/we3)	mempowerdown (1)	low	low	low
/ras0	mempowerdown (1)	low	low	low
/ras1 (/dcs1)	mempowerdown (1)	low	low	low
/dcs0	mempowerdown (1)	low	low	low
dcke	mempowerdown (1)	low	low	low
dclkin	x	x	x	x
dclkout	mempowerdown (1)	low	low	low
dqmh	mempowerdown (1)	low	low	low
dqml	mempowerdown (1)	low	low	low

Table 2-2. PR31700 Power-Down Pin Usage (Continued)

PR31700 pinout	Power-Down Control powerdown = /(vccon&vcc3) (reset state)	/PON state	1st time power-up state	power-down state
/dreq	powerdown & miopd[27] (1)	pull-down	x	selectable
/dgrnt	powerdown & miopd[26] (0)	low	hi	selectable
sysclkin	powerdown	osc off	osc on	osc off
sysclkout	powerdown	osc off	osc on	osc off
c32kin	x	osc on	osc on	osc on
c32kout	x	osc on	osc on	osc on
bc32k	powerdown & miopd[25] (1)	pull-down	x	selectable
vdat[3]	module disable	low	low	low
vdat[2]	module disable	low	low	low
vdat[1]	module disable	low	low	low
vdat[0]	module disable	low	low	low
cp	module disable	low	low	low
load	module disable	low	low	low
df	module disable	low	low	low
frame	module disable	low	low	low
dispon	module disable	low	low	low
viddone	x	low	low	low
pwrcs	x	low	hi	low
pwrint	x	x	x	x
pwrok	x	x	x	x
onbutn	x	x	x	x
/cpures	x	x	x	x
/pon	x	x	x	x
txd	powerdown & miopd[24] (0)	low	low	selectable
rxd	powerdown & miopd[23] (1)	pull-down	in	selectable
/cs0	powerdown	pull-down	hi	pull-down
/cs1	powerdown & miopd[22] (1)	pull-down	hi	selectable

Table 2-2. PR31700 Power-Down Pin Usage (Continued)

PR31700 pinout	Power-Down Control powerdown = /(vccon&vcc3) (reset state)	/PON state	1st time power-up state	power-down state
/cs2	powerdown & miopd[21] (1)	pull-down	hi	selectable
/cs3	powerdown & miopd[20] (1)	pull-down	hi	selectable
chifs	powerdown & miopd[31] (1)	pull-down	in	selectable
chiclk	powerdown & miopd[30] (1)	pull-down	in	selectable
chidout	powerdown & miopd[20] (1)	pull-down	in	selectable
chidin	powerdown0	pull-down	x	selectable
vcc3	powerdown	pull-down	x	pull-down
io6	powerdown & iopd[6] (1)	pull-down	x	selectable
io5	powerdown & iopd[5] (1)	pull-down	x	selectable
io4	powerdown & iopd[4] (1)	pull-down	x	selectable
io3	powerdown & iopd[3] (1)	pull-down	x	selectable
io2	powerdown & iopd[2] (1)	pull-down	x	selectable
io1	powerdown & iopd[1] (1)	pull-down	x	selectable
io0	powerdown & iopd[0] (1)	pull-down	x	selectable
spiclk	powerdown & miopd[15] (0)	low	low	selectable
spiout	powerdown & miopd[14] (0)	low	low	selectable
spiin	powerdown & miopd[13] (1)	pull-down	x	selectable
sibsync	powerdown	low	low	low
sibdout	powerdown	low	low	low
sibdin	powerdown	pull-down	x	pull-down
sibmclk	powerdown & miopd[12] (1)	pull-down	in	selectable
sibsclk	powerdown	low	low	low
sibirq	powerdown	pull-down	x	pull-down
rxpwr	powerdown	low	low	low
cardet	powerdown	pull-down	x	pull-down
irout	powerdown	low	low	low
irin	powerdown	pull-down	x	pull-down

Table 2-2. PR31700 Power-Down Pin Usage (Continued)

PR31700 pinout	Power-Down Control powerdown = /(vccon&vcc3) (reset state)	/PON state	1st time power-up state	power-down state
testsiu	x	x	x	x
testcpu	x	x	x	x
testin	x	x	x	x
/cardreg	powerdown & miopd[11] (1)	pull-down	x	selectable
/cardiowr	powerdown & miopd[10] (1)	pull-down	x	selectable
/cardiord	powerdown & miopd[9] (1)	pull-down	x	selectable
/card1csl	powerdown & miopd[8] (1)	pull-down	x	selectable
/card1csh	powerdown & miopd[7] (1)	pull-down	x	selectable
/card2csl	powerdown & miopd[6] (1)	pull-down	x	selectable
/card2csh	powerdown & miopd[5] (1)	pull-down	x	selectable
/card1wait	powerdown & miopd[4] (1)	pull-down	x	selectable
/card2wait	powerdown & miopd[3] (1)	pull-down	x	selectable
/carddir	powerdown & miopd[2] (1)	pull-down	x	selectable
mfio[1]	powerdown & miopd[1] (1)	pull-down	x	selectable
mfio[0]	powerdown & miopd[0] (1)	pull-down	x	selectable
nc[5:1]	x	x	x	x
Reserved[0]	x	x	x	x
vdd - 34 each	x	x	x	x
vss - 34 each	x	x	x	x

Chapter 3
CPU Module

PR31700 V0.3

This section provides a description of the interface logic to the embedded R3000 CPU core.

3.1 Overview

The PR31700 consists of an embedded MIPS R3000 core along with the System Interface Logic. The R3000 core includes a 4 KByte instruction cache and a 1 KByte data cache used to improve performance, a 16X16/32X32 multiplier with accumulator used to perform DSP functions to support features such as software modems, and a memory protection scheme used to protect regions of user data and also used to set break points for software debugging. Directly interfacing to the CPU core is the CPU Interface logic. The CPU Interface logic consists of a 4-deep Write Buffer used to speed up write transactions, DMA Arbitration logic used to arbitrate with the CPU core for SIU DMA requests and also used to provide cache coherency snooping during DMA operations, and Interface Controller logic used to provide the necessary control to the CPU core and to provide a simple interface for initiating memory transactions to the BIU.

3.1.1 Related Pins

TESTCPU: **INPUT**

This pin allows numerous internal CPU core signals to be brought to external PR31700 pins, in place of the normal signals assigned to these pins. The TESTCPU mode is enabled by asserting this TESTCPU signal, and this function is provided for generating test vectors for the CPU core. This pin is used for debugging purposes only.

3.2 CPU Core

3.2.1 Description

There are several enhancements to the CPU core, these additional key features of the CPU core are:

- Translation Look-aside Buffer (TLB).
- 4 KByte direct-mapped instruction cache (I-cache)
  - 16 bytes (4 words) per line (256 lines total)
  - physical address tag per cache line
  - single valid bit per cache line
  - programmable burst size (16 bytes to 128 bytes) instruction streaming mode supported
- 1 KByte data cache (D-cache)
  - data cache architecture is vendor-dependent
  - 4 bytes (1 word) per line (128 lines total)
  - physical address tag per cache line
  - single valid bit per cache line
  - programmable burst size (16 bytes to 128 bytes); burst refill can be disabled
  - write-through
- cache address snoop mode supported for DMA
- separate read and write protection control for kernel and user space
  - 8 total protectable regions available, each individually programmable, using:
    - eight 30-bit breakpoint address registers
    - eight 30-bit breakpoint mask registers
    - eight breakpoint control registers
    - eight breakpoint status registers
  - causes address exception on illegal reads or writes
    - writes disallowed to illegal space
- 16x16 on-chip hardware multiplier
  - supports 16x16 (single-cycle) or 32x32 (multi-cycle) multiplier operations, with 64-bit accumulator
  - signed or unsigned data formats
  - accumulator supports sign-extension and overflow status
  - existing multiply instructions are enhanced and new multiply and add instructions are added to R3000 instruction set to improve the performance of DSP applications.
- Enhance ISA Implementation.

3.3 CPU Interface

3.3.1 Block Diagram

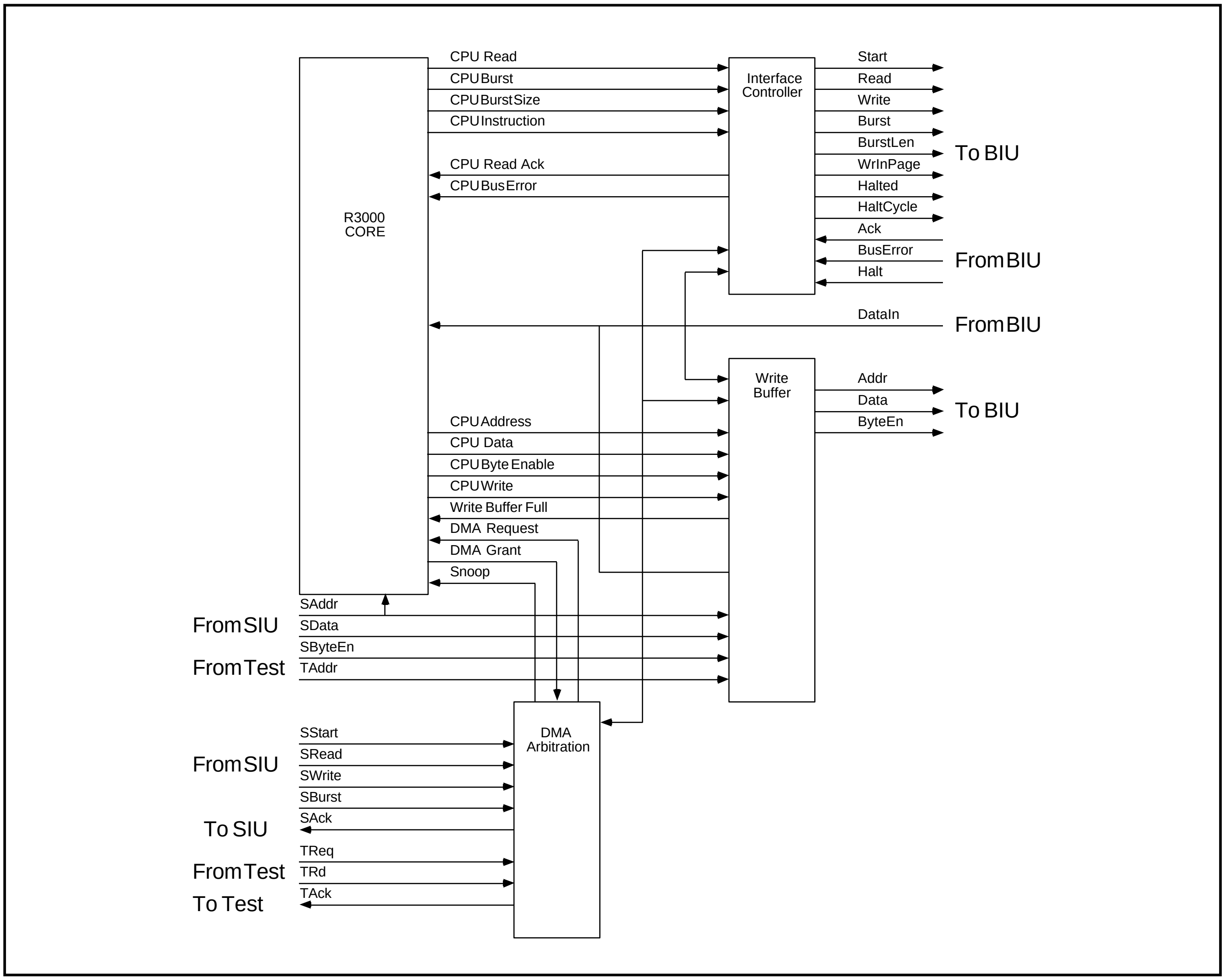


Figure 3-1. CPU Interface Block Diagram

The CPU Interface logic consists of the Interface Controller, Write Buffer and DMA Arbitration. These blocks provide the CPU core with the mechanism to interface to the BIU to launch memory transactions as well as provide the SIU with the interface to launch DMA transactions to and from the BIU. Also provided is a Test Interface that allows reads and writes from and to internal PR31700 registers to bypass the CPU core.

3.3.2 Write Buffer

The CPU core writes into the Write Buffer. The Write Buffer can store up to four write transactions and contains registers to buffer the Address, Data and Byte Enables. As long as there is space in the Write Buffer, the processor will not stall. Once the Write Buffer is full, the CPU is stalled for subsequent writes until there is space in the Write Buffer. When the Write Buffer is not empty, the Write Buffer informs the Interface Controller to start a write transaction to the BIU. Once an acknowledge (Ack) is received from the BIU, the Write Buffer will proceed to the next Address, Data and Byte Enable in the buffer. A Bus Error from the BIU will also cause the Write Buffer to proceed to the next Address, Data and Byte Enable. If the addresses of two consecutive writes are within a 512-byte page, the Interface Controller will assert the WrInPage signal. This signal allows the BIU to keep the SDRAM and/or DRAM in page mode while writing within a 512-byte page.

3.3.3 Interface Controller

The Interface Controller logic contains a state machine to control the interface between the CPU core and the BIU. Read transactions are initiated when the CPU core asserts the CPU Read signal. Write transactions are initiated whenever the Write Buffer is not empty. If the CPU attempts to start a read cycle while the Write Buffer is not empty, the write transaction will complete first and the remaining writes in the write buffer are flushed before allowing the read cycle to occur. The one exception is if the read is an instruction fetch from a cacheable location. In this case, the read will take place before the write buffer is flushed. If the Interface Controller is in the middle of an In Page Write (WrInPage asserted), then one more write transaction must occur to take the BIU out of page mode before the read cycle can occur.

The Interface Controller initiates a read or write to the BIU by asserting the Start signal. The Read and Write signals will indicate whether the BIU should complete a read or write transaction. The CPU core will perform burst reads to fill the instruction and data caches. The CPU core can burst either 4, 8, 16 or 32 words depending on how the CPU core is configured. The state machine inside the Interface Controller will assert the Burst signal to the BIU and provide the burst size to the BIU using the BurstLen control signals. Bursting can greatly enhance the memory bandwidth because the BIU can keep the SDRAM and/or DRAM devices in page mode during a burst read.

The BIU will acknowledge that it is finished with a read or write by asserting the Ack signal. The Ack signal will also indicate that the data input DataIn is valid during a read. During a burst, there will be one Ack for each word of data. If the address that is provided to the BIU via the Addr signals is not mapped into any physical space that is supported by the Address Decoder logic in the BIU, the BusError signal will be asserted by the Watch Dog Timer in the BIU to terminate the bus cycle. The Interface Controller logic will use the BusError signal in place of the Ack signal to end the bus cycle and proceed to the next transaction. If the BusError occurs on a processor read, the CPU Bus Error signal will be asserted to indicate to the CPU that a Bus Error has occurred. For writes the CPU Bus Error will not be asserted since the write would have long ago been completed into the write buffer and the Bus Error will not provide meaningful information.

The BIU must halt the Interface Controller to prevent a bus cycle from starting in order to insert refresh, or when an external device requests the memory bus by asserting the /DMAREQ pin, or when the memory interface is powered down by asserting the MEMPOWERDOWN control bit in the Memory Configuration 4 register. The BIU will halt the Interface Controller by asserting the Halt signal. The Interface Controller must then assert the Halted signal once the state machine has completed the current memory transaction. Once the BIU sees the Halted signal asserted, it can proceed to insert refresh or provide a /DMAGRNT if an external device has requested the memory bus. Once refresh is finished or the external device de-asserts the /DMAREQ signal, the BIU will release the Halt signal. If the Halt signal is asserted due to the MEMPOWERDOWN signal being set then the BIU will not release the Halt signal until the HaltCycle signal is asserted by the Interface Controller.

The HaltCycle signal will be set whenever the Interface Controller wishes to start a read or write cycle, but the Halt signal is asserted. When the BIU sees the HaltCycle signal asserted, the BIU will take one of two possible actions.

On the one hand, if the address indicated by the Addr signals corresponds to an internal PR31700 register, the BIU will immediately de-assert the Halt signal to allow the transaction to begin. Once the Start signal is asserted, the BIU will immediately re-assert the Halt signal to prevent subsequent memory transactions from beginning. On the other hand, if the address does not correspond to an internal PR31700 register, the BIU must bring the memory interface out of power-down mode. This will take several cycles because the SDRAM and/or DRAM devices must be brought out of self refresh mode. Once the memory interface is brought out of power-down mode, the BIU will de-assert the Halt signal. The action taken if the address is a PR31700 register allows the CPU core to access internal PR31700 registers without having to take the memory interface out of power-down mode. This is useful if the processor is running a program out of cache that only requires access to internal PR31700 registers.

3.3.4 DMA Arbitration

The DMA Arbitration logic allows the SIU to launch DMA transactions to the BIU. The SIU will initiate a DMA transaction by pulsing the SStart signal. Once this occurs, a state machine inside the DMA Arbitration logic will assert the DMA Request signal to the CPU core. The CPU core will then assert the DMA Grant signal to indicate that the CPU is now idling. If the SIU is performing a DMA write (as indicated by the SWrite signal), the SAddr, SData and SByteEn signals will then be loaded into the Write Buffer once there is space available in the Write Buffer. As soon as the Write Buffer loads the signals, the DMA Arbitration logic will assert the SAck signal to indicate to the SIU that the DMA write is complete.

At the same time that the data is loaded into the Write Buffer, the Snoop signal is asserted to the CPU core. The snoop logic in the CPU core will invalidate the cache locations defined by the SAddr bits when the Snoop signal is asserted. This provides cache coherency during DMA to simplify managing the cache.

For a DMA read transaction, the SRead signal will be asserted by the SIU. The Write Buffer will always be flushed before allowing the DMA read to begin. Once the Write Buffer is empty, the DMA read will begin and the SAck signal will be asserted to end the DMA read once the BIU asserts the Ack signal. The SIU will receive data input via the DataIn bus. The SBurst signal from the SIU is used to initiate burst reads. The SIU will only request 4-word bursts, thus the BurstLen signals will be set to a burst length of 4 whenever the SBurst signal is asserted during DMA reads.

3.3.5 CPU Stop Mode

In order to reduce power dissipation in the CPU core, the clock to the CPU core can be disabled. This is possible since the CPU core designs are implemented as fully static. The CPU core will normally be in Stop Mode and will only be active to process interrupts. CPU Stop Mode is entered by the CPU setting the STOPCPU control bit in the Power Control Register. The STOPCPU control bit will go to the DMA Arbitration logic, which will assert DMA Request to the CPU core upon the STOPCPU signal being asserted. Once the CPU core responds with a DMA Grant, the DMA Arbitration logic will assert a signal FSTOPCPU that is given to the Clock Generator to disable the clock to the CPU core.

During CPU Stop Mode, DMA transactions from the SIU can still take place. In addition, snooping during CPU Stop Mode may be possible depending on the CPU core implementation. Some CPU core implementations may provide two levels of clock shut down that would allow for snooping to occur even though the rest of the core does not have a clock. Other CPU core implementations may not provide snooping while the clock is disabled. Refer to the appropriate CPU core specifications for more information. If snooping is not possible during CPU Stop Mode, the software must flush the data cache whenever exiting CPU Stop Mode.

3.4 Test Modes

There are two test modes provided in the PR31700 to simplify generating test vectors. They are Test Bypass Mode and CPU Test Mode.

3.4.1 Test Bypass Mode

The TESTSIU pin is used for two functions in the PR31700. The first function is to define if the Boot ROM is 16 or 32 bits wide. The second function is to enable or disable the Test Bypass Mode. Test Bypass Mode allows external test logic to read and write the PR31700 registers without using the CPU core.

If the TESTSIU pin is asserted during RESET, the BIU will assume a 32-bit Boot ROM. If the TESTSIU pin is cleared during RESET, the BIU will assume a 16-bit Boot ROM. As long at the TESTSIU pin remains static (either high or low), the Test Bypass Mode will not be enabled. However, once the TESTSIU pin transitions from low to high, the Test Bypass Mode is enabled. A subsequent high to low transition on TESTSIU will disable the Test Bypass Mode.

Once Test Bypass Mode is enabled, the external test logic can initiate a test transaction by asserting /DMAREQ. The DMA Arbitration logic will then assert DMA Request to the CPU core. Once the CPU core responds with DMA Grant, the /DGRNT pin will be asserted. The external test logic then uses other pins that are redefined during Test Bypass Mode to generate TRd and TAddr. TRd indicates whether the test logic wants to read or write the PR31700 registers and TAddr provides the address. These signals are sampled internally after /DGRNT is asserted. A test write will be loaded directly into the Write Buffer when the Write Buffer is empty. A test read will always wait for the Write Buffer to flush before beginning the read cycle. The TAck signal is used to indicate the end of the read or write.

3.4.2 CPU Test Mode

CPU Test Mode is used to isolate the CPU core for testing. This mode is entered by asserting the TESTCPU pin. Once this mode is entered, all signals going in and out of the embedded core are multiplexed to the PR31700 pins. The only PR31700 functionality that remains is the CPU core. All PR31700 initialization and clock enable functions are bypassed so that it appears that only the CPU core exists. This test mode allows the PR31700 IC manufacturers to run their test vectors directly on the isolated CPU core without having to worry about any of the PR31700 specific functionality. This greatly simplifies the process of generating test vectors for the core.

Chapter 4 BIU Module

This section describes the Bus Interface Unit (BIU) logic, which is used to decode the system address space and to interface with memory devices.

4.1 Overview

The BIU provides an interface to the memory devices in the system, including static devices, as well as DRAM and SDRAM devices. An overall block diagram of the BIU is shown in Figure 4-1. The BIU provides control to support the following memory devices:

- 2 Banks of SDRAM and/or DRAM
  - 8-bit or 16-bit SDRAM configuration
  - 16-bit or 32-bit DRAM configuration
  - 4 Mbit, 16 Mbit and 64 Mbit parts supported
  - page mode reads and writes supported
  - independent refresh counters for each bank
  - self refreshing parts supported to retain memory when system is powered down
- 4 general purpose chip selects (/CS3-/CS0)
  - 16-bit or 32-bit ports
  - programmable wait states
  - read page mode
- 4 general purpose chip selects (/MCS3-/MCS0)
  - 16-bit ports
  - programmable wait states
  - read page mode
- 2 full PCMCIA slots
  - 8 or 16-bit ports
  - IORD and IOWR provided to support I/O cards
  - WAIT signal supported

Most of the BIU logic uses the CLK signal (nominally 36.864 MHz) as the clock, but in order to increase the memory bandwidth, SDRAM devices are accessed using the signal DCLKOUT (nominally 73.728 MHz). The DCLKOUT signal is also used to provide more precise timing for generating the control signals for the chip selects, as well as the SDRAM and DRAM control signals. Input data is sampled using the DCLKIN signal which is connected externally to the DCLKOUT signal. The external connection ensures that the input data and DCLKIN will not be skewed, which is necessary to meet the hold time specification on the SDRAM devices. Also, the external looping of DCLKOUT to DCLKIN helps to improve the access time of normal DRAMs, since the input data will be sampled closer to the end of the access.

4.1.1 Block Diagram

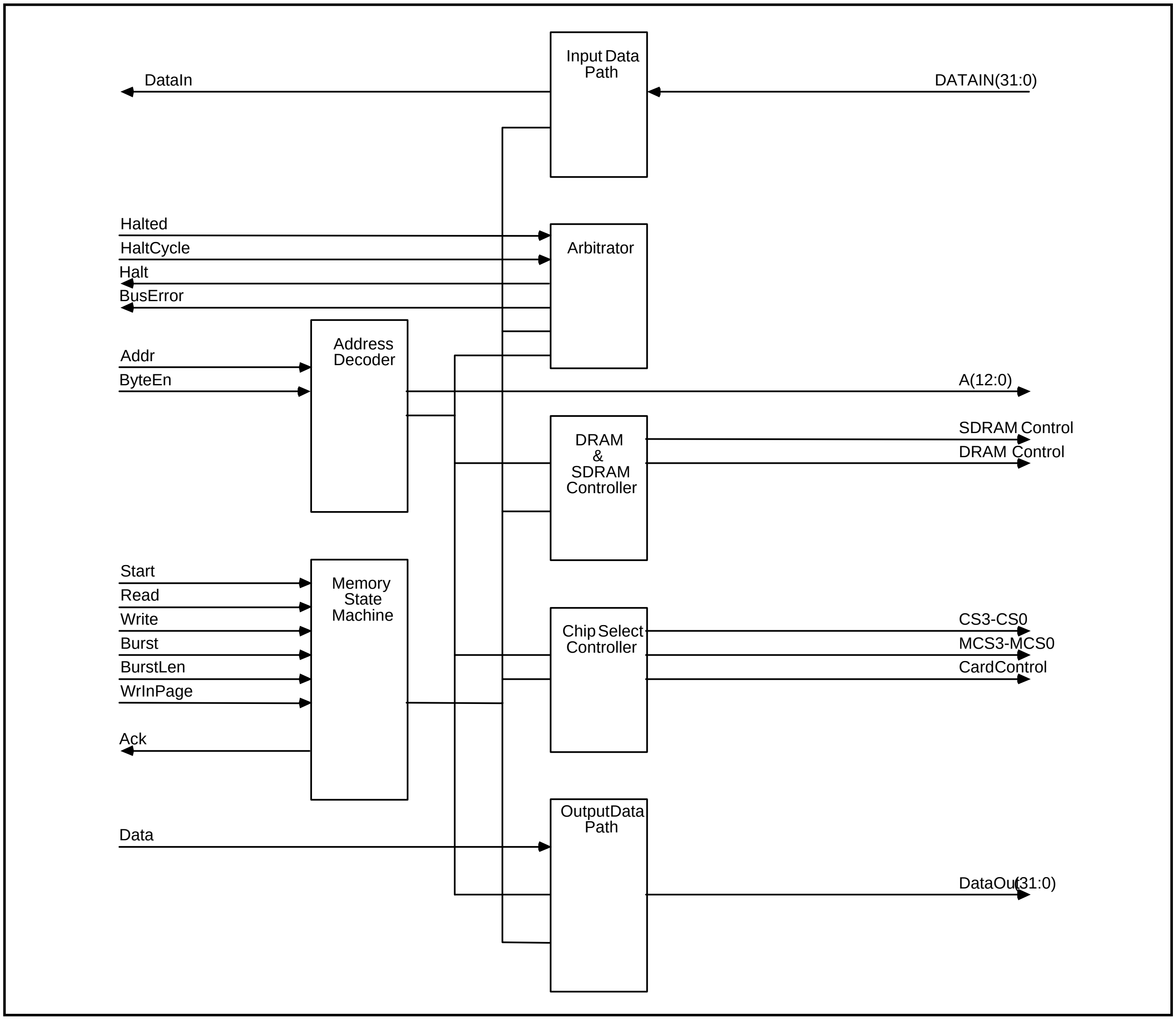


Figure 4-1. BIU Block Diagram

4.2 Address Decoder

The Address Decoder provides the decoding for the address space in the system. The Address Decoder also contains the logic to generate the address signals A(12:0) for the memory devices, along with Address Re-Mapper logic used to re-direct memory addresses.

4.2.1 System Address Map

The System Address Map is shown in Table 4-1. The address space is partitioned to support two banks (Bank 1 and Bank 0) of DRAM and/or SDRAM, four general purpose 16/32-bit chip selects (CS3-CS0), four general purpose 16-bit-only chip selects (MCS3-MCS0), two PCMCIA slots (Card 2 and Card 1), and internal PR31700 register access. The address space contains two primary regions: kseg0/kseg1 are only accessible in kernel mode, while kuser is accessible in either user or kernel mode.

MCS3-MCS0 are available in kuser space and are thus always available to either user or kernel mode accesses. CS3-CS1 are located in both kseg and kuser space, but accesses in kuser space can be independently disabled for each chip select using the ENCS3USER, ENCS2USER, and ENCS1USER control bits in the Memory Configuration 0 Register. CS0 is used as the chip select for the boot ROM in the system. It is mapped starting at address $11000000 in kernel space, which will map into the R3000 boot vector location at address $1FC00000. CS0 is also mapped in kuser space to allow access to the ROM by programs operating in user mode.

Card 2 and Card 1 are mapped into kuser space for memory accesses and kernel-only space for IO and Attribute accesses. To access the attribute space in a card, the CARD2IOEN or CARD1IOEN bits in the Memory Configuration 3 register must be cleared prior to accessing the Card space. For IO accesses the CARD2IOEN or CARD1IOEN bits must be set prior to accessing the Card space.

The processor can write to the Mode Registers in the SDRAMs in kernel mode at address $10F00000 and $10E00000 for Bank 1 and Bank 0 SDRAM banks, respectively. Bank 1 and Bank 0 are each mapped into four different locations. Each location is just an image of the other locations. Each Bank can contain either DRAM or SDRAM devices, with the only restriction being that Bank 1 cannot contain SDRAM if Bank 0 contains DRAM.

A system can be built without DRAM or SDRAM and instead can contain SRAM as the main memory. In this case at least one bank of SRAM should be connected to CS1 and the ENCS1DRAM control bit in the Memory Configuration 0 Register should be set. Setting this bit will cause CS1 to map in place of Bank 1 and Bank 0, as shown in Table 4-1. This is required because exception vectors are mapped into low memory, thus it is required that some memory exist in this region to support exception handling. Accesses to reserved or non-enabled locations will respond with a Bus Error if the Watch Dog Timer is enabled.

Table 4-1. System Address Map

Size	Address	Segment	Devices
16 MByte	FF00_0000	reserved	reserved
1 GByte	C000_0000	kseg2	reserved
1 GByte	7C00_0000	kuser	reserved
64 MByte	7800_0000	kuser	MCS3
64 MByte	7400_0000	kuser	MCS2
64 MByte	7000_0000	kuser	MCS1
64 MByte	6C00_0000	kuser	MCS0
64 MByte	6800_0000	kuser	Card 2 (Memory)
64 MByte	6400_0000	kuser	Card 1 (Memory)
64 MByte	6000_0000	kuser	CS3 (if enabled)
64 MByte	5C00_0000	kuser	CS2 (if enabled)
64 MByte	5800_0000	kuser	CS1 (if enabled)
128 MByte	5000_0000	kuser	CS0 (ROM)
128 MByte	4800_0000	kuser	reserved
32 MByte	4600_0000	kuser	DRAM BANK 1
32 MByte	4400_0000	kuser	DRAM BANK 0
32 MByte	4200_0000	kuser	CS1 (if enabled, else DRAM BANK 0)
32 MByte	4000_0000	kuser	CS1 (if enabled, else DRAM BANK 1)
512 MByte	2000_0000	reserved	reserved
240 MByte	1100_0000	kseg0,kseg1	CS0 (ROM)
1 MByte	10F0_0000	kseg0,kseg1	SDRAM BANK 1 Mode Register
1 MByte	10E0_0000	kseg0,kseg1	SDRAM BANK 0 Mode Register
2 MByte	10C0_0000	kseg0,kseg1	PR31700 Registers
4 MByte	1080_0000	kseg0,kseg1	CS3
4MByte	1040_0000	kseg0,kseg1	CS2
4MByte	1000_0000	kseg0,kseg1	CS1
64MByte	0C00_0000	kseg0,kseg1	Card 2 (I/O or Attribute)
64MByte	0800_0000	kseg0,kseg1	Card 1 (I/O or Attribute)
32MByte	0600_0000	kseg0,kseg1	DRAM BANK 1
32MByte	0400_0000	kseg0,kseg1	DRAM BANK 0
32MByte	0200_0000	kseg0,kseg1	CS1 (if enabled, else DRAM BANK 1)
32MByte	0000_0000	kseg0,kseg1	CS1 (if enabled, else DRAM BANK 0)

4.2.2 Address Generation

The Address Decoder logic generates the A(12:0) address bus for the memory devices. Addresses are multiplexed on the A(12:0) address bus. For static devices such as CS3-CS0, MCS3-MCS0, Card 2 and Card 1, the addresses are multiplexed such that Addr(25:13) is provided first-in-time followed by Addr(12:0), as shown in Figure 4-2. Addr(25:13) must be latched with an external latch using the ALE pin. The latched addresses concatenated with A(12:0) will provide a 26-bit address bus for the system, HA(25:0).

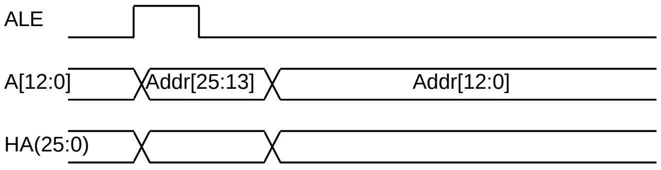


Figure 4-2. ALE Timing

DRAM and SDRAM devices contain multiplexed address buses. Thus, A(12:0) will be directly connected to the DRAM and SDRAM devices. In order to support a wide range of DRAM and SDRAM types and configurations, there are control bits in the Memory Configuration 0 Register that provide soft configuration for the Row and Column address bit positions. These control bits are ROWSEL1(1:0) and COLSEL1(2:0) for Bank 1, and ROWSEL0(1:0) and COLSEL0(2:0) for Bank 0. Table 4-2 shows the position that specific address bits are placed, depending on the configuration of the control bits. This flexibility in address bit positioning makes it possible to support 4-Mbit, 16-Mbit, and 64-Mbit and 256-Mbit DRAM and SDRAM devices with different bus width configurations. The two sets of control bits make it possible to support different devices in Bank 1 and Bank 0. The "Ø" character in the table means that this bit position will be a zero and the "X" implies a "don't care" bit position.

Table 4-2. DRAM and SDRAM Address Mapping

ROW A(12:0)	ROWSEL	COL A(12:0)	COLSEL
18,17:9	00	22,20,18,8:1	0000
22,18,20,19,17:9	01	19,18,8:2	0001
20,22,21,19,17:9	10	21,20,18,8:2	0010
22,23,21,19,17:9	11	23,22,20,18,8:2	0011
		24,22,20,18,8:2	0100
		18,Ø,X,8:0	0101
		22,Ø,X,21,8:0	0110
		18,Ø,X,21,8:1	0111
		22,Ø,X,23,21,8:1	1000
		24,23,21,8:2	1001

4.2.3 Address Re-Mapper Logic

The Address Decoder logic provides an Address Re-Mapper that allows addresses to be re-directed. There are two separate sets of Re-Mapper registers provided. The Memory Configuration 5-8 Registers contain the STARTVAL2(31:9), MASK2(3:0) and RMAPADD2(31:9) control bits for one Re-Mapper, and the STARTVAL1(31:9), MASK1(3:0) and RMAPADD1(31:9) control bits for the other Re-Mapper. Each Re-Mapper contains it's own enable bit defined by the ENRMAP2 and ENRMAP1 control bits in the Memory Configuration 0 Register. If the enable bits are not set, then no address Re-Mapping will occur. The Re-Mapper works as described in the following paragraph.

The Addr(31:2) address output of the CPU Interface is compared with the STARTVAL(31:9) bits. If the addresses compare, the upper address bits are replaced with the RMAPADD(31:9) bits. The lower bits (8:2) are always passed through. This provides a 512-byte block that can be re-mapped to any other 512-byte block. The MASK(3:0) bits are used to select either a 512, 1K, 2K, 4K or 8K byte block. A bitwise "AND" of the MASK(3:0) bits with Addr(12:9) is performed before the address comparison. If the address compares to the STARTVAL bits then the address is replaced with the RMAPADD bits. The resulting address is then provided to the Address Decoder logic. Table 4-3 shows the resulting address when the comparison is true for the different possible MASK settings. The STARTVAL(12:9) bits are valid as shown by the "V" in the table. Otherwise the bits must be set to zero as shown.

Table 4-3. Address Re-Mapper

MASK(3:0)	STARTVAL(12:9)	Address Result
F	V,V,V,V	RMAPADD(31:9),Addr(8:2)
E	V,V,V,0	RMAPADD(31:10),Addr(9:2)
C	V,V,0,0	RMAPADD(31:11),Addr(10:2)
8	V,0,0,0	RMAPADD(31:12),Addr(11:2)
0	0,0,0,0	RMAPADD(31:13),Addr(12:2)

The primary function of the Re-Mapper logic is to support Flash devices, which have slow write access times, are block-oriented, and require a lot of overhead. The re-mapping of Flash writes into normal memory space allows the Flash to be written much less frequently. Anytime a write occurs within the re-mapped range, the write will instead take place into the desired re-mapped memory location. The software can then setup memory protection to protect writes to the rest of the Flash, outside of the previously selected range or block. Once a memory protection exception occurs, the software can disable the Re-Mapper, move the data from normal memory to the Flash, then move the STARTVAL bits to point to a new address, corresponding to a new address block.

4.3 Chip Select Controller

4.3.1 Description

The Chip Select Controller provides the logic required to generate CS3-CS0, MCS3-MCS0, Card 2, and Card 1 control signals. CS3-CS0 can each independently be set as either 32-bit or 16-bit ports. MCS3-MCS0, Card 2, and Card 1 are always 16-bit ports.

Reads from a 32-bit port will result in a single access with the appropriate chip select being asserted along with the /RD signal. Writes to a 32-bit port will result in a single access with the appropriate chip select being asserted along with the /WE signal, as well as /CAS3-/CAS0 being asserted to select the byte to be written. The ByteEn signals from the CPU Interface define the desired byte lane for memory accesses. A Word access will read or write all 32 bits, a Tri access will read or write either the upper 24 bits or lower 24 bits, a Half-Word access will read or write either the upper 16 bits or lower 16 bits, and a Byte access will read or write any of the 4 bytes.

For writes to CS3-CS0 when the ports are configured as 32 bits, the /CAS3-CAS0 signals are used as individual write enables for each byte, with /CAS3 used for the upper byte, /CAS2 used for the next byte, etc. Word or Tri accesses to 16-bit ports will be split into two consecutive accesses. HA(1) is used to select between the upper or lower 16 bits to be accessed. The /CAS3 and /CAS2 signals are used as write enables for the byte lanes for accesses to 16-bit ports. Accesses to Card 2 or Card 1 do not use /CAS3 and /CAS2, since the card already contains an upper and lower chip select. All 16-bit port memory devices should always be connected to D(31:16).

The access time for each chip select is individually programmed using the control bits in the Memory Configuration 1-3 Registers. Also provided for CS3-CS0 and MCS3-MCS0 is separate access time control for Read Page Mode that can be enabled or disabled independently for each chip select using the ENxxPAGE control bits in the Memory Configuration 3 Register. The Read Page Mode provides burst read capability to support memory devices such as page mode ROMs. There is separate access time control provided for memory accesses to Card 2 or Card 1, or IO/Attribute accesses to Card 2 or Card 1. The address inputs for all 32-bit memory devices should be connected starting with HA(2) and all 16-bit memory devices should be connected starting with HA(1). The PCMCIA cards will use all the address bits HA(25:0).

4.3.2 Access Mapping

Tables 4-4 and 4-5 show how write and read accesses are presented to the memory devices. For 32-bit ports there is only one access required, but Word and Tri accesses to 16-bit ports must be split into two accesses.

Table 4-4. Chip Select Write Map

Access	ByteEn	HA(1:0)	Port Size	D(31:24)	D(23:16)	D(15:8)	D(7:0)
Word	1111	00	32-bit	Data(31:24)	Data(23:16)	Data(15:8)	Data(7:0)
Word	1111	00	16-bit	Data(31:24)	Data(23:16)	X	X
Word	1111	10	16-bit	Data(15:8)	Data(7:0)	X	X
Tri	1110	00	32-bit	Data(31:24)	Data(23:16)	Data(15:8)	X
Tri	1110	00	16-bit	Data(31:24)	Data(23:16)	X	X
Tri	1110	10	16-bit	Data(15:8)	X	X	X
Tri	0111	01	32-bit	X	Data(23:16)	Data(15:8)	Data(7:0)
Tri	0111	01	16-bit	X	Data(23:16)	X	X
Tri	0111	10	16-bit	Data(15:8)	Data(7:0)	X	X
Half-Word	1100	00	32-bit	Data(31:24)	Data(23:16)	X	X
Half-Word	1100	00	16-bit	Data(31:24)	Data(23:16)	X	X
Half-Word	0011	10	32-bit	X	X	Data(15:8)	Data(7:0)
Half-Word	0011	10	16-bit	Data(15:8)	Data(7:0)	X	X
Byte	1000	00	32-bit	Data(31:24)	X	X	X
Byte	1000	00	16-bit	Data(31:24)	X	X	X
Byte	0100	01	32-bit	X	Data(23:16)	X	X
Byte	0100	01	16-bit	X	Data(23:16)	X	X
Byte	0010	10	32-bit	X	X	Data(15:8)	X
Byte	0010	10	16-bit	Data(15:8)	X	X	X
Byte	0001	11	32-bit	X	X	X	Data(7:0)
Byte	0001	11	16-bit	X	Data(7:0)	X	X

Table 4-5. Chip Select Read Map

Access	ByteEn	HA(1:0)	Port Size	DataIn(31:24)	DataIn(23:16)	DataIn(15:8)	DataIn(7:0)
Word	1111	00	32-bit	D(31:24)	D(23:16)	D(15:8)	D(7:0)
Word	1111	00	16-bit	D(31:24)	D(23:16)	X	X
Word	1111	10	16-bit	X	X	D(31:24)	D(23:16)
Tri	1110	00	32-bit	D(31:24)	D(23:16)	D(15:8)	X
Tri	1110	00	16-bit	D(31:24)	D(23:16)	X	X
Tri	1110	10	16-bit	X	X	D(31:24)	X
Tri	0111	01	32-bit	X	D(23:16)	D(15:8)	D(7:0)
Tri	0111	01	16-bit	X	D(23:16)	X	X
Tri	0111	10	16-bit	X	X	D(31:24)	D(23:16)
Half-Word	1100	00	32-bit	D(31:24)	D(23:16)	X	X
Half-Word	1100	00	16-bit	D(31:24)	D(23:16)	X	X
Half-Word	0011	10	32-bit	X	X	D(15:8)	D(7:0)
Half-Word	0011	10	16-bit	X	X	D(31:24)	D(23:16)
Byte	1000	00	32-bit	D(31:24)	X	X	X
Byte	1000	00	16-bit	D(31:24)	X	X	X
Byte	0100	01	32-bit	X	D(23:16)	X	X
Byte	0100	01	16-bit	X	D(23:16)	X	X
Byte	0010	10	32-bit	X	X	D(15:8)	X
Byte	0010	10	16-bit	X	X	D(31:24)	X
Byte	0001	11	32-bit	X	X	X	D(7:0)
Byte	0001	11	16-bit	X	X	X	D(23:16)

4.3.3 CS3-CS0 and MCS3-MCS0 Timing

Figures 4-3 through 4-6 show the timing for CS3-CS0 and MCS3-MCS0. The timing is the same for all these signals, with the only exception being that 32-bit port timing only applies to CS3-CS0 while 16-bit port timing applies to CS3-CS0 and MCS3-MCS0.

Timing is also shown for Read Page Mode accesses. If Read Page Mode is enabled, the chip select will be asserted for the entire read cycle. If the processor is performing a burst read to re-fill either a data or instruction cache line, the chip select will remain asserted for the entire burst read. The first access of a Read Page Mode access uses the ACCVAL1 control bits to define the access time, while the next three accesses will use the ACCVAL2 control bits to define the access time. The pattern will repeat until the read is complete.

Figure 4-3 shows the timing for a single Word read and write for a 32-bit memory device. The access time is defined using the ACCVAL1 control bits. There are separate ACCVAL1 control bits for each of the chip selects, such that each chip select has an independently configurable access time. The ACCVAL1 bits should be set to one less than the desired number of CLK times for the chip select.

Figure 4-4 shows the timing for a Word read from a 16-bit port. The access is split into two parts to read the full 32 bits. During the first read HA(1) will be low and during the second read HA(1) will be high. The input data path will latch both accesses and concatenate them together as 32 bits for the processor before asserting the Ack signal to the CPU Interface logic.

Figure 4-5 shows a 4-Word burst read from a 32-bit port. The ALE signal is only asserted at the beginning of the cycle since the upper address bits will not be changed during the burst. The burst will always start on a 16-byte boundary. The address bits will increment as follows: HA(3:2) = 00, HA(3:2) = 01, HA(3:2) = 10, then HA(3:2) = 11.

Figure 4-6 shows the same 4-Word burst read from a 32-bit port, except this time Read Page Mode is enabled. In this case the chip select will remain asserted for the entire access and the addresses will change to access the new address location. The first access uses the ACCVAL1 control bits, while the next three accesses use the ACCVAL2 control bits. If the burst had been longer then 4 Words, the ACCVAL1 control bits would have been used for every fourth access. This type of bursting is compatible with ROM devices that support page mode access. If the 4-Word burst would have been from a 16-bit port, eight accesses would have been required, but the ACCVAL1 bits would still be used for the first and fifth access, while the ACCVAL2 bits would have been used for the rest. If Read Page Mode is enabled for a particular chip select, then the ACCVAL1 and ACCVAL2 control bits must not be set to $0. The "up arrow" markings on DCLKOUT in Figures 4-3 through 4-6 show the points where input data is sampled.

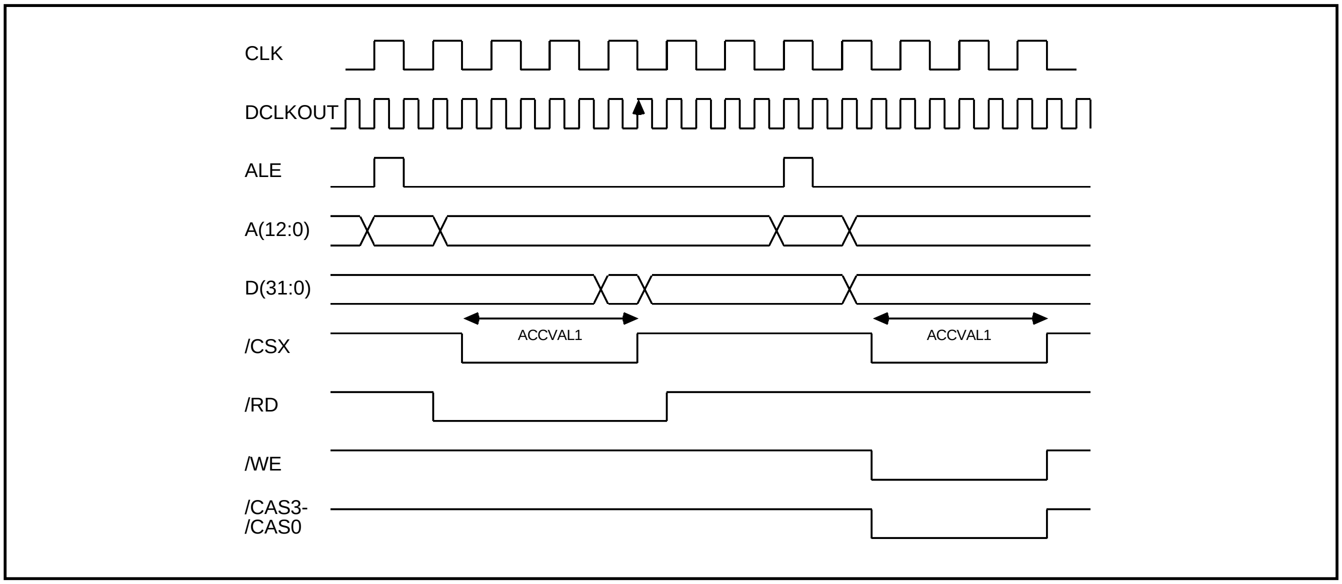


Figure 4-3. Word Read/Write 32-Bit Port

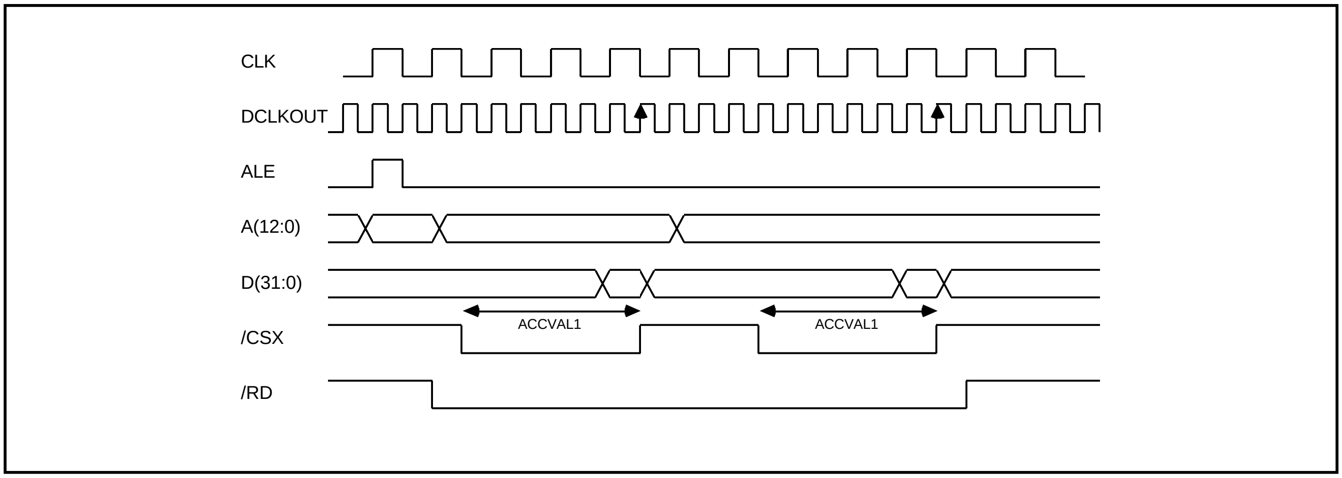


Figure 4-4. Word Read 16-Bit Port

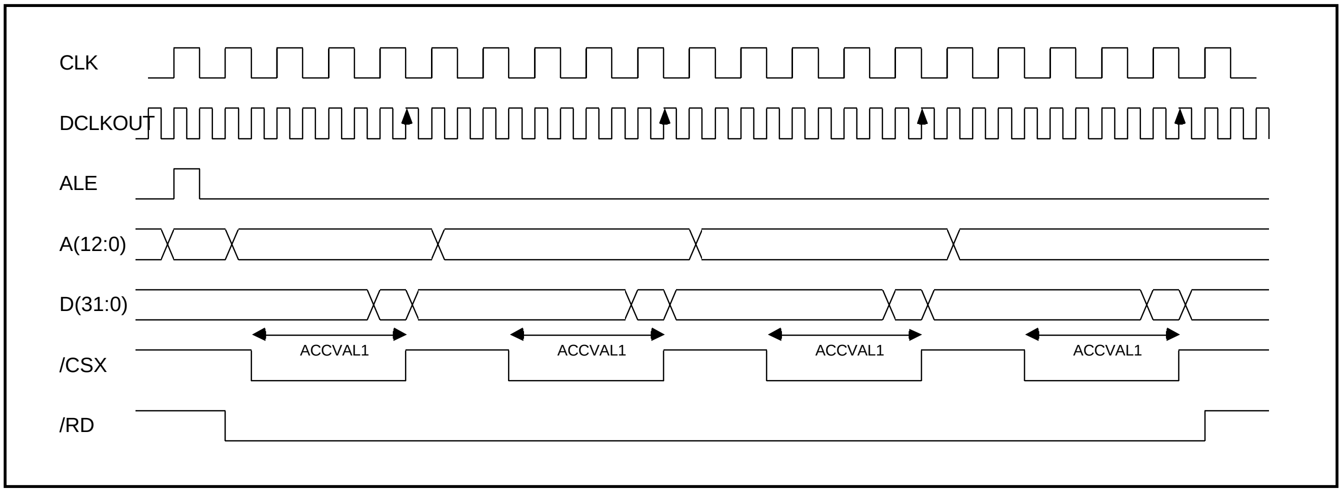


Figure 4-5. 4-Word Burst Read 32-Bit Port

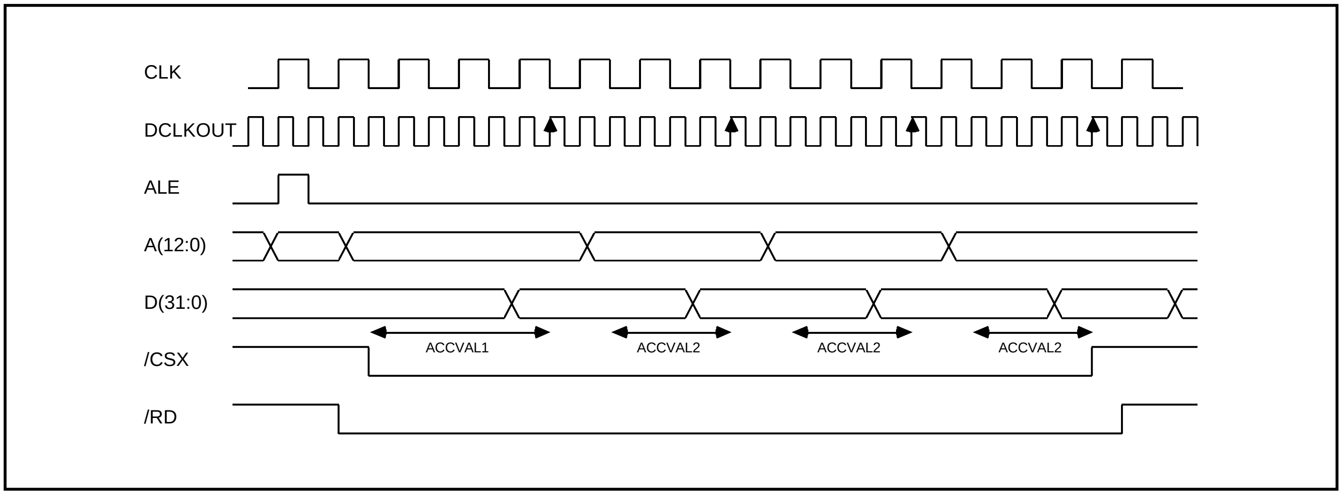


Figure 4-6. 4-Word Burst Read 32-Bit Port Read with Page Mode Enabled

4.3.4 Card 2 and Card 1

The chip selects for Card 2 and Card 1 are designed to comply with Revision 2.01 of the PCMCIA specification. Table 4-6 shows the byte lanes on the PC-card. Modification have been made to the PR31700 to enable 8-bit port access and the control bit is added to Memory Configuration 3 Register (offset = $00C) to select 8-bit PC-card port size. To support this mapping it is required that CD(7:0) be connected to D(31:24) and CD(15:8) be connected to D(23:16). This swap is required because the PCMCIA card is mapped as Little Endian while the PR31700 is mapped as Big Endian.

Table 4-6. PCMCIA Byte Lanes

	/CARDxCSH	/CARDxCSL	HA(0)	Card CD(15:8)	Card CD(7:0)
Standby	1	1	X	High-Z	High-Z
Byte	1	0	0	High-Z	Even-Byte
Byte (8-bit)	1	0	1	High-Z	Odd-Byte
Byte (16-bit)	0	1	X	Odd-byte	High-Z
Word	0	0	X	Odd-Byte	Even-Byte

There are three different types of accesses to the Card: Memory, Attribute and I/O. Memory space is accessed by reading or writing from or to the Card Memory locations. Attribute or IO space is accessed by reading or writing from or to the Card IO/Attribute locations. If an Attribute access is desired, the CARDxIOEN bit should be cleared. If an IO access is desired, the CARDxIOEN bit should be set.

To support bi-directional buffering of the data bus to the Cards, the /CARDDIR signal provides direction control for the bi-directional buffer. The /CARDDIR signal will be asserted during a read to either Card 2 or Card 1.

There is separate access time control provided for Memory access versus IO/Attribute access. CARDxACCVAL is used to control the access time for memory accesses and CARDxIOACCVAL is used to control the access time for IO or Attribute access. Adding one to either access value control setting will increase the access time by two CLK periods. A value of $0 will result in an access time of 1 CLK period. Thus, the formula for the access time for the card is: Access Time = (ACCVAL * 2 + 1) CLK periods. Figures 4-7 through 4-9 show the different timing for Memory, Attribute and IO accesses. If a Word access is made to the Card, the access will be split into two accesses in the same manner as is done for CS3-CS0 and MCS3-MCS0. Read Page Mode is not supported for the Cards since it is not supported by the PCMCIA specification.

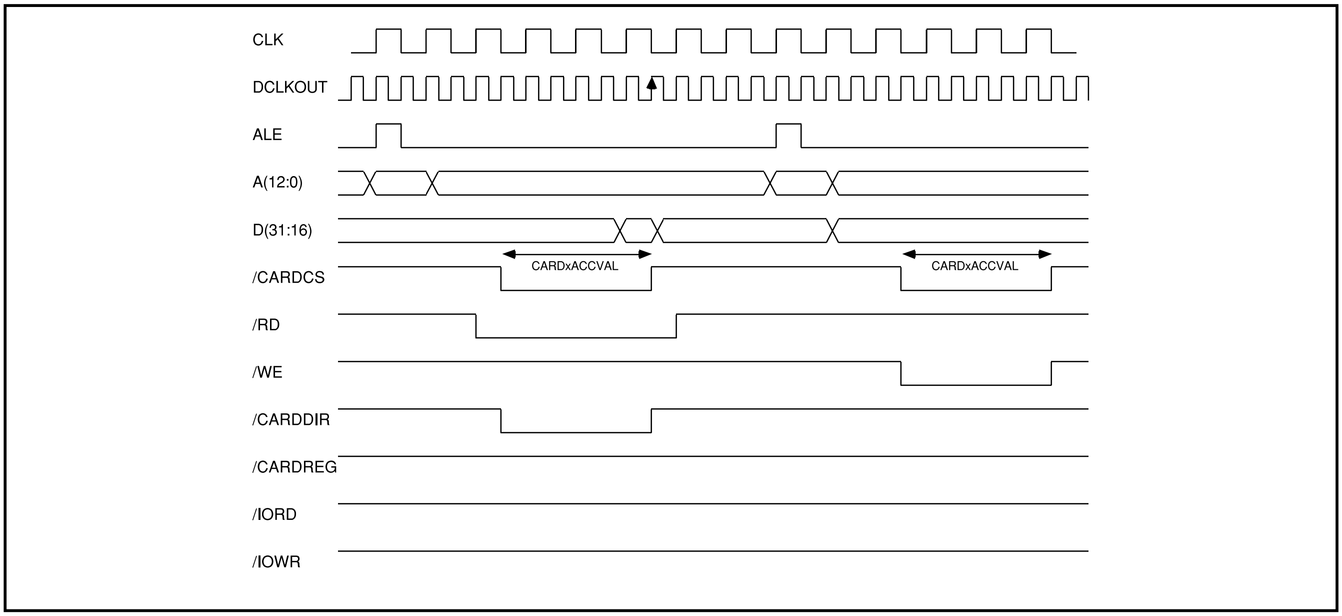


Figure 4-7. Memory Access to Card

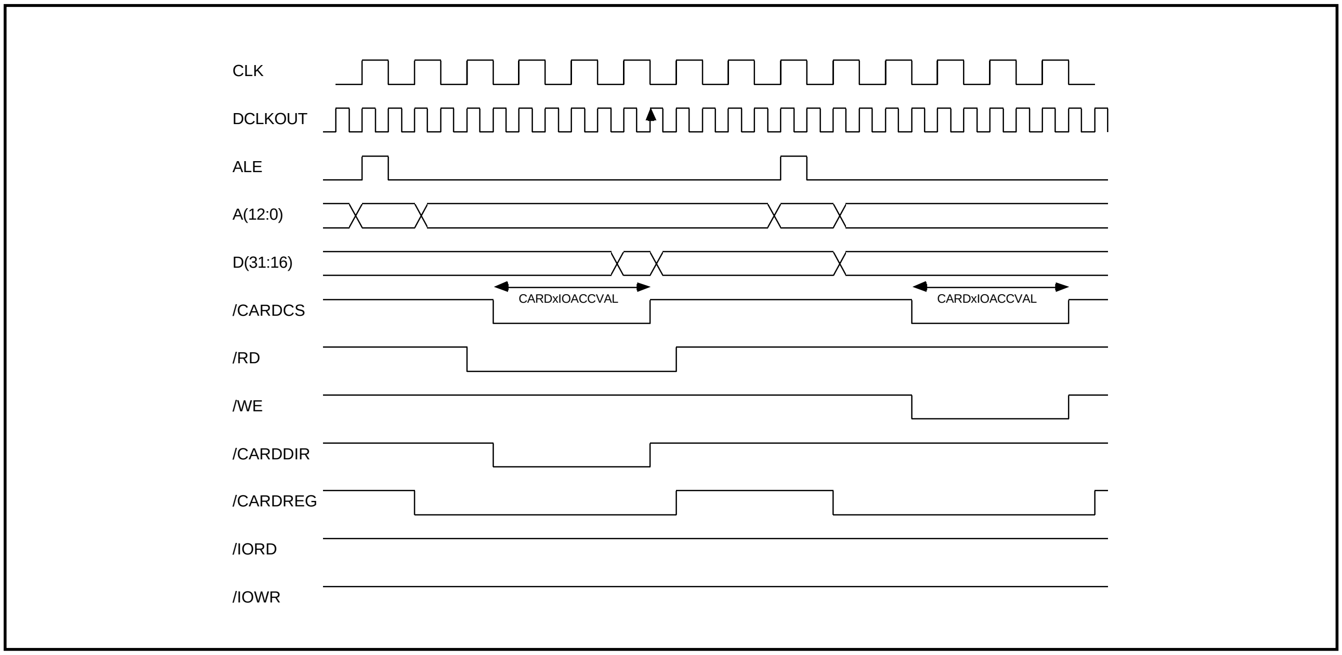


Figure 4-8. Attribute Access to Card

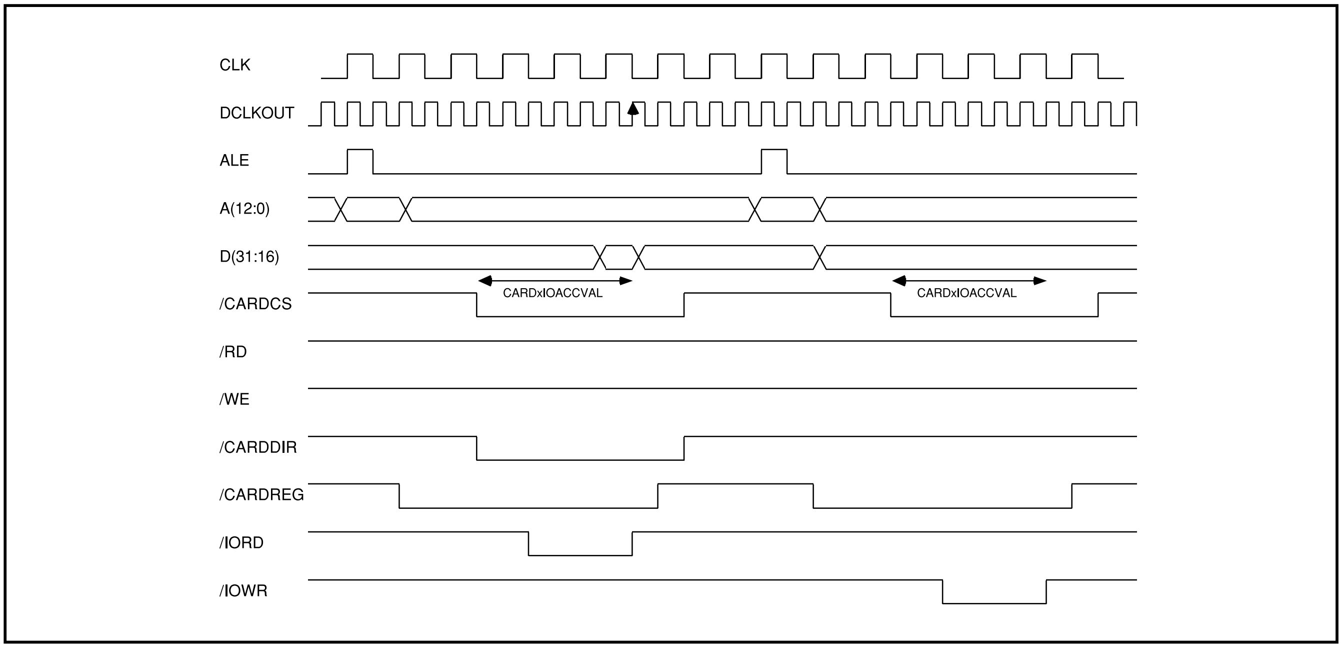


Figure 4-9. IO Access to Card

The PCMCIA specification provides for a /WAIT signal that allows the Card to halt the bus cycle until the /WAIT signal is de-asserted. This allows the Card to control the access time. The /WAIT signal will assert at the beginning of the cycle and the cycle should not terminate until the /WAIT signal becomes de-asserted. There are separate /WAIT signals provided for each Card: /WAIT2 and /WAIT1. The /WAIT signals are sampled on two consecutive rising edges of CLK. If the /WAIT signal is low, the access time counter will freeze until the /WAIT signal goes high. This function will only occur if the WAIT function is enabled using the CARDxWAITEN control bit. If the WAIT function is not enabled, the /WAIT signal is ignored. Figure 4-10 shows the timing for Card WAIT accesses.

The ACCVAL for the card must be set to a minimum value such that the cycle will not end before the /WAIT signal has been sampled for two clock periods. Support for the /WAIT signal has been included to comply with the PCMCIA specification. Support of the /WAIT signal should be used cautiously depending on the maximum /WAIT time for a particular card and the other demands of the system.

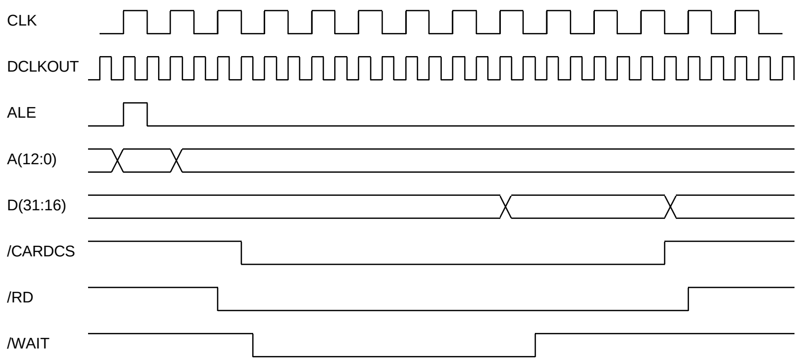


Figure 4-10. Card Wait Timing

4.3.5 CS0 Size Configuration

In order for the system to boot properly when powered up for the first time, it is necessary that CS0 and it's associated control be initialized properly since this chip select is used for the Boot ROM. When RESET is asserted, the ACCVAL1 and ACCVAL2 control bits for CS0 will be set to their maximum value and Read Page Mode will be disabled. The port size, 16-bit or 32-bit, is configured using the TESTSIU input pin. If the TESTSIU pin is high during RESET, CS0 is configured as a 32-bit port, and if TESTSIU is low during RESET, CS0 is configured as a 16-bit port. The CPU Module Section discusses the TESTSIU pin in more detail.

4.3.6 Show PR31700

For debugging purposes, a mode called "SHOWPR31700" will cause internal accesses between the CPU and the PR31700 registers to show externally. This mode is enabled by asserting the SHOWPR31700 control bit in the Memory Configuration 0 Register. When this bit is set, a write or read to or from internal PR31700 registers will be seen externally on the D(31:0) data bus and A(12:0) address bus. As shown in Figure 4-11, the /CARDREG signal is used as a strobe signal for latching the data and address, and the /RD signal indicates if the processor is performing a write or a read.

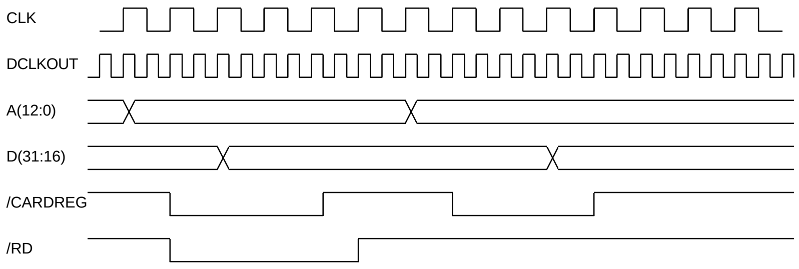


Figure 4-11. SHOWPR31700 Timing

4.4 DRAM and SDRAM Controller

The DRAM and SDRAM Controller generates the required control signals to interface to DRAM and SDRAM devices. Two separate banks are supported and each bank can contain either DRAM or SDRAM, with the only restriction being that Bank 1 cannot contain SDRAM if Bank 0 contains DRAM. (SDRAM in Bank 0 while DRAM in Bank 1 would be ok) The DRAM and SDRAM Controller supports 4-Mbit, 16-Mbit, 64-Mbit and 256-Mbit devices in various bus configurations. In order to maximize the memory bandwidth, both the DRAM and SDRAM devices can be kept in page mode to complete burst read cycles and in-page write cycles.

4.4.1 Memory Chips Supported

Table 4-7 shows the different DRAM and SDRAM chips that are supported by the DRAM and SDRAM Controller. Also shown is the Row and Column configuration for the supported chips. SDRAM chips also contain two banks that are selectable through the address space.

Table 4-7. Supported DRAM and SDRAM Chips

Bank	ROW	COL	Memory Size	Memory Type
	9	9	256K X 16	4-Mbit DRAM
	10	8	256K X 16	4-Mbit DRAM
	10	9	512K X 8	4-Mbit DRAM
	10	10	1M X 16	16-Mbit DRAM
	11	9	1M X 16	16-Mbit DRAM
	12	8	1M X 16	16-Mbit DRAM
	11	10	2M X 8	16-Mbit DRAM
	12	9	2M X 8	16-Mbit DRAM
1	11	9	2M X 8	16-Mbit SDRAM
1	11	8	1M X 16	16-Mbit SDRAM
	11	11	4M X 16	64-Mbit DRAM
	12	10	4M X 16	64-Mbit DRAM
	12	11	8M X 8	64-Mbit DRAM
1	12	10	8M X 8	64-Mbit SDRAM
1	12	9	4M X 16	64-Mbit SDRAM
	12	11	16M X 16	256-Mbit DRAM
	13	10	16M X 16	256-Mbit DRAM

4.4.2 DRAM and SDRAM Configurations

Table 4-8 shows the various combinations of DRAM and SDRAM chips that are supported. Bank 1 and Bank 0 are independently programmable so each can contain any of the combinations. The only restriction is that Bank 1 cannot contain SDRAM if Bank 0 contains DRAM.

Table 4-8. Supported DRAM and SDRAM Configurations

ROW	COL	Memory Size	Memory Type	# of Chips
18,17:9	18,8:1	256K X 16	4-Mbit DRAM	1 X 256K X 16
20,19,17:9	20,18,8:1	1M X 16	16-Mbit DRAM	1 X 1M X 16
18,20,19,17:9	8:1	1M X 16	16-Mbit DRAM	1 X 1M X 16
20,22,21,19,17:9	22,20,18,8:1	4M X 16	64-Mbit DRAM	1 X 4M X 16
19,17:9	19,18,8:2	256K X 32	4-Mbit DRAM	2 X 256K X 16
19,17:9	20,18,8:2	512K X 32	4-Mbit DRAM	4 X 512 X 8
21,19,17:9	21,20,18,8:2	1M X 32	16-Mbit DRAM	2 X 1M X 16
18,20,19,17:9	24,23,21,8:2	1M X 32	16-Mbit DRAM	2 X 1M X 16
20,22,21,19,17:9	22,20,18,8:2	2M X 32	16-Mbit DRAM	4 X 2M X 8
22,23,21,19,17:9	23,22,20,18,8:2	4M X 32	64-Mbit DRAM	2 X 4M X 16
22,23,21,19,17:9	24,22,20,18,8:2	8M X 32	64-Mbit DRAM	4 X 8M X 8
22,23,21,19,17:9	22,20,18,8:1	16M X 16	256-Mbit DRAM	1 X 16M X 16
18,20,19,17:9	18,Ø,X ,8:0	2M X 8	16-Mbit SDRAM	1 X 2M X 8
22,18,20,19,17:9	22,Ø,X,21,8:0	8M X 8	64-Mbit SDRAM	1 X 8M X 8
18,20,19,17:9	18,Ø,X,X,8:1	1M X 16	16-Mbit SDRAM	1 X 1M X 16
18,20,19,17:9	18,Ø,X,21,8:1	2M X 16	16-Mbit SDRAM	2 X 2M X 8
22,18,20,19,17:9	22,Ø,X,X,21,8:1	4M X 16	64-Mbit SDRAM	1 X 4M X 16
22,18,20,19,17:9	22,Ø,X,23,21,8:1	8M X 16	64-Mbit SDRAM	2 X 8M X 8

4.4.3 DRAM

DRAM in Bank 1 is controlled using /RAS1 and /CAS3-/CAS0, while DRAM in Bank 0 is controlled using /RAS0 and /CAS3-/CAS0. The Memory Configuration 0 Register provides control bits that allow Bank 1 or Bank 0 to be independently set as either 32-bit or 16-bit DRAM configurations. The 32-bit DRAM configurations are connected to all 32 data bus bits D(31:0), while 16-bit DRAM configurations are connected to D(15:0). The 16-bit DRAM configurations will only use /CAS1 and /CAS0 for control, while 32-bit configurations will use all four /CAS lines. Word and Tri accesses to 16-bit configured DRAMs will be split into two accesses in the same manner as was done for the chip selects, except that the data is connected to D(15:0) instead of D(31:16). Connecting DRAM devices to the lower data bus and chip select memory devices to the upper data bus helps to balance the capacitive loading on the data bus.

The DRAM timing is fixed to support 80 ns devices or faster. The only programmability is the ENBANK1OPT and ENBANK0OPT control bits in the Memory Configuration 4 Register that allow an extra clock of time for the first access to the DRAM. This option is included to support 80 ns DRAM devices with a 36.864 MHz clock for the CPU and BIU. DRAMs faster than 80 ns will not need the ENBANKxOPT bit to be set.

During a burst read cycle the /RASx line will be asserted to latch the ROW address into the DRAM, and the /CAS signals are then strobed for the required number of accesses. Page mode writes are supported when the ENWRINPAGE control bit is set in the Memory Configuration 0 Register. An in-page write will occur when the CPU Interface asserts the WrInPage signal during a write. If this signal is asserted, the DRAM will be kept in page mode until the WrInPage signal is no longer asserted.

Refresh is inserted using /CAS before /RAS refresh. The Refresh Control is generated by the Arbitration Logic. Figures 4-12 through 4-14 show various timing examples, with refresh timing shown in Figure 4-14. The timing is the same for 32-bit and 16-bit ports, except there are twice as many accesses to the 16-bit port. For example, a Word read from a 16-bit port will result in a burst of two 16-bit reads to fetch the 32-bits of data.

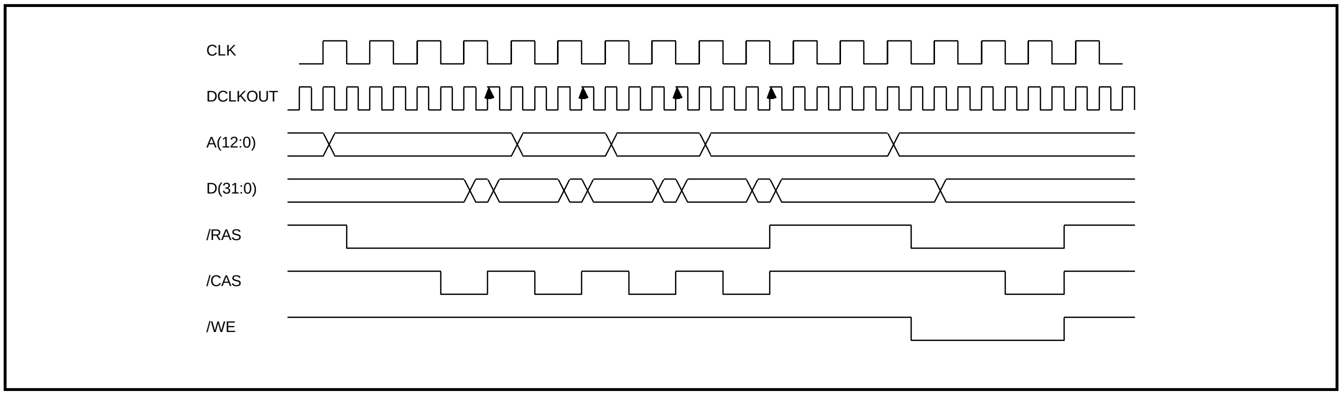


Figure 4-12. Burst Read followed by Word Write

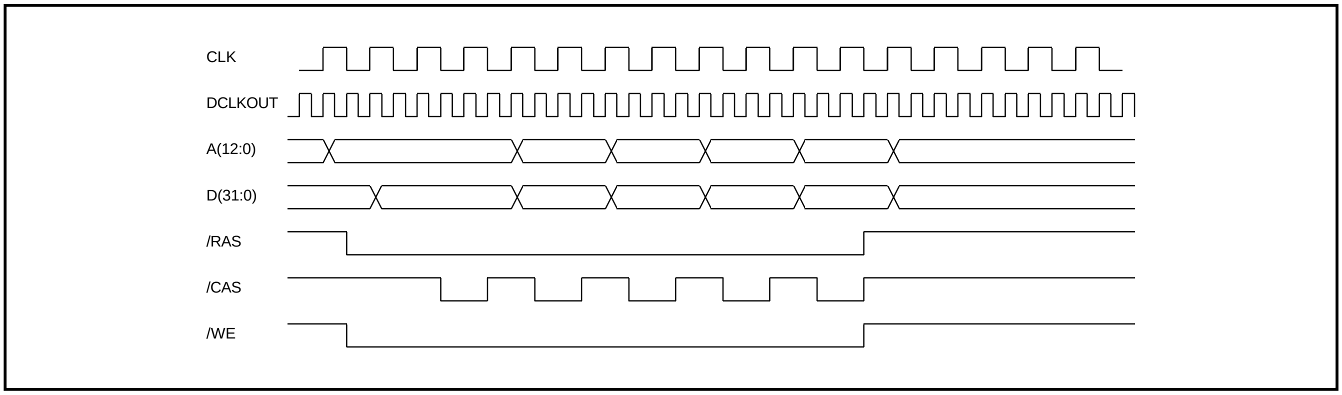


Figure 4-13. In-Page Write

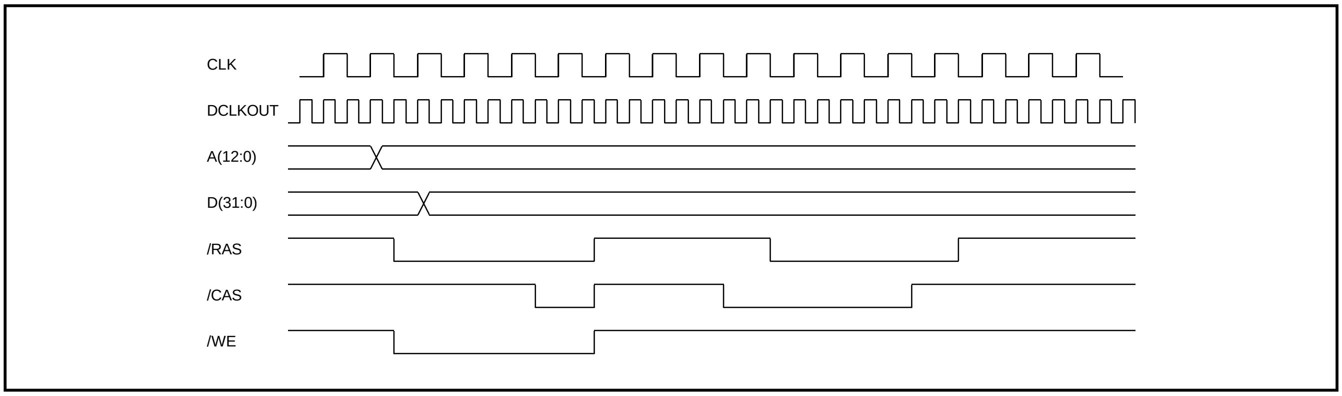


Figure 4-14. Write with ENBANKOPT Set followed by Refresh Cycle

4.4.4 DRAM Initialization

When the DRAM is first powered up there are several things that must be done before accessing the DRAM. First the Memory Configuration Registers must be properly configured for the chosen DRAM and bus configuration. Next, after a delay of at least 100 s after the power is asserted to the DRAM, the refresh should be enabled. The DRAM should not be accessed until at least eight /CAS before /RAS refresh cycles occur. After this procedure is completed the DRAM can be accessed.

4.4.5 SDRAM

SDRAM devices are supported in Bank 1 and Bank 0. Each Bank can independently be configured to support either 16-bit or 8-bit SDRAMs. Bank 1 uses /RAS1 as the chip select for the SDRAM and Bank 0 uses /DCS0. All other control signals are shared between the two banks.

The clock for the SDRAM devices is the DCLKOUT signal which nominally runs at 73.728 MHz. This clock is twice the rate of the CLK signal (nominally 36.864 MHz). Reads from the SDRAMs occur on every DCLKOUT cycle, such that the reading of all 32 bits for 16-bit SDRAMs can happen within a single CLK cycle. A 16-bit SDRAM configuration provides the maximum memory bandwidth and once a read cycle begins, the CPU will not stall since 32 bits are read on each cycle. All 8-bit SDRAMs require 2 CLK cycles to read 32 bits of data. During a burst read the SDRAM devices are kept in page mode for the entire read cycle.

The SDRAM devices are also kept in page mode during in-page write cycles. All 16-bit SDRAMs can keep up at 32 bits per CLK cycle, which can provide a long write block without stalling the CPU. All 8-bit SDRAMs require twice the number of cycles. Tri, Half-Word and Byte writes are supported to the SDRAM devices using the DQMH and DQML signals. These signals provide a data mask so that only the desired bytes will be written. DQMH is used for the upper 8 bits in a 16-bit SDRAM configuration and DQML is used for the lower 8 bits. For 8-bit SDRAMs only DQML is used. The data mask signals DQMH and DQML are required because the SDRAM is setup to always read or write 32 bits of data at a time. The data mask signals will "mask out" the bytes that should not be written during partial Word stores.

4.4.6 SDRAM Initialization

Prior to accessing an SDRAM device, an initialization procedure must be followed. First, all Memory Configuration Registers must be properly initialized for the chosen SDRAM and bus configuration. Next, after a pause of at least 100 s after power is applied to the SDRAM, a command must be sent to precharge the SDRAM banks. The command is sent by writing to the SDRAM Bank 1 and/or Bank 0 Mode Register address with Data(31) set high. Setting this bit will cause a Precharge All command to be sent to the SDRAM instead of writing to the Mode Register. Next, the Mode Register must be properly initialized. Finally, refresh should be enabled and at least 2 refresh cycles must occur prior to reading or writing from or to the SDRAM.

4.4.7 SDRAM Mode Register

An SDRAM contains a write-only Mode Register that defines the parameters for accessing the device. The mode register must be initialized prior to reading or writing from the SDRAM. They can be written via the SDRAM Bank 1 and SDRAM Bank 0 addresses as shown in the System Address Map. When writing to the mode register, the data from the CPU Interface on Data(12:0) is placed on A(12:0), and Data(31) must be a zero. This is done because the Mode Register receives it's data on the address bus instead of the data bus. The SDRAM should be configured for the following settings: Burst Length = 4 (for 8-bit SDRAM) or Burst Length = 2 (for 16-bit SDRAM), CAS Latency = 3, Burst Type = Sequential.

4.4.8 SDRAM Power Down Mode

The DCKE signal is provided for putting the SDRAM into Power Down Mode. The Power Down Mode feature can be enabled or disabled using the ENSDRAMPD control bit in the Memory Configuration 0 Register. When this bit is set, the DCKE signal will be held low whenever there are no accesses to the SDRAM device for 7 CLK periods. Once a read, write or refresh is initiated to the SDRAM, the DCKE signal is asserted to bring the device out of Power Down Mode. Power Down Mode helps to reduce power dissipation in the SDRAM device when there are no accesses. During Memory Power Down the SDRAM device is placed into self refresh mode for maximum power savings and the DCLKOUT signal is stopped.

4.4.9 SDRAM Timing

Figures 4-15 through 4-17 show a few examples of the SDRAM timing. The first chip select (assertion of /CS) is used to latch the ROW address. Subsequent chip selects will either read or write data, and the final chip select is used to precharge the bank. Note from Table 4-8 that address bit 10 for 16-Mbit parts and address bit 11 for 64-Mbit parts is always a zero during the column address. This is done to ensure that auto precharge is never latched during a read or write command and also will cause only the bank that is being accessed to be precharged during the precharge command. Since the burst size of the SDRAMs is always setup to 32 bits, reads will always be 32 bits without regard to the actual number of bytes requested by the CPU. Writes are also 32 bits, but the data mask signals DQMH and DQML will mask out the bytes that should not be written, thus allowing for Tri, Half-Word and Byte writes.

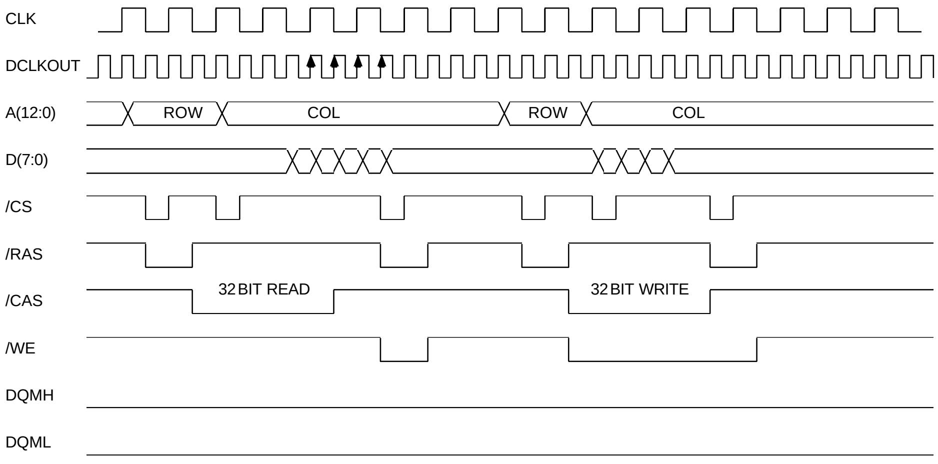


Figure 4-15. Word Read and Write to 8-Bit SDRAM

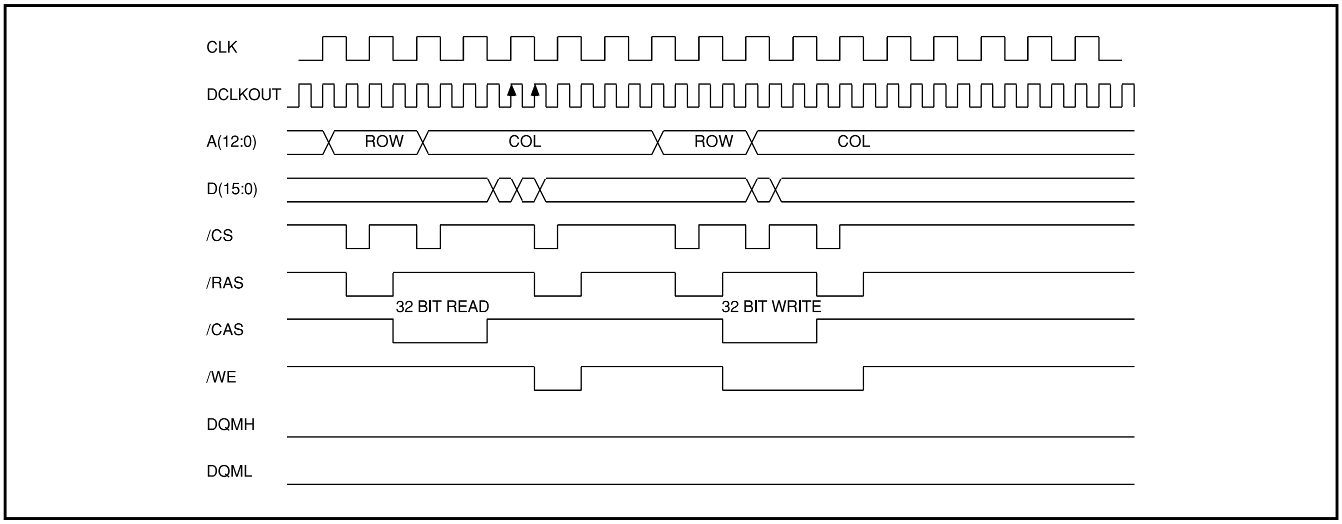


Figure 4-16. Word Read and Write to 16-Bit SDRAM

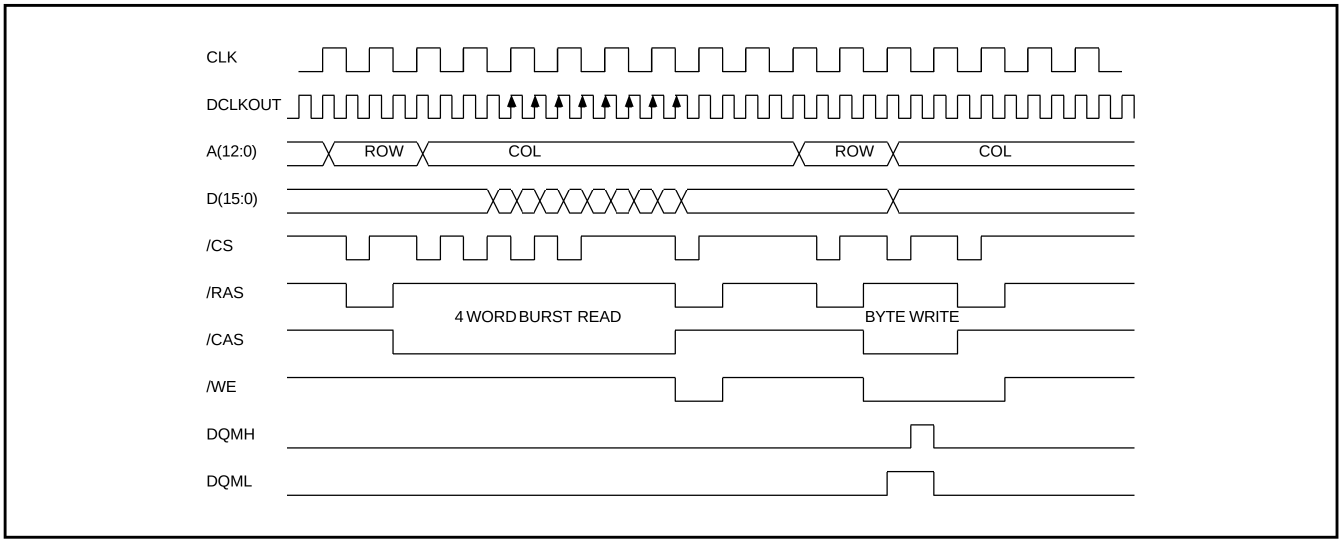


Figure 4-17. 4-Word Burst Read and Byte Write

4.5 Arbitration

The Arbitration logic contains the Refresh Controller for Bank 1 and Bank 0, support for External Bus Master, the Watch Dog Timer for terminating hung memory accesses, and Memory Power Down logic for placing the SDRAM and/or DRAM into self refresh mode.

4.5.1 Refresh Controller

The Refresh Controller logic contains separate refresh counters for Bank 1 and Bank 0. Each counter is a 6-bit counter that uses the 32 KHz clock for the counter. The refresh rate is set using the RFSHVAL1(5:0) and RFSHVAL2(5:0) control bits in the Memory Configuration 4 Register. These values are loaded into the respective refresh counter. The refresh counter will down count and when it reaches a count of zero, refresh will be inserted. If the RFSHVALx control bits are set to $00, the counter is not used and instead refresh is generated based on both the rising and falling edge of the 32 KHz clock. This is required to generate a 15.26 srefresh rate required by some DRAMs or SDRAMs. RFSHVALx control bits set to $01 will cause a refresh rate of 61.04 s and additional values will increase the refresh rate by increments of 30.52 s.When it is time for the Refresh Controller to insert refresh, the Halt signal will be asserted to the CPU Interface. Once the CPU Interface asserts the Halted signal, the Refresh Controller will generate the appropriate control signals that will cause the DRAM & SDRAM control logic to provide refresh for the memory devices. Once the Refresh Controller has completed inserting refresh, the Halt signal will be de-asserted so that the CPU Interface can continue running.

4.5.2 External Bus Master

The External Bus Master logic allows an external device to take control of the memory signals so that memory can be shared with an external device. The external device will request the bus by asserting the /DMAREQ signal. Once the /DMAREQ signal is asserted, the Halt signal will be asserted to the CPU Interface logic. Once the CPU Interface logic asserts the Halted signal, the External Bus Master logic will then assert the /DMAGRNT signal to inform the external device that it now controls the memory bus. At the same time that /DMAGRNT is asserted, the following signals are tri-stated: D(31:0), A(12:0), ALE, /RD, /WE, /CAS3-/CAS0, /RAS1, /RAS0, /DCS0, DCKE, DQMH, DQML and /CS0. DCLKOUT will be tri-stated if the ENDCLKOUTTRI control bit is set in the Memory Configuration 0 Register. Otherwise, DCLKOUT will continue to run. Once the external device de-asserts /DMAREQ, /DMAGRNT will be de-asserted, the memory signals will be driven by PR31700, and the Halt signal will be de-asserted so the CPU Interface can begin launching memory transactions. The external device should not attempt to take over the memory signals for a long period of time, otherwise the PR31700 DMA events and refresh will be interrupted. The External Bus Master logic will not respond with a /DMAGRNT if the memory interface is powered down.

4.5.3 Watch Dog Timer

The Watch Dog Timer prevents accesses to reserved memory locations from hanging the BIU. The Watch Dog Timer consists of a 10-bit down-counter whose initial value is determined by the WATCHTIMEVAL(3:0) control bits in the Memory Configuration 4 Register. The WATCHTIMEVAL(3:0) control bits are loaded into the upper 4 bits of the 10-bit counter and the lower 6 bits are loaded with $3F. The counter uses the CLK signal (nominally 36.864 MHz) to count. This will imply a maximum count of 28 sif the WATCHTIMEVAL(3:0) control bits are set to $F. The Watch Dog Timer rate is determined by the following equation:

$$\text{Watch Dog Rate} = \frac{(\text{WATCHTIMEVAL}(3{:}0) + 1) * 64}{36.864\ \text{MHz}}$$

The Watch Dog Timer is enabled using the ENWATCH control bit in the Memory Configuration 4 Register. When enabled, the counter will load whenever the CPU Interface asserts the Start signal to begin a memory transaction. If the Watch Dog Timer reaches a count of zero before the Memory State Machine asserts the Ack signal, the BusError signal is asserted to end the memory transaction. There are two possible ways that this can occur. First, if the address provided by the CPU interface to the Address Decoder is pointing to any of the reserved locations or to any of the chip selects that are not enabled, then the Memory State Machine will not respond with an Ack. Secondly, if the WAIT function is enabled to Card 2 or Card 1, then the /WAIT2 or /WAIT1 signals must de-assert before the Watch Dog Timer reaches a count of zero, otherwise the BusError signal will be asserted and the Card access will be aborted.

4.5.4 Memory Power Down

The memory interface is powered down whenever the MEMPOWERDOWN control bit is set in the Memory Configuration 4 Register. When this bit is set, the Halt signal will be asserted to the CPU Interface, the DRAM and/or SDRAM will be placed into self refresh mode, and all memory interface signals will be driven low. The memory interface will wake up whenever the CPU Interface asserts the HaltCycle signal and the address defined by the Addr(31:2) bits correspond to an address other than the internal PR31700 registers. When the memory interface wakes up, the DRAM and/or SDRAM are taken out of self refresh mode and the Halt signal is de-asserted to allow the CPU Interface to launch memory transactions. Figures 4-18 and 4-19 show timing for the wake-up and power-down of the memory interface.

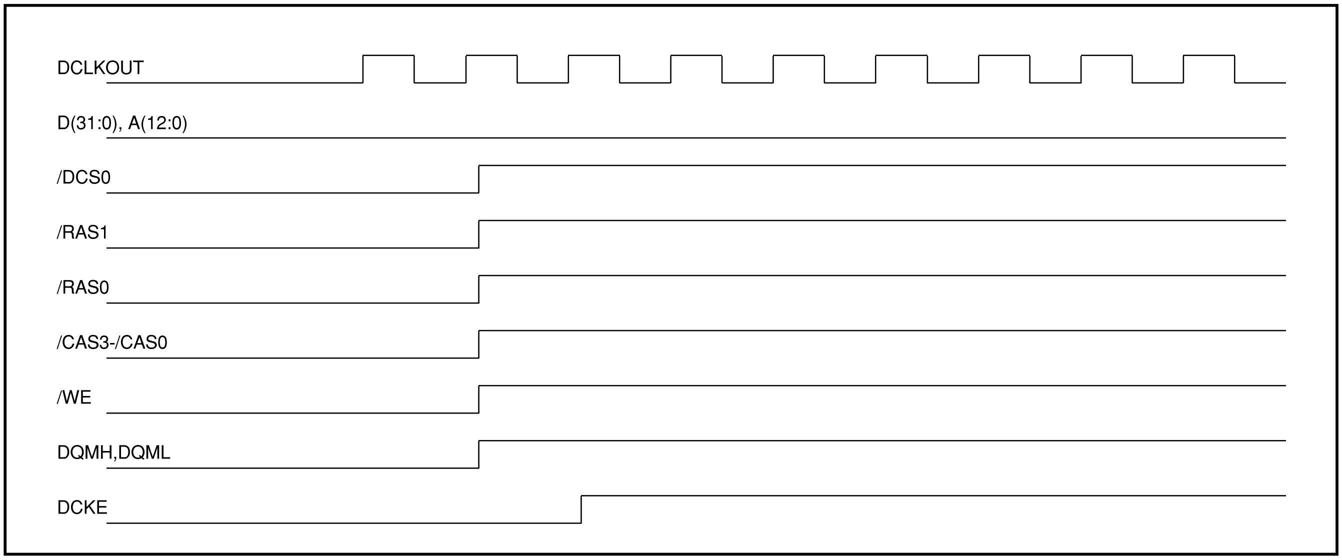


Figure 4-18. Memory Interface Wake-Up

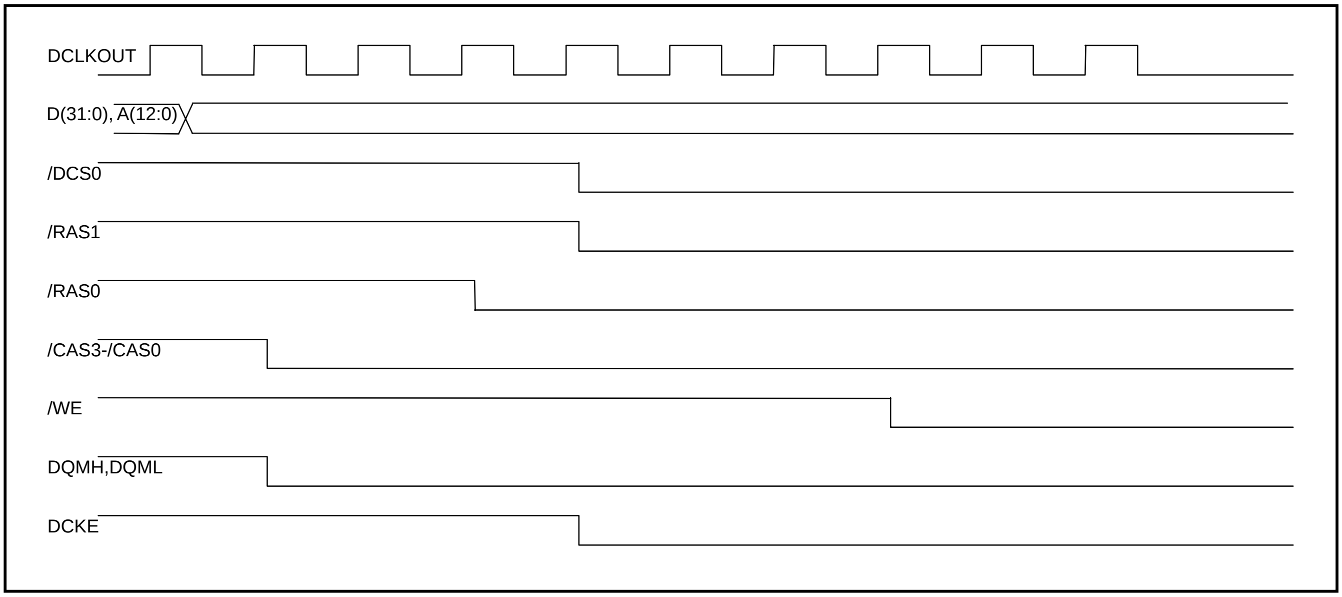


Figure 4-19. Memory Interface Power-Down

When the memory interface is powered down all memory signals are driven low. The order in which they are driven low is important because DRAM devices enter self refresh mode by asserting /CAS before /RAS, and SDRAM devices enter self refresh mode by de-asserting DCKE while the chip select to the SDRAM, /DCS0 and/or /RAS1, /RAS0 and /CAS0 are asserted with /WE not asserted. Once the SDRAM enters self refresh mode, DCLKOUT can be turned off. The ordering makes it possible to put both DRAM and SDRAM devices into self refresh mode with only one timing set required. When the memory interface comes out of self refresh mode, all signals are de-asserted. DCKE must de-assert one clock later in order to properly bring the SDRAM out of self refresh mode. After exiting self refresh mode, the Halt signal will not be released for several clocks to allow the DRAM and SDRAM to finish exiting self refresh mode.

4.6 Memory Connections

Figures 4-20 through 4-22 illustrate the various signal connections between the PR31700 and external SDRAM, DRAM, and static memory devices. These figures show the connections for various bus width configurations (e.g., 8-bit, 16-bit, 32-bit).

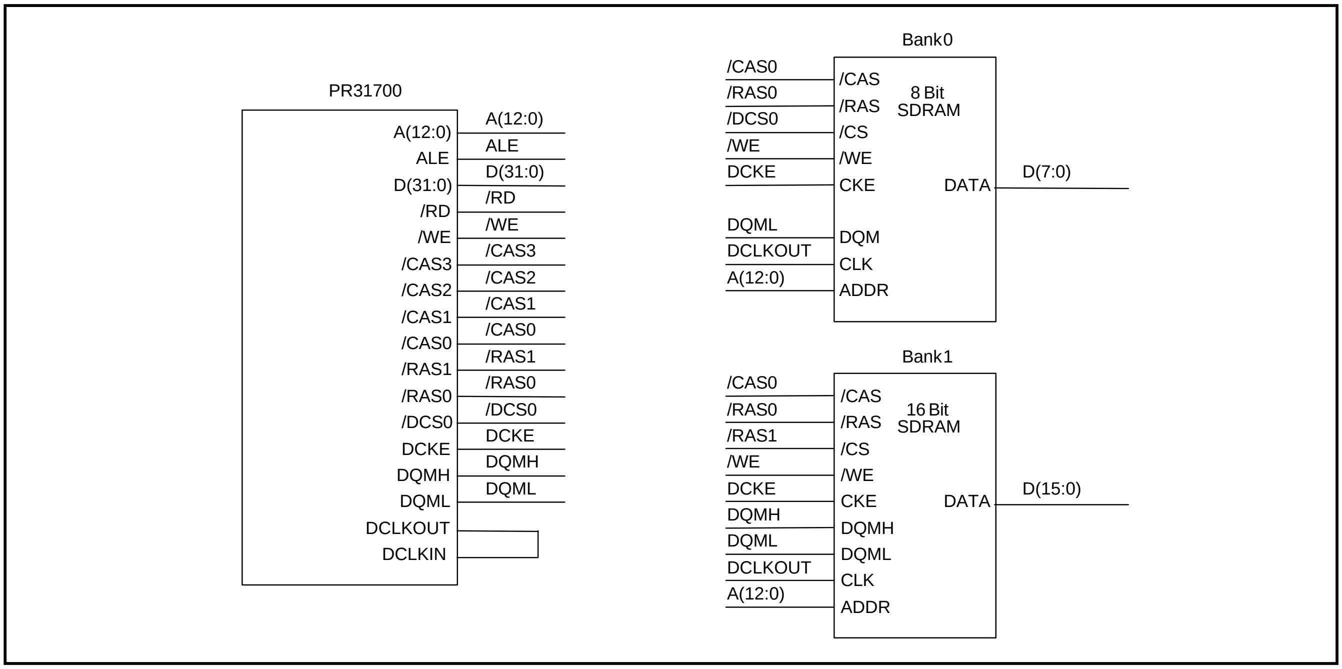


Figure 4-20. SDRAM Memory Connection

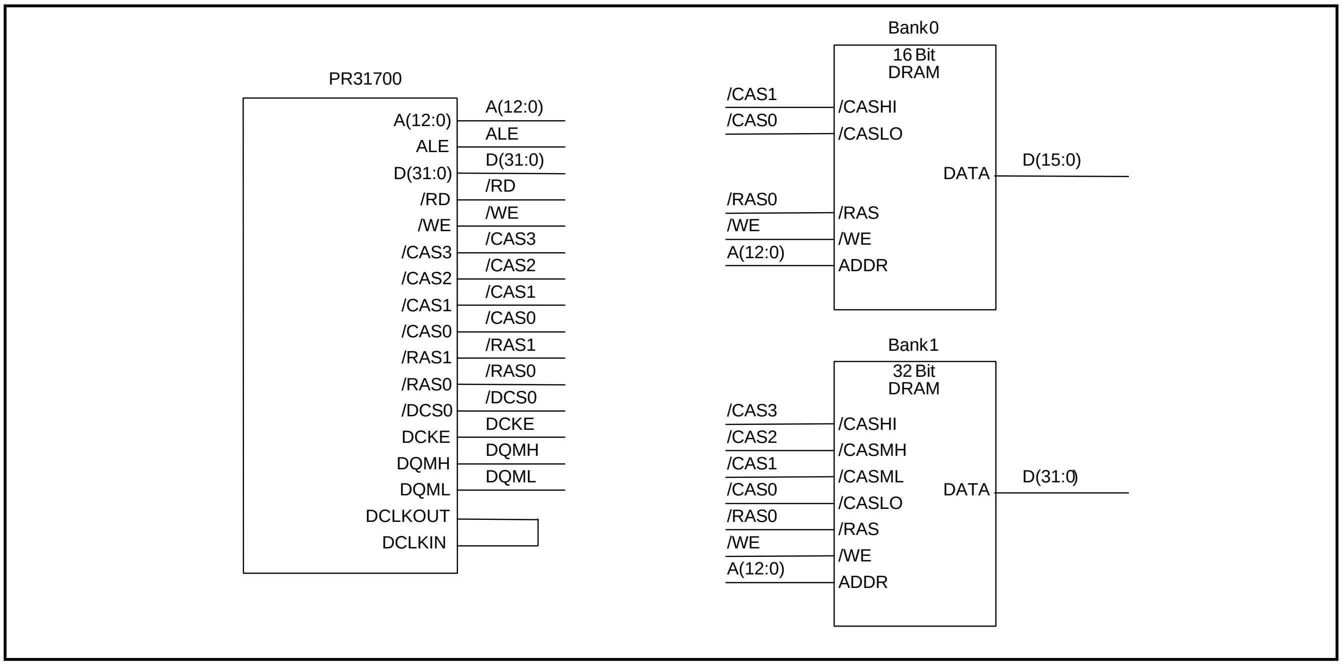


Figure 4-21. DRAM Memory Connection

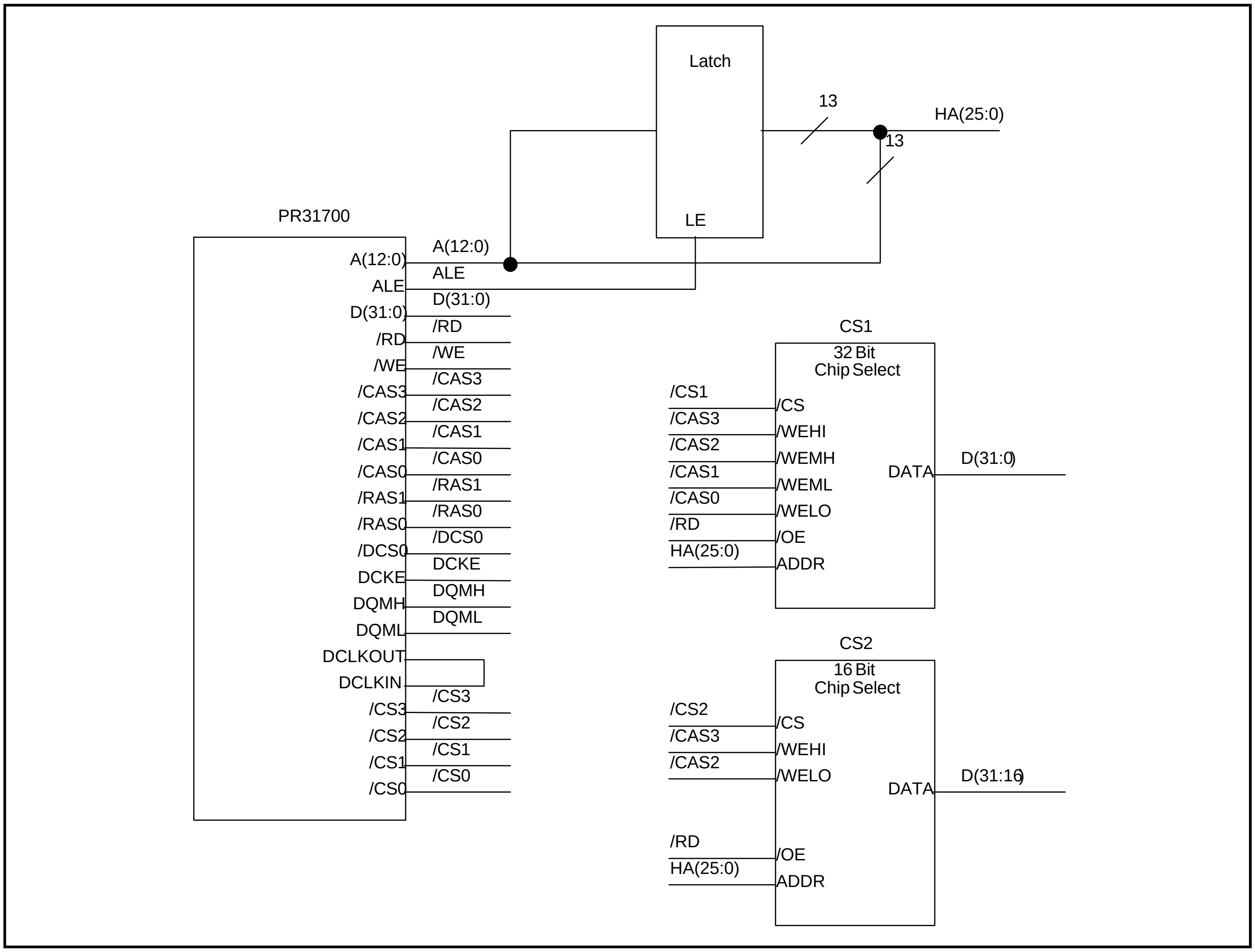


Figure 4-22. Chip Select Memory Connection

4.7 BIU Registers

4.7.1 Memory Configuration 0 Register

OFFSET = $000:

Bit	Label	RESET	Read/Write
31	Reserved		
30	ENDCLKOUTTRI	0	R/W
29	DISDQMINIT	0	R/W
28	ENSDRAMPD	0	R/W
27	SHOWPR31700	0	R/W
26	ENRMAP2	0	R/W
25	ENRMAP1	0	R/W
24	ENWRINPAGE	0	R/W
23	ENCS3USER	X	R/W
22	ENCS2USER	X	R/W
21	ENCS1USER	X	R/W
20	ENCS1DRAM	X	R/W
19-18	BANK1CONF(1:0)	X	R/W
17-16	BANK0CONF(1:0)	X	R/W
15-14	ROWSEL1(1:0)	X	R/W
13-12	ROWSEL0(1:0)	X	R/W
11-8	COLSEL1(2:0)	X	R/W
7-4	COLSEL0(2:0)	X	R/W
3	CS3SIZE	X	R/W
2	CS2SIZE	X	R/W
1	CS1SIZE	X	R/W
0	CS0SIZE	See Description	R/W

ENDCLKOUTTRI:
Setting this bit will cause the DCLKOUT pin to tri-state when /DMAGRNT is asserted during external bus arbitration. Otherwise, DCLKOUT will continue to operate.

DISDQMINIT:
If this bit is a cleared, the DQMH and DQML signals will go high when coming out of power-down and remain high until the first access to SDRAM. If this bit is set, this function is disabled and the DQMH and DQML signals will remain low.

ENSDRAMPD:
Setting this bit will cause the BIU to put the SDRAMs into Power Down mode by lowering DCKE whenever there are no accesses to the SDRAMs for 7 CLK periods.

SHOWPR31700:
Setting this bit will cause internal reads and writes by the processor to the PR31700 registers to show externally for debug. The /RD signal provides the read or write status, The D(31:0) signals provide the data, A(12:0) provide the address, and the /CARDREG signal provides a chip select on the rising edge for sampling the signals.

ENRMAP2:
Setting this bit will enable Address Re-Mapper 2.

ENRMAP1:
Setting this bit will enable Address Re-Mapper 1.

ENWRINPAGE:
Setting this bit will cause the DRAMs and/or SDRAMs to remain in page mode when the WrInPage signal is asserted by the CPU Interface. If this bit is cleared, the WrInPage signal is ignored.

ENCS3USER:
Setting this bit will enable CS3 in the kuser Space.

ENCS2USER:
Setting this bit will enable CS2 in the kuser Space.

ENCS1USER:
Setting this bit will enable CS1 in the kuser Space.

ENCS1DRAM:
Setting this bit will enable CS1 in Bank 1 and Bank 0 Space.

BANK1CONF(1:0):
These bits provide the configuration for Bank 1 according to the following:

11	16-bit SDRAM
10	8-bit SDRAM
01	32-bit DRAM (standard or EDO type)
00	16-bit DRAM (standard or EDO type)

BANK0CONF(1:0):
These bits provide the configuration for Bank 0 according to the following:

11	16-bit SDRAM
10	8-bit SDRAM
01	32-bit DRAM (standard or EDO type)
00	16-bit DRAM (standard or EDO type)

ROWSEL1(1:0):

ROWSEL0(1:0):

COLSEL1(2:0):

COLSEL0(2:0):
These bits determine the Row and Column configuration for addressing according to the following:

ROW A(12:0)	ROWSEL	COL A(12:0)	COLSEL
18,17:9	00	22,20,18,8:1	0000
22,18,20,19,17:9	01	19,18,8:2	0001
20,22,21,19,17:9	10	21,20,18,8:2	0010
22,23,21,19,17:9	11	23,22,20,18,8:2	0011
		24,22,20,18,8:2	0100
		18,Ø,X,8:0	0101
		22,Ø,X,21,8:0	0110
		18,Ø,X,21,8:1	0111
		22,Ø,X,23,21,8:1	1000
		24,23,21,8:2	1001

CS3SIZE:
Setting this bit defines CS3 to be a 32-bit port. Otherwise, a 16-bit port is assumed.

CS2SIZE:
Setting this bit defines CS2 to be a 32-bit port. Otherwise, a 16-bit port is assumed.

CS1SIZE:
Setting this bit defines CS1 to be a 32-bit port. Otherwise, a 16-bit port is assumed.

CS0SIZE:
Setting this bit defines CS0 to be a 32-bit port. Otherwise, a 16-bit port is assumed. On RESET, this signal is set high if the TESTSIU pin is tied high and low if the TESTSIU pin is tied low.

4.7.2 Memory Configuration 1 Register

OFFSET = $004:

Bit	Label	RESET	Read/Write
31-28	MCS3ACCVAL1(3:0)	X	R/W
27-24	MCS3ACCVAL2(3:0)	X	R/W
23-20	MCS2ACCVAL1(3:0)	X	R/W
19-16	MCS2ACCVAL2(3:0)	X	R/W
15-12	MCS1ACCVAL1(3:0)	X	R/W
11-8	MCS1ACCVAL2(3:0)	X	R/W
7-4	MCS0ACCVAL1(3:0)	X	R/W
3-0	MCS0ACCVAL2(3:0)	X	R/W

MCS3ACCVAL1(3:0):
These bits define the access time for MCS3. If ENMCS3PAGE is set, these bits will be used for the first access and MCS3ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENMCS3PAGE is not set, these bits and MCS3ACCVAL2(3:0) should be set with the same value. If ENMCS3PAGE is set, these bits must be set to at least $1.

MCS3ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENMCS3PAGE is set. If ENMCS3PAGE is not set, these bits and MCS3ACCVAL1(3:0) should be set with the same value. If ENMCS3PAGE is set, these bits must be set to at least $1.

MCS2ACCVAL1(3:0):
These bits define the access time for MCS2. If ENMCS2PAGE is set, these bits will be used for the first access and MCS2ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENMCS2PAGE is not set, these bits and MCS2ACCVAL2(3:0) should be set with the same value. If ENMCS2PAGE is set, these bits must be set to at least $1.

MCS2ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENMCS2PAGE is set. If ENMCS2PAGE is not set, these bits and MCS2ACCVAL1(3:0) should be set with the same value. If ENMCS2PAGE is set, these bits must be set to at least $1.

MCS1ACCVAL1(3:0):
These bits define the access time for MCS1. If ENMCS1PAGE is set, these bits will be used for the first access and MCS1ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENMCS1PAGE is not set, these bits and MCS1ACCVAL2(3:0) should be set with the same value. If ENMCS1PAGE is set, these bits must be set to at least $1.

MCS1ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENMCS1PAGE is set. If ENMCS1PAGE is not set, these bits and MCS1ACCVAL1(3:0) should be set with the same value. If ENMCS1PAGE is set, these bits must be set to at least $1.

MCS0ACCVAL1(3:0):
These bits define the access time for MCS0. If ENMCS0PAGE is set, these bits will be used for the first access and MCS0ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENMCS0PAGE is not set, these bits and MCS0ACCVAL2(3:0) should be set with the same value. If ENMCS0PAGE is set, these bits must be set to at least $1.

MCS0ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENMCS0PAGE is set. If ENMCS0PAGE is not set, these bits and MCS0ACCVAL1(3:0) should be set with the same value. If ENMCS0PAGE is set, these bits must be set to at least $1.

4.7.3 Memory Configuration 2 Register

OFFSET = $008:

Bit	Label	RESET	Read/Write
31-28	CS3ACCVAL1(3:0)	X	R/W
27-24	CS3ACCVAL2(3:0)	X	R/W
23-20	CS2ACCVAL1(3:0)	X	R/W
19-16	CS2ACCVAL2(3:0)	X	R/W
15-12	CS1ACCVAL1(3:0)	X	R/W
11-8	CS1ACCVAL2(3:0)	X	R/W
7-4	CS0ACCVAL1(3:0)	$F	R/W
3-0	CS0ACCVAL2(3:0)	$F	R/W

CS3ACCVAL1(3:0):
These bits define the access time for CS3. If ENCS3PAGE is set, these bits will be used for the first access and CS3ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENCS3PAGE is not set, these bits and CS3ACCVAL2(3:0) should be set with the same value. If ENCS3PAGE is set, these bits must be set to at least $1.

CS3ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENCS3PAGE is set. If ENCS3PAGE is not set, these bits and CS3ACCVAL1(3:0) should be set with the same value. If ENCS3PAGE is set, these bits must be set to at least $1.

CS2ACCVAL1(3:0):
These bits define the access time for CS2. If ENCS2PAGE is set, these bits will be used for the first access and CS2ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENCS2PAGE is not set, these bits and CS2ACCVAL2(3:0) should be set with the same value. If ENCS2PAGE is set, these bits must be set to at least $1.

CS2ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENCS2PAGE is set. If ENCS2PAGE is not set, these bits and CS2ACCVAL1(3:0) should be set with the same value. If ENCS2PAGE is set, these bits must be set to at least $1.

CS1ACCVAL1(3:0):
These bits define the access time for CS1. If ENCS1PAGE is set, these bits will be used for the first access and CS1ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENCS1PAGE is not set, these bits and CS1ACCVAL2(3:0) should be set with the same value. If ENCS1PAGE is set, these bits must be set to at least $1.

CS1ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENCS1PAGE is set. If ENCS1PAGE is not set, these bits and CS1ACCVAL1(3:0) should be set with the same value. If ENCS1PAGE is set, these bits must be set to at least $1.

CS0ACCVAL1(3:0):
These bits define the access time for CS0. If ENCS0PAGE is set, these bits will be used for the first access and CS0ACCVAL2(3:0) will be used for the next three accesses for reads only. For writes, only these bits are used. If ENCS0PAGE is not set, these bits and CS0ACCVAL2(3:0) should be set with the same value. If ENCS0PAGE is set, these bits must be set to at least $1.

CS0ACCVAL2(3:0):
These bits define the access time for the next three reads in a burst sequence if ENCS0PAGE is set. If ENCS0PAGE is not set, these bits and CS0ACCVAL1(3:0) should be set with the same value. If ENCS0PAGE is set, these bits must be set to at least $1.

4.7.4 Memory Configuration 3 Register

OFFSET = $00C:

Bit	Label	RESET	Read/Write
31-28	CARD2ACCVAL(3:0)	X	R/W
27-24	CARD1ACCVAL(3:0)	X	R/W
23-20	CARD2IOACCVAL(3:0)	X	R/W
19-16	CARD1IOACCVAL(3:0)	X	R/W
15	ENMCS3PAGE	X	R/W
14	ENMCS2PAGE	X	R/W
13	ENMCS1PAGE	X	R/W
12	ENMCS0PAGE	X	R/W
11	ENCS3PAGE	X	R/W
10	ENCS2PAGE	X	R/W
9	ENCS1PAGE	X	R/W
8	ENCS0PAGE	0	R/W
7	CARD2WAITEN	X	R/W
6	CARD1WAITEN	X	R/W
5	CARD2IOEN	X	R/W
4	CARD1IOEN	X	R/W
3	8PORTSEL	0	R/W
2-0	Reserved		

CARD2ACCVAL(3:0):
These bits define the access time for Card 2 Memory Space.

CARD1ACCVAL(3:0):
These bits define the access time for Card 1 Memory Space.

CARD2IOACCVAL(3:0):
These bits define the access time for Card 2 IO and Attribute Space.

CARD1IOACCVAL(3:0):
These bits define the access time for Card 1 IO and Attribute Space.

ENMCS3PAGE:
Setting this bit will enable Read Page Mode for MCS3.

ENMCS2PAGE:
Setting this bit will enable Read Page Mode for MCS2.

ENMCS1PAGE:
Setting this bit will enable Read Page Mode for MCS1.

ENMCS0PAGE:
Setting this bit will enable Read Page Mode for MCS0.

ENCS3PAGE:
Setting this bit will enable Read Page Mode for CS3.

ENCS2PAGE:
Setting this bit will enable Read Page Mode for CS2.

ENCS1PAGE:
Setting this bit will enable Read Page Mode for CS1.

ENCS0PAGE:
Setting this bit will enable Read Page Mode for CS0.

CARD2WAITEN:
Setting this bit will enable the /CARD2WAIT signal.

CARD1WAITEN:
Setting this bit will enable the /CARD1WAIT signal.

CARD2IOEN:
Setting this bit will cause accesses to Card 2 IO space to assert the /IORD or /IOWR signals. This is used for accessing IO cards. If this bit is not set, /RD or /WE is asserted, which provides Attribute space access.

CARD1IOEN:
Setting this bit will cause accesses to Card 1 IO space to assert the /IORD or /IOWR signals. This is used for accessing IO cards. If this bit is not set, /RD or /WE is asserted, which provides Attribute space access.

8PORTSEL:
This bit defines the PCMCIA port size.

0 :	16 bit port access
1 :	8 bit port access

4.7.5 Memory Configuration 4 Register

OFFSET = $010:

Bit	Label	RESET	Read/Write
31	ENEDOBANK1	0	R/W
30	ENEDOBANK0	0	R/W
29	ENARB	0	R/W
28	DISSNOOP	0	R/W
27	CLRWRBUSERRINT	0	R/W
26	ENBANK1OPT	0	R/W
25	ENBANK0OPT	0	R/W
24	ENWATCH	0	R/W
23-20	WATCHTIMEVAL(3:0)	X	R/W
19-17	Reserved		
16	MEMPOWERDOWN	1	R/W
15	ENRFSH1	0	R/W
14	ENRFSH0	0	R/W
13-8	RFSHVAL1(5:0)	X	R/W
7-6	Reserved		
5-0	RFSHVAL0(5:0)	X	R/W

ENEDOBANK1:
Setting this bit will enable EDO DRAM in Bank 1. The BANK1CONF bit in the Memory Configuration 0 register will configure the PR31700 to support either a 16-bit or 32-bit EDO configuration in the same manner that standard DRAM is setup. The 60 ns EDO DRAMs is recommended to use when running with a CLK2X of 73.728 MHz for optimum system performance.

ENEDOBANK0:
Setting this bit will enable EDO DRAM in Bank 0. The BANK0CONF bit in the Memory Configuration 0 register will configure the PR31700 to support either a 16-bit or 32-bit EDO configuration in the same manner that standard DRAM is setup. The 60 ns EDO DRAMs is recommended to use when running with a CLK2X of 73.728 MHz for optimum system performance.

ENARB:
Setting this bit will always cause the CPU interface logic to externally arbitrate with the CPU during DMA reads and writes when the DISSNOOP control bit is set. Normally, DMA transactions take place through the CPU interface logic by stalling the CPU via the Bus Request and Bus Acknowledge mechanism in the CPU. When the Bus Request signal is asserted, the CPU will stall and assert the Bus Acknowledge signal. Setting the ENARB bit will cause the CPU interface logic to instead externally arbitrate between the CPU and the DMA events without using the Bus Request and Bus Acknowledge mechanism. This has the advantage that the processor will not stall during DMA events unless there is a Cache miss which will cause a contention for external memory.

DISSNOOP:
Setting this bit will prevent the CPU interface logic from causing the SNOOP signal to assert to the CPU Core during DMA writes.

CLRWRBUSERRINT:
An interrupt has been added that will occur whenever a write is timed out by the Watch Dog Timer. This interrupt is connected directly to INT[0] on the CPU and is completely independent of the rest of the PR31700 interrupt logic. When this interrupt is set, it will remain set until the CLRWRBUSERRINT bit is asserted . The bit must be set and subsequently cleared to clear the interrupt.

ENBANK1OPT:
Setting this bit will cause the BIU to insert an extra clock of delay between the assertion of /RAS1 and the assertion of the /CAS signals for DRAM accesses. This is needed for interfacing to 80 ns DRAMs when running with a clock greater than 70 MHz.

ENBANK0OPT:
Setting this bit will cause the BIU to insert an extra clock of delay between the assertion of /RAS0 and the assertion of the /CAS signals for DRAM accesses. This is needed for interfacing to 80 ns DRAMs when running with a clock greater than 70 MHz.

ENWATCH:
Setting this bit will enable the Watch Dog Timer.

WATCHTIMEVAL(3:0):
These bits define the length of the Watch Dog Timer. The Watch Dog Timer rate is determined by the following equation:

MEMPOWERDOWN:
Setting this bit will cause the memory interface to be put into Memory Power Down mode which will cause the DRAMs and SDRAMs to be put into self-refresh and all memory signals will be driven low. The memory interface will remain powered down until the processor attempts to access a location in memory other than the PR31700 registers.

ENRFSH1:
Setting this bit will cause the BIU to begin refreshing Bank 1 DRAMs or SDRAMs. This bit should not be set until meeting the start up specification requirements for the memory devices.

ENRFSH0:
Setting this bit will cause the BIU to begin refreshing Bank 0 DRAMs or SDRAMs. This bit should not be set until meeting the start up specification requirements for the memory devices.

RFSHVAL1(5:0):
These bits define the refresh period for Bank 1 refresh according to the following:

$0	15.26 s
$1	61.04 s
$2	91.55 s
$3	122.07 s

.
.

RFSHVAL0(5:0):
These bits define the refresh period for Bank 0 refresh according to the following:

$0	15.26 s
$1	61.04 s
$2	91.55 s
$3	122.07 s

.
.

4.7.6 Memory Configuration 5 Register

OFFSET = $014: **write-only**

Bit	Label	RESET	Read/Write
31-9	STARTVAL2(31:9)	X	W
8-4	Reserved		
3-0	MASK2(3:0)	X	W

STARTVAL2(31:9): **write-only**

These bits define the start address for re-map region 2.

MASK2(3:0): **write-only**

These bits define the mask for re-map region 2.

4.7.7 Memory Configuration 6 Register

OFFSET = $018: **write-only**

Bit	Label	RESET	Read/Write
31-9	STARTVAL1(31:9)	X	W
8-4	Reserved		
3-0	MASK1(3:0)	X	W

STARTVAL1(31:9): **write-only**

These bits define the start address for re-map region 1.

MASK1(3:0): **write-only**

These bits define the mask for re-map region 1.

4.7.8 Memory Configuration 7 Register

OFFSET = $01C: **write-only**

Bit	Label	RESET	Read/Write
31-9	RMAPADD2(31:9)	X	W
8-0	Reserved		

RMAPADD2(31:9): **write-only**

These bits define the destination address for re-map region 2.

4.7.9 Memory Configuration 8 Register

OFFSET = $020: **write-only**

Bit	Label	RESET	Read/Write
31-9	RMAPADD1(31:9)	X	W
8-0	Reserved		

RMAPADD1(31:9): **write-only**

These bits define the destination address for re-map region 1.

Chapter 5 SIU Module

PR31700 V0.3

This section describes the System Interface Unit (SIU) Module, which contains the multi-channel DMA controller, as well as the address decoding for all the submodules within the System Interface Module (SIM).

5.1 Overview

The SIU Module within the PR31700 provides the multi-channel 32-bit DMA controller used to select and arbitrate the DMA channels from each SIM DMA source. Independent DMA channels are provided video, sound, telecom, CHI, UART-A, and UART-B sources.

The SIU Module also contains the address decoder for all of the submodules contained within the SIM. This decoder generates the read and write enable pulses for each of the internal PR31700 registers.

5.2 Implementation

5.2.1 Block Diagram

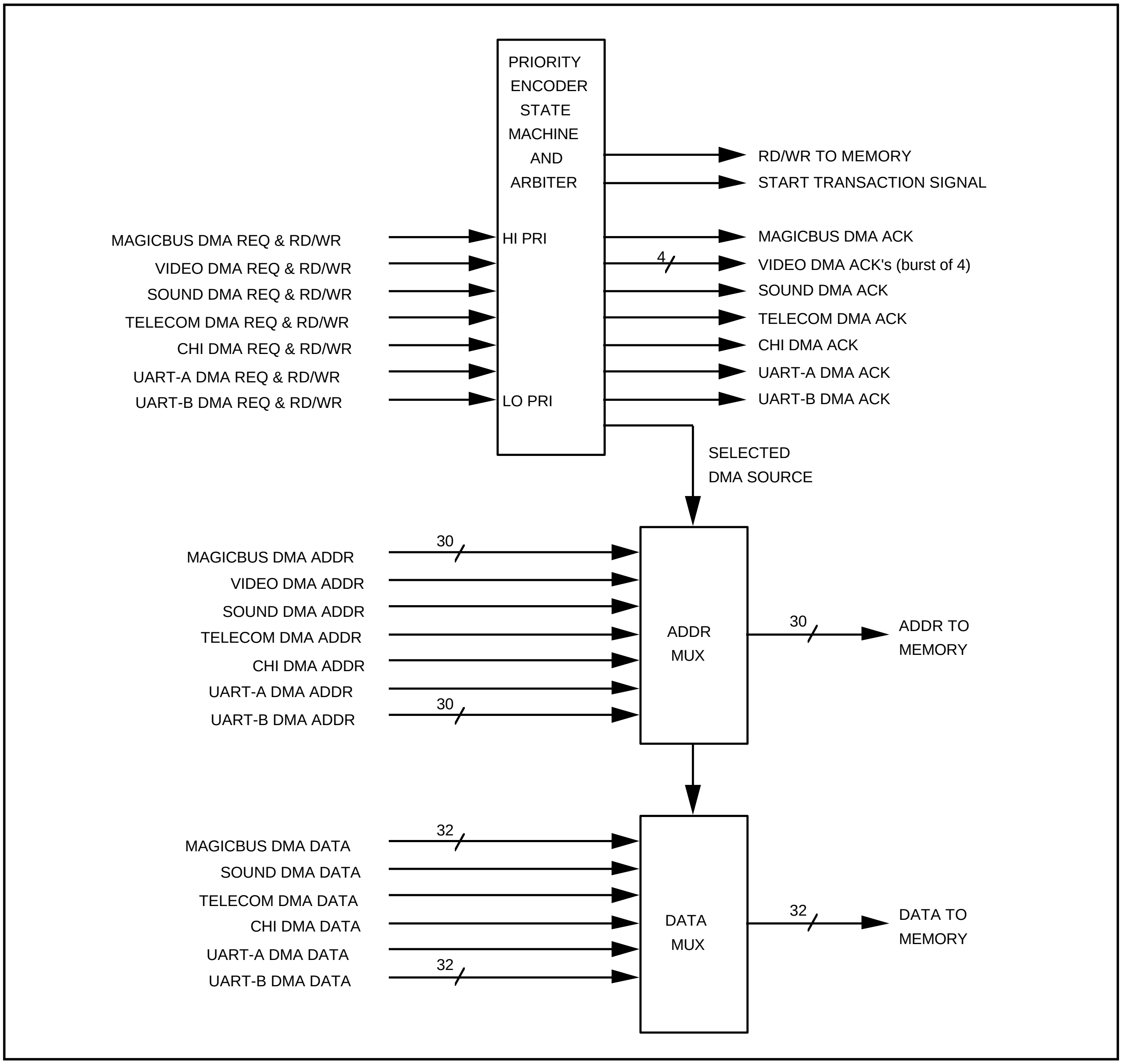


Figure 5-1. SIU DMA Controller Block Diagram

5.2.2 DMA Controller Description

Figure 5-1 shows a block diagram of the SIU DMA Controller. The DMA Controller receives the DMA requests and read/write status from all of the 6 possible DMA sources (video, sound, telecom, CHI, UART-A, and UART-B) and uses a Priority Encoder to arbitrate and select the highest priority request. If a new request is received and the DMA controller is currently busy, that request will remain pending and will be processed as soon as all other higher priority pending requests have been processed. The priority of the DMA sources and the DMA direction (read or write) supported are shown in Table 5-1.

Table 5-1. DMA Sources

DMA source	priority	DMA read (from memory)	DMA write (to memory)
DMA		yes	NO
video (burst of 4 longs)		yes	yes
sound		yes	yes
telecom		yes	yes
CHI		yes	yes
UART-A (1 byte)		yes	yes
UART-B (1 byte)	lowest	yes	yes

The state machine for the Priority Encoder also generates the START transaction pulse and read versus write status signal to the CPU Interface for each DMA transaction. After the arbiter decodes all received DMA requests and selects the highest priority request as the DMA channel to be processed, the DMA address and data busses for the selected DMA channel are routed to the memory subsystem. All DMA sources transfer 1 longword per transaction, except for the video which always transfers 4 consecutive longwords at a time (and thus returns 4 separate ACK's to the Video Module), and the UART's which transfer only 1 byte at a time. The UART's are thus tagged with the appropriate Byte Enable (corresponding to 1 of 4 byte positions) depending on the byte count in the DMA address counter within the respective UART Module. The state machine also generates the required acknowledge (ACK) signal at the end of each DMA transaction. This ACK signal is used by the circuit corresponding to the respective DMA source to either latch the data from memory (for a read transaction) or to enable the data to memory (for a write transaction). The SIU also contains several test bits (see Section 5.3) which allow detailed testing of the DMA Controller for various combinations of simultaneous DMA requests. These bits are used for IC testing and should never be set.

5.2.3 Address Decoder Description

Figure 5-2 shows a block diagram of the SIM Address Decoder. The Address Decoder generates the read and write enable pulses for each of the internal PR31700 control and status registers. The lower 6 bits of the internal address bus are decoded, along with the SIM chip select and read/write status signal, to generate any 1 of 64 possible write enable pulses or any 1 of 64 possible read enable pulses. These enable pulses are used by the circuit/module targeted by the respective read or write pulse to either latch the data from the CPU (for a write transaction) or to enable the data to the CPU (for a read transaction). See the next section for a full listing of all the internal PR31700 Registers, including their address offset and read versus write support. The address offset for PR31700 Registers is the offset from the base address ($10C00000) of the PR31700 Registers in the System Address Map.

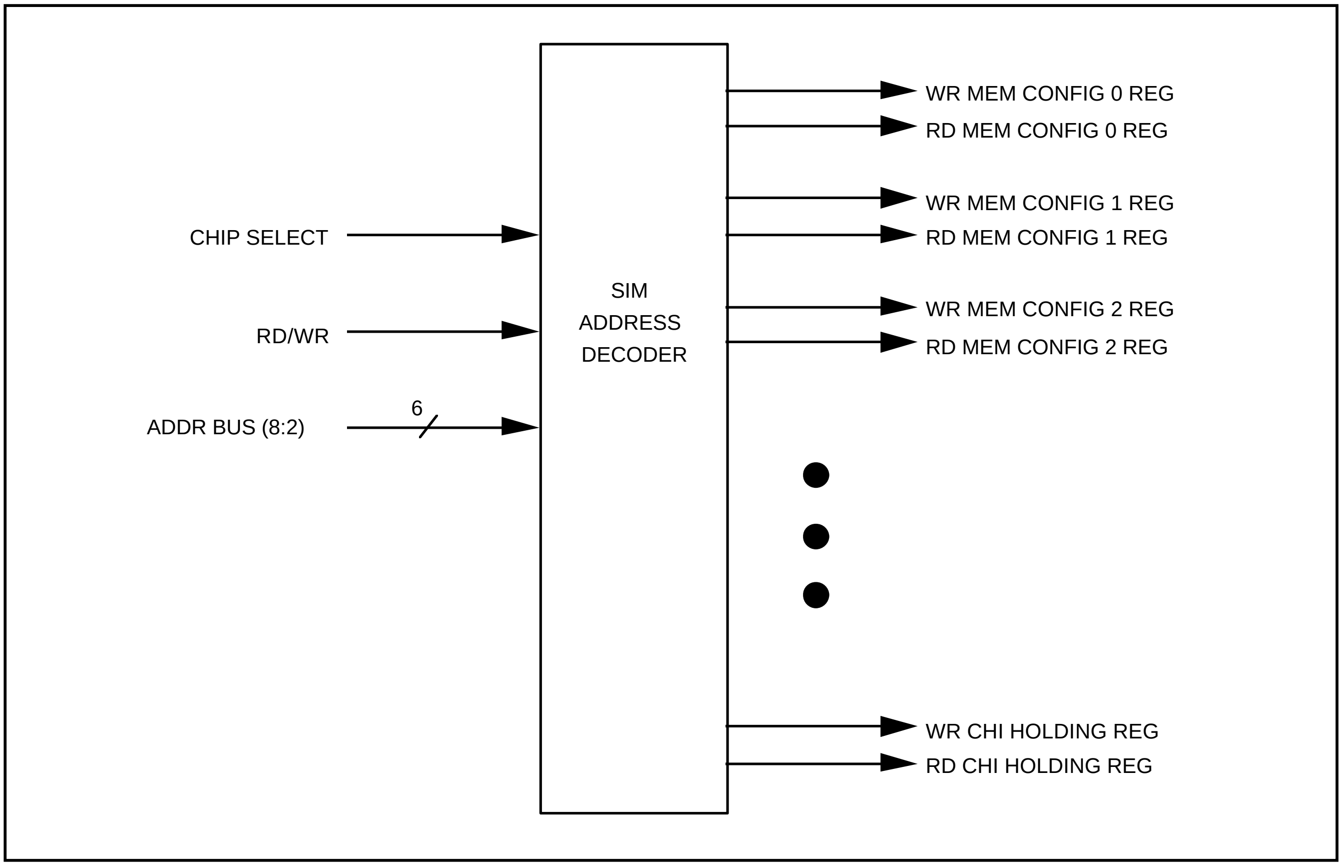


Figure 5-2. SIM Address Decoder Block Diagram

5.2.4 Internal PR31700 Registers

Offset	Read/Write	Register
000	R/W	Memory Configuration 0
004	R/W	Memory Configuration 1
008	R/W	Memory Configuration 2
00C	R/W	Memory Configuration 3
010	R/W	Memory Configuration 4
014	W	Memory Configuration 5
018	W	Memory Configuration 6
01C	W	Memory Configuration 7
020	W	Memory Configuration 8
028	R/W	Video Control 1
02C	W	Video Control 2
030	W	Video Control 3
034	W	Video Control 4
038	W	Video Control 5
03C	W	Video Control 6
040	W	Video Control 7
044	W	Video Control 8
048	W	Video Control 9
04C	W	Video Control 10
050	W	Video Control 11
054	W	Video Control 12
058	W	Video Control 13
05C	W	Video Control 14
060	W	SIB Size
064	W	SIB Sound Receive Start
068	W	SIB Sound Transmit Start
06C	W	SIB Tel Receive Start
070	W	SIB Tel Transmit Start
074	R/W	SIB Control
078	R/W	SIB Sound Holding Register
07C	R/W	SIB Tel Holding Register
080	R/W	SIB SF0 Control
084	R/W	SIB SF1 Control
088	R	SIB SF0 Status
08C	R	SIB SF1 Status
090	R/W	SIB DMA Control
0A0	R/W	IR Control 1
0A4	W	IR Control 2
0A8	W IR	Holding Register
0B0	R/W	UARTA Control 1
0B4	W	UARTA Control 2
0B8	W	UARTA DMA Control 1
0BC	W	UARTA DMA Control 2
0C0	R	UARTA DMA Count
0C4	R/W	UARTA Holding Register
0C8	R/W	UARTB Control 1
0CC	W	UARTB Control 2
0D0	W	UARTB DMA Control 1
0D4	W	UARTB DMA Control 2
0D8	R	UARTB DMA Count
0DC	R/W	UARTB Holding Register
0E0	R/W	Reserved
0E4	W	Reserved
0E8	W	Reserved
0EC	W	Reserved
0F0	R	Reserved
0F4	W	Reserved
0F8	R/W	Reserved
100	W	Clear Interrupt 1

Offset	Read/Write	Register
100	R	Interrupt Status 1
104	W	Clear Interrupt 2
104	R	Interrupt Status 2
108	W	Clear Interrupt 3
108	R	Interrupt Status 3
10C	W	Clear Interrupt 4
10C	R	Interrupt Status 4
110	W	Clear Interrupt 5
110	R	Interrupt Status 5
114	R	Interrupt Status 6
118	R/W	Enable Interrupt 1
11C	R/W	Enable Interrupt 2
120	R/W	Enable Interrupt 3
124	R/W	Enable Interrupt 4
128	R/W	Enable Interrupt 5
12C	R/W	Enable Interrupt 6
140	R	RTC High
144	R	RTC Low
148	R/W	Alarm High
14C	R/W	Alarm Low
150	R/W	Timer Control
154	R/W	Periodic Timer
160	R/W	SPI Control
164	R/W	SPI Holding Register
180	R/W	IO Control
184	R/W	Multi-function IO Data Out
188	R/W	Multi-function IO Direction
18C	R	Multi-function IO Data In
190	R/W	Multi-function IO Select
194	R/W	IO Power Down
198	R/W	Multi-function IO Power Down
1C0	R/W	Clock Control
1C4	R/W	Power Control
1C8	R/W	SIU Test
1D8	R/W	CHI Control
1DC	R/W	CHI Pointer Enable
1E0	W CHI	Receive Pointer A
1E4	W CHI	Receive Pointer B
1E8	W CHI	Transmit Pointer A
1EC	W CHI	Transmit Pointer B
1F0	R/W	CHI Size
1F4	W	CHI Receive Start
1F8	W	CHI Transmit Start
1FC	R/W	CHI Holding Register

5.3 SIU Registers

5.3.1 SIU Test Register

OFFSET = $1C8:

Bit	Label	RESET	Read/Write
31-8	Reserved		
7	ENDMATEST	0	R/W
6	ENNOTIMETEST	0	R/W
5-0	DMATESTWR(5:0)	0	R/W

ENDMATEST:
This bit is used for IC testing and should never be set.

ENNOTIMETEST:
This bit is used for IC testing and should never be set.

DMATESTWR(5:0):
These bits are used for IC testing and should never be set.

Chapter 6 Clock Module

PR31700 V0.3

This section describes the Clock Module, which contains logic for rate generation and enabling of the clocks for all other PR31700 modules.

6.1 Overview

The Clock Module within the PR31700 contains logic for generating the clocks for all other internal PR31700 modules, as well as for generating certain externally driven PR31700 output clocks. The Clock Module contains dividers for generating the correct rates for each clock and also contains logic for independently enabling or disabling individual clocks under software control.

6.1.1 Related Pins

SYSCLKIN: INPUT

This pin should be connected along with SYSCLKOUT to an external crystal which is the main PR31700 clock source.

SYSCLKOUT: OUTPUT

This pin should be connected along with SYSCLKIN to an external crystal which is the main PR31700 clock source.

6.2 Implementation

6.2.1 Block Diagram

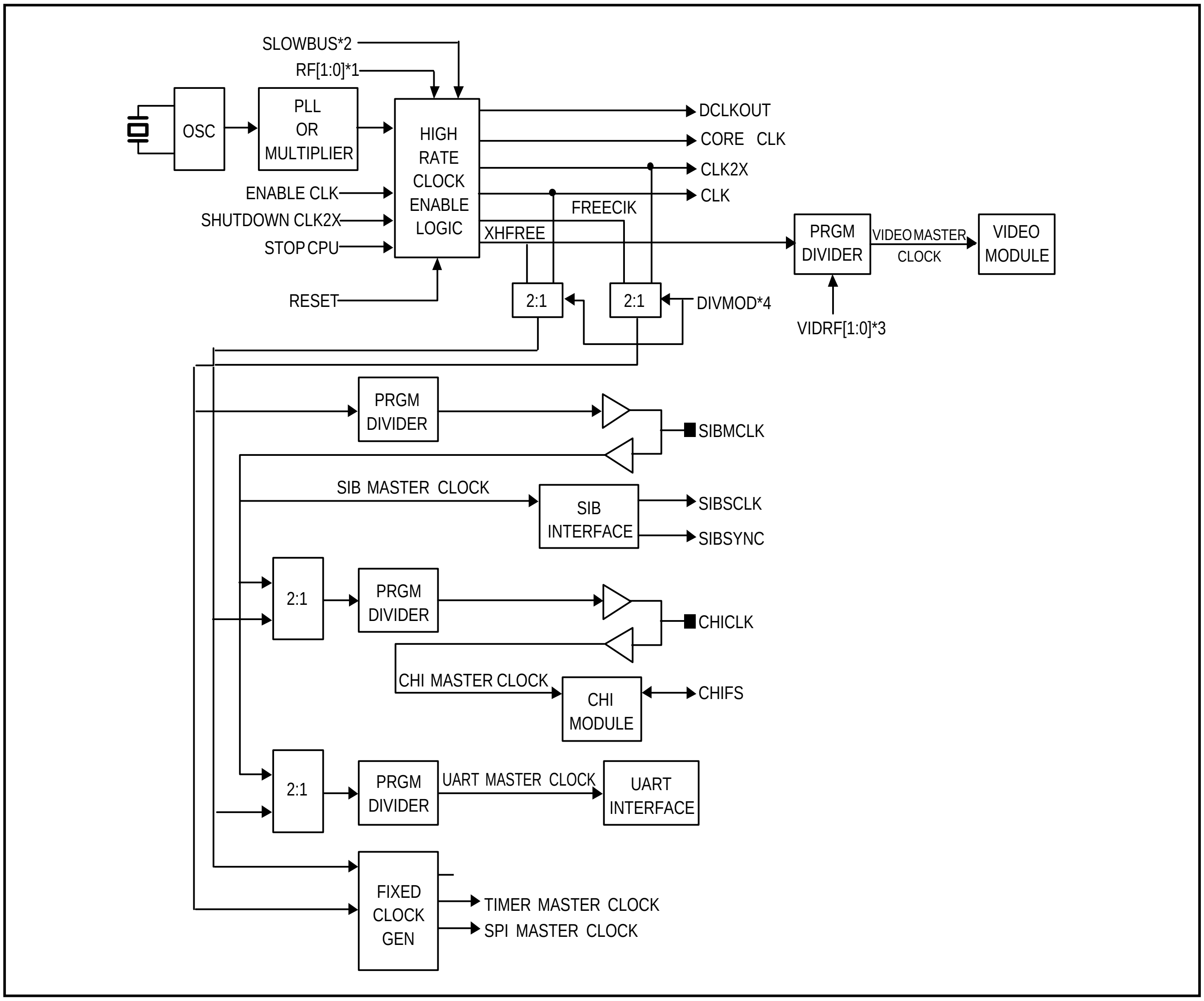


Figure 6-1. Clock Module Block Diagram

*1 : RF[1:0] is defined in the Config. Register in PR3910 Core
*2 : SLOWBUS is defined in the Power Control Register
*3 : VIDREF[1:0] is defined in the Power Control Register
*4 : DIVMOD is defined in the Power Control Register

Note: As shown in this block diagram, when DIVMOD bit is on default state (low); CLK2X and CLK are selected as input clocks for each function module such as UART and Timer etc. which would affect by changing of RF [1:0] Config. Register. If this bit is set; and then, FREECLK and XHFREE which are independent of RF[1:0] configuration, will be selected as clock sources for these module.

6.2.2 Clock Module Description

Figure 6-1 shows a block diagram of the Clock Module. The main PR31700 clock source is from an external crystal, connected to the PR31700 via the SYSCLKIN/SYSCLKOUT pins. The PR31700 contains an oscillator which is internally connected to these pins. The output of the oscillator feeds a phase-locked-loop (or multiplier) circuit which takes the lower rate oscillator output and generates the higher-rate clocks required by the PR31700. For example, if the external SYSCLK crystal rate is 9.216 MHz, the PLL multiplier circuit would perform a 8x rate multiplication to generate a high-speed core clock rate of 73.728 MHz.

The main high rate clocks generated by the Clock Module are the following:

- CLK - one-half the rate of CLK2X; used as the master clock for many of the PR31700 modules and for generating other lower rate clocks
- CLK2X - highest rate clock and 2x the rate of CLK; used for generating other lower rate clocks and for certain memory interface circuits in the BIU
- CORE CLOCKS – same rate of CLK2X (highest rate clock); typically consists of multi-phase non-overlapping clocks (vendor-dependent implementation); used as master clocks for CPU core
- DCLKOUT - same rate as CLK2X; used as high-speed externally driven clock for SDRAMs
- XHFREE – independent clock input for video master clock.

In the R3000 MIP core, the core clock can be dynamically switched , º, 1/8 of its normal operating frequency by setting the Reduced Frequency (RF) mode bits in the Config Register in the Core. This events is initiated by external logic which sets the RF bits in the Config register via software. As core clock switches, the rest of internal system clocks such as CLK, CLK2X, DCLKOUT also switches as accordingly. The following table is for relationship between RF[1:0] vs. system clocks.

The enhancement has been made to PR31700 for reducing power consumption in Idle mode by providing further division of clock. The extra control bit “SLOWBUS” has been added to Power Control Register (OFFSET = $ 1C4).

RF[1:0]	SLOWBUS	CORECLK	DCLKOUT	CLK2X	FREECLK	CLK	XHFREE
00	0	F	F	F	F	F/2	F/2
01	0	F/2	F/2	F/2	F	F/4	F/2
10	0	F/4	F/4	F/4	F	F/8	F/2
11	0	F/8	F/8	F/8	F	F/16	F/2
00	1	F/2	F/2	F/2	F	F/4	F/2
01	1	F/4	F/4	F/4	F	F/8	F/2
10	1	F/8	F/8	F/8	F	F/16	F/2
11	1	F/16	F/16	F/16	F	F/32	F/2

The CLK2X and CLK signals are used to generate the clocks for all other PR31700 modules except for the video master clock.

Video master clock is generated from XHFREE clock, independent of DIVMOD bit in Power Control Register (OFFSET = $1C4). The XHFREE clock is not affected by switching RF[1:0] bits, constant one-half rate of highest clock rate will be fed into the video master clock. This video master clock rate is only controlled by VIDRF[1:0]. (Refer Power Control Register (offset = $ 1C4) for VIDRF[1:0] bit description).

The SIBMCLK pin can be configured as an output. In this mode, SIB master clock is generated internally by dividing down CLK2X/FREE clock with the internal programmable divider. The SIBMCLK pin can also be configured as input. In this mode, all SIB clocks are derived from an external SIBMCLK oscillator source, which is asynchronous with respect to CLK. This mode allows optional decoupling of the SIB (as well as the CHI and UART) rates from the CPU core clock rate. The selected SIBMCLK source is then used as the SIB Master Clock for the SIB Module circuits, and is also used to generate the SIBSCLK and SIBSYNC externally driven PR31700 output signals.

The CHICLK pin can be configured as an output, for which the programmable rate is generated by dividing down from either CLK/XHFREE or the SIB Master Clock. The CHICLK pin can also be configured as input. In this mode, all CHI clocks are derived from an external peripheral source and the CHI Module will slave to this external clock. The selected CHICLK source is used as the CHI Master Clock for the CHI Module circuits, and is also used to generate the CHIFS externally driven PR31700 output signal.

The programmable UART Master Clock is generated by dividing down from either CLK/XHFREE or the SIB Master Clock, and is used as the master clock for the baud generator circuit within each UART Module.

The Clock Module also contains several fixed dividers for generating the master clocks for the RTC timer, and SPI circuits. These clocks are divided down from either CLK/XHFREE or CLK2X/FREE.

6.3 Clock Registers

6.3.1 Clock Control Register

OFFSET = $1C0:

Bit	Label	RESET	Read/Write
31-24	CHICLKDIV(7:0)	0	R/W
23	ENCLKTEST	0	R/W
22	CLKTESTSELSIB	0	R/W
21	CHIMCLKSEL	0	R/W
20	CHICLKDIR	0	R/W
19	ENCHIMCLK	0	R/W
18	ENVIDCLK	0	R/W
17	ENMBUSCLK	0	R/W
16	ENSPICLK	0	R/W
15	ENTIMERCLK	0	R/W
14	ENFASTTIMERCLK	0	R/W
13	SIBMCLKDIR	0	R/W
12	Reserved		
11	ENSIBMCLK	0	R/W
10-8	SIBMCLKDIV(2:0)	0	R/W
7	CSERSEL	1	R/W
6-4	CSERDIV(2:0)	0	R/W
3	ENCSERCLK	0	R/W
2	ENIRCLK	0	R/W
1	ENUARTACLK	0	R/W
0	ENUARTBCLK	0	R/W
C	HICLKDIV(7:0):		

These bits select the start count value and the stop count value for the 8-bit programmable counter used to generate CHICLK, which is derived by dividing down from either SIBMCLK or CLK; CHICLK is the 1X or 2X bit clock used for the CHI Module and is generated by the PR31700 whenever the CHI is in master mode. Since the MSB of the counter output is used for CHICLK, the start count and stop count values are chosen to provide (as close as possible) a 50% duty cycle CHICLK. The counter divide modulus is equal to (CHICLKDIV+2). The equations used to compute these counter start and stop values are as follows:

start count value = 127 - ((CHICLKDIV + 1) >> 1)

stop count value = 128 + (CHICLKDIV >> 1)

For example, if CHICLKDIV = 7, then the start count value = 123 and the stop count value = 131, resulting in a divide-by-9 counter.

ENCLKTEST:
This bit is used for IC testing and should not be set.

CLKTESTSELSIB:
This bit is used for IC testing and should not be set.

CHIMCLKSEL:
Setting this bit to a logic "1" selects the external SIBMCLK source to be the CHI master clock. Clearing this bit to a logic "0" selects an internal clock source (of rate equal to CLK) to be the CHI master clock.

CHICLKDIR:
This bit controls the direction of the CHICLK pin. Setting this bit to a logic "1" configures CHICLK to be an output (CHI master mode). Clearing this bit to a logic "0" configures CHICLK to be an input (CHI slave mode).

ENCHIMCLK:
This bit is used to enable or disable the CHICLK counter and CHICLK clock generation. Setting this bit to a logic "1" enables the CHICLK counter and CHICLK generation. Clearing this bit to a logic "0" disables the CHICLK counter and CHICLK generation, halting the clock to the CHI Module in order to reduce power consumption. The CHICLK counter is a programmable 8-bit divider.

ENVIDCLK:
This bit is used to enable or disable the video master clock. Setting this bit to a logic "1" enables the VIDCLK. Clearing this bit to a logic "0" disables the VIDCLK, halting the clock to the Video Module in order to reduce power consumption.

ENMBUSCLK:
This bit is used to enable or disable the MBus master clock. Setting this bit to a logic "1" enables the MBUSCLK. Clearing this bit to a logic "0" disables the MBUSCLK, halting the clock to the MBus Module in order to reduce power consumption.

ENSPICLK:
This bit is used to enable or disable the SPICLK counter and SPICLK clock generation. Setting this bit to a logic "1" enables the SPICLK counter and SPICLK clock generation. Clearing this bit to a logic "0" disables the SPICLK counter and SPICLK clock generation, halting the clock to the SPI Module in order to reduce power consumption. The SPICLK counter is a fixed divide-by-5 of CLK.

ENTIMERCLK:
This bit is used to enable or disable the RTC periodic timer clock counter. Setting this bit to a logic "1" enables the timer clock counter. Clearing this bit to a logic "0" disables the timer clock counter. The timer clock counter is a fixed divide-by-32 of CLK.

ENFASTTIMERCLK:
This bit is used for IC testing and should not be set.

SIBMCLKDIR:
This bit controls the direction of the SIBMCLK pin. Setting this bit to a logic "1" configures SIBMCLK to be an output. In this SIB master mode, the SIB clocks are slaved to CLK2X. Clearing this bit to a logic "0" configures SIBMCLK to be an input. In this SIB slave mode, the SIB clocks are derived from an external SIBMCLK oscillator source, which is asynchronous with respect to CLK2X.

ENSIBMCLK:
This bit is used to enable or disable the SIBMCLK counter and SIBMCLK clock generation. Setting this bit to a logic "1" enables the SIBMCLK counter and SIBMCLK generation. Clearing this bit to a logic "0" disables the SIBMCLK counter and SIBMCLK generation, halting the clock to the SIB Module in order to reduce power consumption. The SIBMCLK counter is a programmable 3-bit divider.

SIBMCLKDIV(2:0):
These bits select the start count value and the stop count value for the 3-bit programmable counter used to generate SIBMCLK, which is derived by dividing down CLK2X; SIBMCLK is the master clock used for the SIB Module and is generated by thr PR31700 whenever the SIB is in SIB master mode. Since the MSB of the counter output is used for SIBMCLK, the start count and stop count values are chosen to provide (as close as possible) a 50% duty cycle SIBMCLK. The table used to compute these counter start and stop values are as follows:

SIBMCLKDIV	start value	stop value	divide-modulus
0	3	4	2
1	2	4	3
2	2	5	4
3	1	5	5
4	1	6	6
5	0	6	7
6	0	7	8

CSERSEL:
Setting this bit to a logic "1" selects the external SIBMCLK source to be the CSERCLK master clock. Clearing this bit to a logic "0" selects an internal clock source (of rate equal to CLK2X   4) to be the CSERCLK master clock.

CSERDIV(2:0):
These bits select the start count value and the stop count value for the 3-bit programmable counter used to generate IRCLK, UARTACLK, and UARTBCLK; these clocks are derived by dividing down CSERCLK. Since the MSB of the counter output is used for these clocks, the start count and stop count values are chosen to provide (as close as possible) a 50% duty cycle for these clocks. The table used to compute these counter start and stop values are as follows:

CSERDIV	**start value**	**stop value**	**divide-modulus**
0	3	4	2
1	2	4	3
2	2	5	4
3	1	5	5
4	1	6	6
5	0	6	7
6	0	7	8

ENCSERCLK:
This bit is used to enable or disable the CSERCLK counter. Setting this bit to a logic "1" enables the CSERCLK counter. Clearing this bit to a logic "0" disables the CSERCLK counter. The CSERCLK counter is a programmable 3-bit divider.

ENIRCLK:
This bit is used to enable or disable the IRCLK clock generation. Setting this bit to a logic "1" enables the IRCLK generation. Clearing this bit to a logic "0" disables the IRCLK generation, halting the clock to the IR Module in order to reduce power consumption.

ENUARTACLK:
This bit is used to enable or disable the UARTACLK clock generation. Setting this bit to a logic "1" enables the UARTACLK generation. Clearing this bit to a logic "0" disables the UARTACLK generation, halting the clock to the UARTA Module in order to reduce power consumption.

ENUARTBCLK:
This bit is used to enable or disable the UARTBCLK clock generation. Setting this bit to a logic "1" enables the UARTBCLK generation. Clearing this bit to a logic "0" disables the UARTBCLK generation, halting the clock to the UARTB Module in order to reduce power consumption.

Chapter 7 CHI Module

This section describes the Concentration Highway Interface (CHI) Module, which contains logic for interfacing to external full-duplex serial time-division-multiplexed (TDM) communication peripherals.

7.1 Overview

The CHI Module within the PR31700 contains holding registers, shift registers, DMA support, and other logic to support interfacing to external full-duplex serial TDM communication peripherals, including ISDN communication devices and PCM/TDM serial highways. The PR31700 implementation of the CHI Module is based on the Concentration Highway Interface standard specified by Intel® and AT&T®, which is intended to allow glueless interface to various TDM highways used by numerous commercial products.

The PR31700 CHI Module utilizes a 4-signal interface consisting of clock, sync, transmit serial data, and receive serial data. The data is organized as a TDM format, with up to 64 timeslots (nominally 8 bits each) per frame, with a nominal frame rate of 8 KHz (8 bits x 8 KHz = 64 Kbps nominal data rate per channel). The number of timeslots and the data rate (up to 4.096 Mbps) and frame rate are programmable, providing flexibility for supporting various TDM communication peripherals. These timeslots are commonly used to carry voice, data, or control and status information.

The CHI Module provides independent DMA support for receive and transmit (two total independent DMA channels). The DMA buffers can be configured in a continuous (circular) buffer mode or a one-time (empty or fill, then stop) buffer mode. Half-buffer and end-of-buffer DMA address counter interrupts are available, allowing the CPU to minimize overhead and efficiently empty or fill half of the DMA buffer in a ping-pong fashion. The DMA buffer size is programmable (up to a maximum of 16 KBytes) and the receive and transmit buffers can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation). Also available is a direct CPU read/write mode for bypassing the DMA, allowing the CPU to read or write the CHI data on a sample by sample basis, if so desired.

7.1.1 Related Pins

CHIFS: **INPUT/OUTPUT**

This pin is the CHI frame synchronization signal. This pin is available for use in one of two modes. As an output, this pin allows the PR31700 to be the master CHI sync source. As an input, this pin allows an external peripheral to be the master CHI sync source and the PR31700 CHI Module will slave to this external sync.

CHICLK: **INPUT/OUTPUT**

This pin is the CHI clock signal. This pin is available for use in one of two modes. As an output, this pin allows the PR31700 to be the master CHI clock source. As an input, this pin allows an external peripheral to be the master CHI clock source and the PR31700 CHI Module will slave to this external clock.

CHIDOUT: **OUTPUT**

This pin is the CHI serial data output signal.

CHIDIN: **INPUT**

This pin is the CHI serial data input signal.

7.2 Interface Requirements

7.2.1 Frame Structure and Serial Timing

Each CHI frame (nominally 8 KHz rate) is time-division-multiplexed into several timeslots or channels. The total number of timeslots per frame is programmable, with a maximum of 64 timeslots allowed, and the number of timeslots is also restricted to an even number. Each timeslot is 8 bits, although 16-bit or 32-bit channels can be supported by accessing adjacent timeslots.

The PR31700 CHI Module supports a master or slave mode for both the clock (CHICLK) and sync (CHIFS). For the master mode, the PR31700 contains programmable dividers for generating the clock and/or sync signal, synchronously dividing down from the main core clock (CLK). For the slave mode, the PR31700 accepts external clock and/or sync signals and utilizes "digital-PLL" type circuitry to stayed "locked" to the external source. The CHI Module supports the following programmable features which allow support for various clock and sync timing formats:

- 1x versus 2x clock modes for CHICLK (2x clock mode uses two CHICLK periods per data bit)
- MSB-first versus LSB-first serial formats for transmit and receive
- rising versus falling edge (polarity) used for frame sync triggering
- CHIFS signal can be sampled on either rising or falling edge of CHICLK
- CHIDIN receive data can be sampled on either rising or falling edge of CHICLK
- CHIDOUT transmit data can be pushed on either rising or falling edge of CHICLK
- CHIDIN receive data can have programmable bit offset (timeslot 0 offset from CHIFS)
- CHIDOUT transmit data can have programmable bit offset (timeslot 0 offset from CHIFS)
- CHIDOUT transmit data (tri-state) output buffer enable is dynamically asserted for only active timeslots; for sleep mode CHIDOUT is always tri-stated
- for CHIFS master mode, the CHIFS pulse width and polarity is programmable

The PR31700 CHI Module allows for a programmable bit offset for both CHIDIN and CHIDOUT, which is related to the number of clock cycles between the start of timeslot 0 and CHIFS. This flexibility allows the PR31700 CHI Module to support a wide variety of interface clock and sync timing formats. The control bits for controlling the CHIDIN bit offset are CHIRXBOFF[3:0], while the control bits for controlling the CHIDOUT bit offset are CHITXBOFF[3:0].

Table 7-1 shows a summary matrix for the values of CERX and CETX for all possible settings of CHIRXBOFF and CHITXBOFF, respectively. These values are shown for various configurations of CHICLK mode (1x versus 2x), CHIFSEDGE, CHIRXEDGE, and CHITXEDGE. The CHIFSEDGE settings determine whether to use the rising edge (CHIFSEDGE = 1) or falling edge (CHIFSEDGE = 0) of CHICLK to sample CHIFS. The CHIRXEDGE settings determine whether to use the rising edge (CHIRXEDGE = 1) or falling edge (CHIRXEDGE = 0) of CHICLK to sample the CHIDIN input. The CHITXEDGE settings determine whether to use the rising edge (CHITXEDGE = 1) or falling edge (CHITXEDGE = 0) of CHICLK to push the CHIDOUT output. CERX is defined as the number of CHICLK clock edges (rising and falling) between the edge where CHIFS is sampled and the edge where CHIDIN is sampled. CETX is defined as the number of CHICLK clock edges (rising and falling) between the edge where CHIFS is sampled and the edge where CHIDOUT is pushed. The CHI frame structure and bit offsets are shown in Figure 7-1 for various clock and sync configurations.

Table 7-1a. CERX Values for CHIRXBOFF Versus Clock and Edge Configurations

CHIRXBOFF																		
chiclk mode	**chifsedge**	**chirxedge**	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1x	0	0	0		2		4		6		8		10		12		14	
1x	1	1	0		2		4		6		8		10		12		14	
1x	0	1	-1		1		3		5		7		9		11		13	
1x	1	0	-1		1		3		5		7		9		11		13	
2x	0	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32
2x	1	1	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32
2x	0	1	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31
2x	1	0	1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31

Table 7-1b. CETX Values for CHITXBOFF Versus Clock and Edge Configurations

CHITXBOFF																		
chiclk mode	**chifsedge**	**chirxedge**	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1x	0	0	-2		0		2		4		6		8		10		12	
1x	1	1	-2		0		2		4		6		8		10		12	
1x	0	1	-1		1		3		5		7		9		11		13	
1x	1	0	-1		1		3		5		7		9		11		13	
2x	0	0	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
2x	1	1	0	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30
2x	0	1	-1	1	3	5	7	9	11	13	15	17	19	21	23	25	27	5
2x	1	0	-1	1	3	5	7	9	11	13	15	17	19	21	23	25	27	5

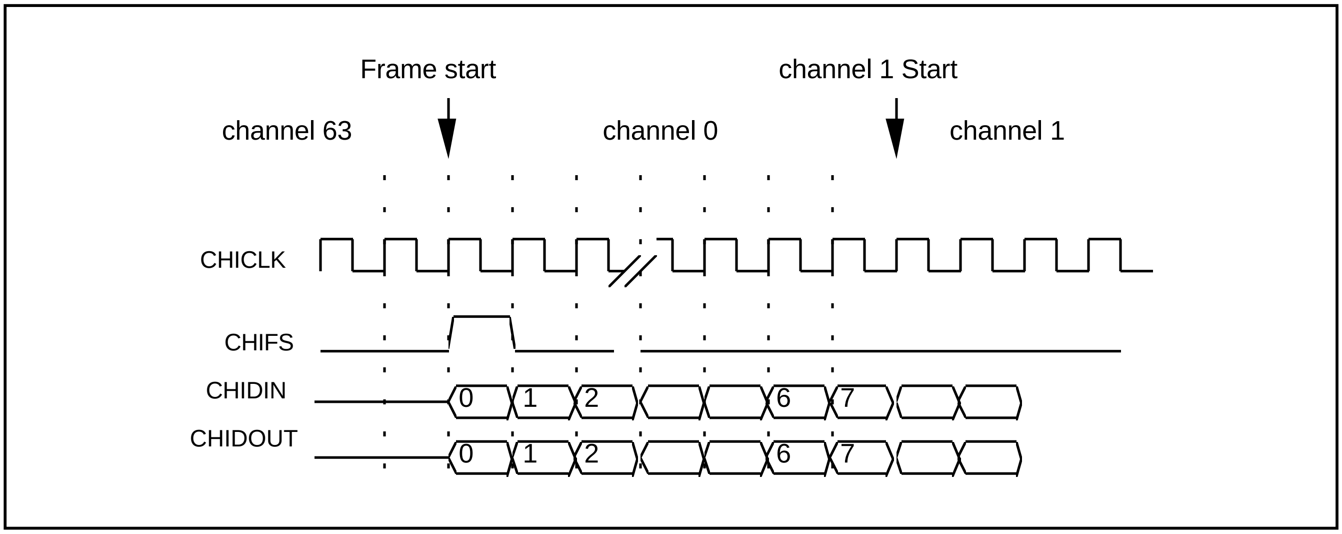


Figure 7-1a. CHI Frame Structure Example

CHICLK 1X mode
CHIFS sampled on falling edge
CHIDIN sampled on falling edge; RXBOFF = 0; CERX = 1
CHIDOUT pushed on rising edge; TXBOFF = 0; CETX = -1

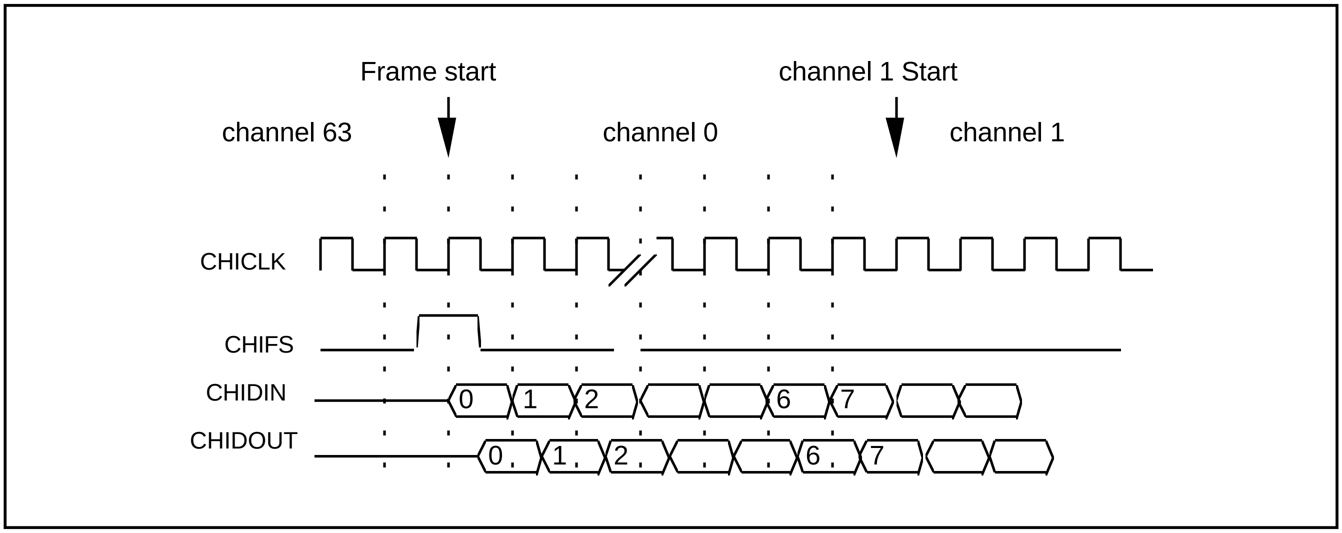


Figure 7-1b. CHI Frame Structure Example

CHICLK 1X mode
CHIFS sampled on rising edge
CHIDIN sampled on falling edge; RXBOFF = 2; CERX = 1
CHIDOUT pushed on falling edge; TXBOFF = 2; CETX = 1

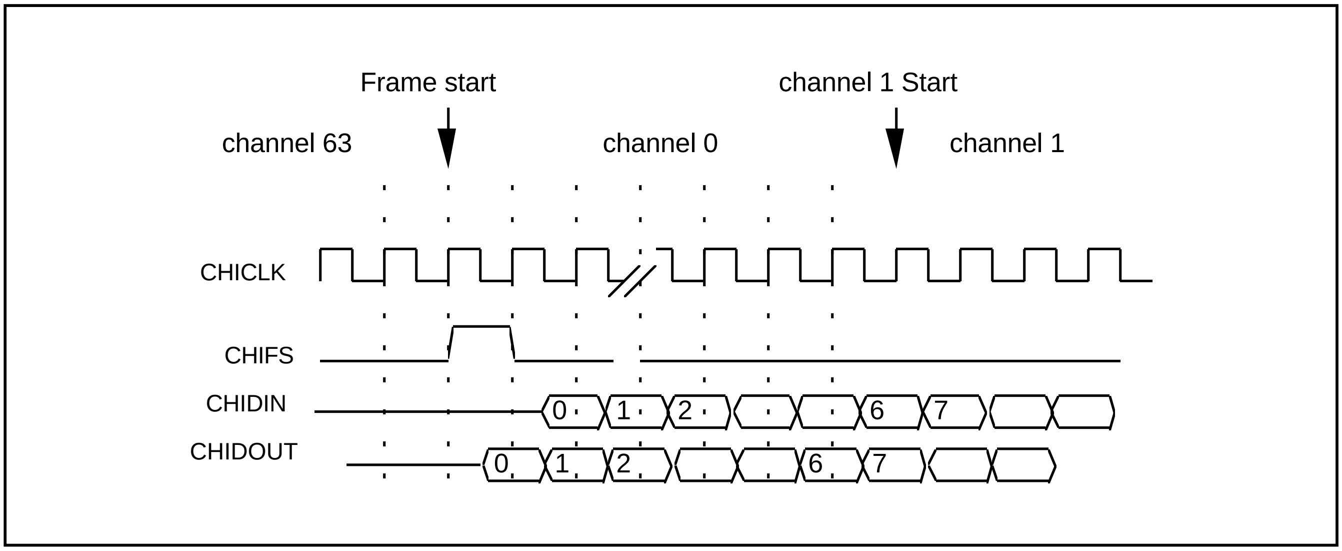


Figure 7-1c. CHI Frame Structure Example

CHICLK 1X mode
CHIFS sampled on falling edge
CHIDIN sampled on rising edge; RXBOFF = 4; CERX = 3
CHIDOUT pushed on falling edge; TXBOFF = 2; CETX = 0

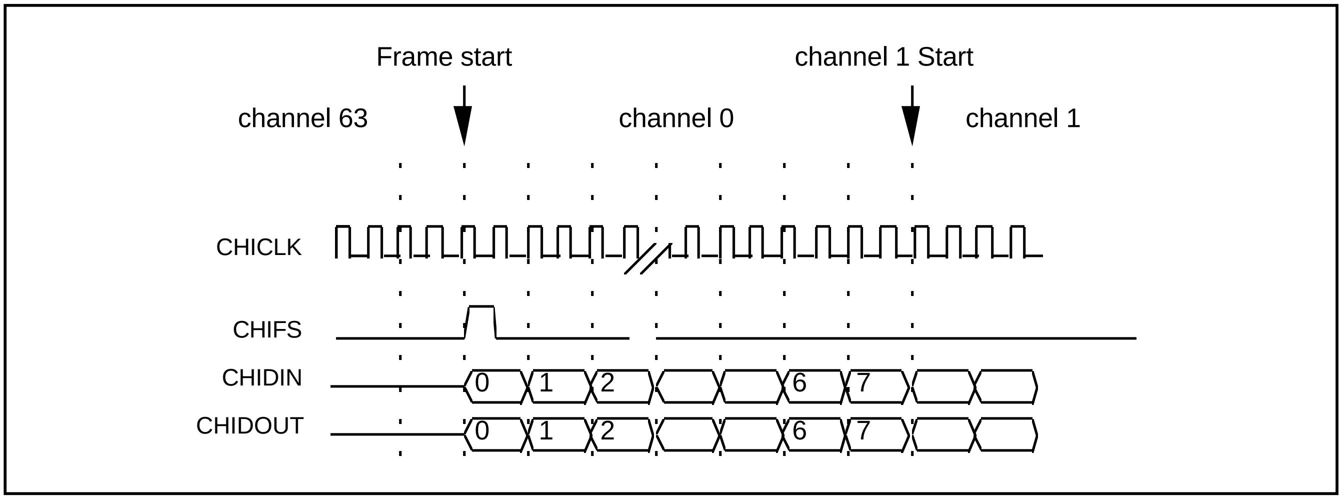


Figure 7-1d. CHI Frame Structure Example

CHICLK 2X mode
CHIFS sampled on falling edge
CHIDIN sampled on rising edge; RXBOFF = 0; CERX = 1
CHIDOUT pushed on rising edge; TXBOFF = 0; CETX = -1

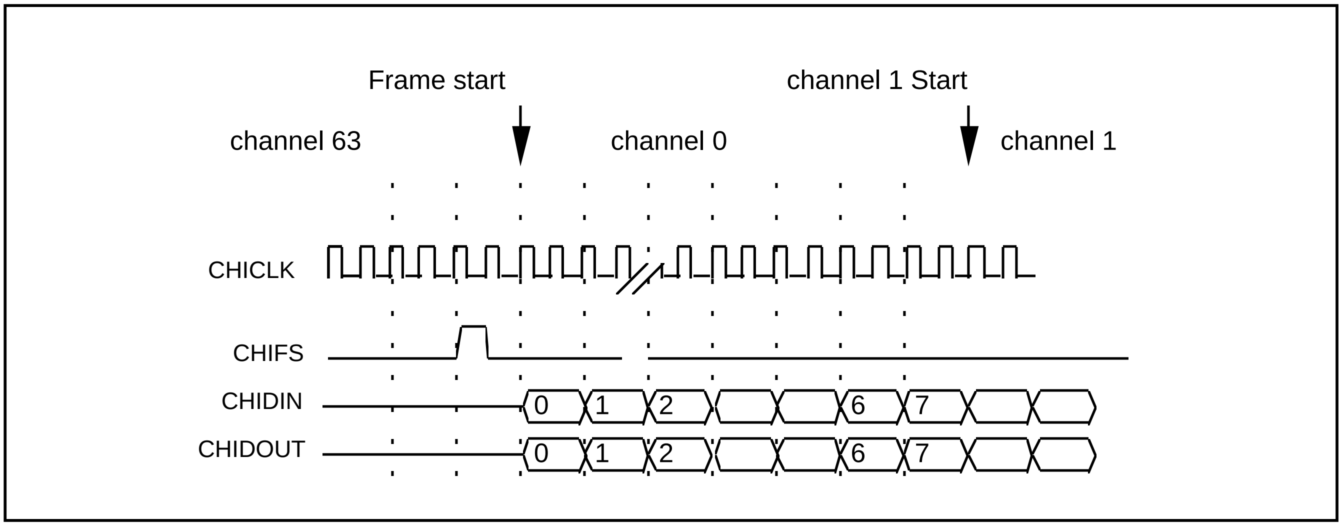


Figure 7-1e. CHI Frame Structure Example

CHICLK 2X mode
CHIFS sampled on falling edge
CHIDIN sampled on rising edge; RXBOFF = 2; CERX = 5
CHIDOUT pushed on rising edge; TXBOFF = 2; CETX = 3

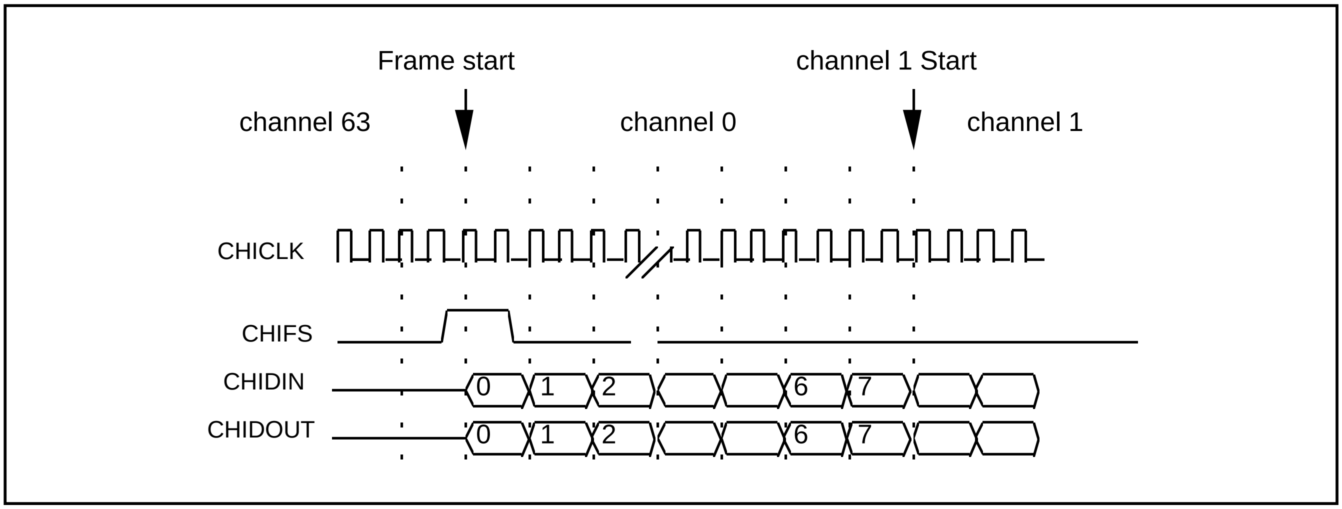


Figure 7-1f. CHI Frame Structure Example

CHICLK 2X mode
CHIFS sampled on rising edge
CHIDIN sampled on rising edge; RXBOFF = 0; CERX = 2
CHIDOUT pushed on rising edge; TXBOFF = 0; CETX = 0

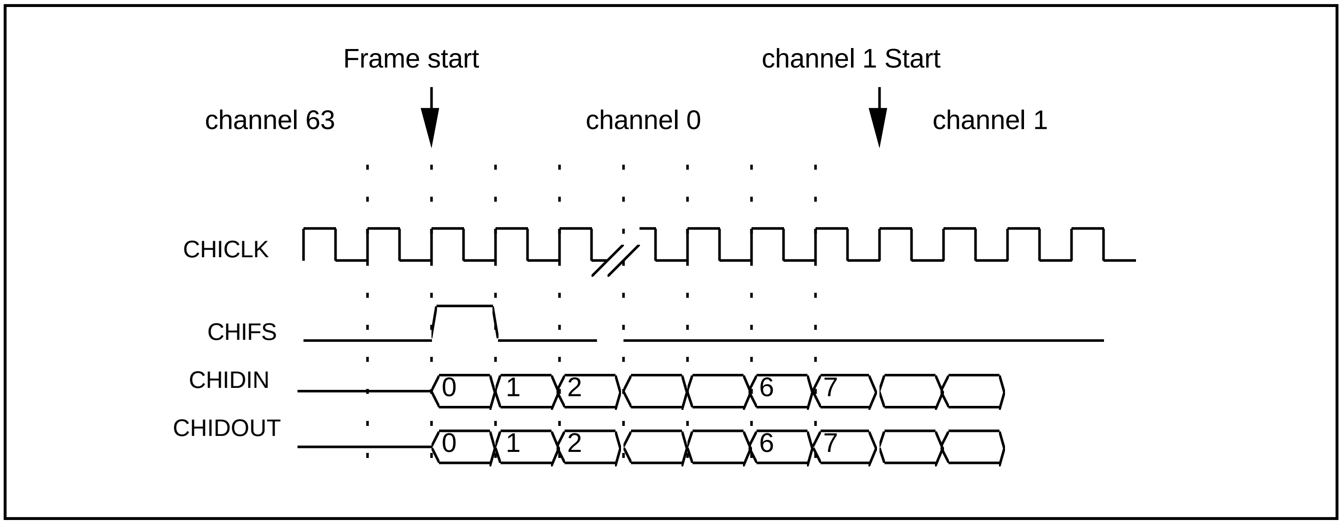


Figure 7-1g. CHI Frame Structure Example

CHICLK 2X mode
CHIFS sampled on falling edge
CHIRXFSPOL = 1 (negative polarity)
CHIDIN sampled on rising edge; RXBOFF = 1; CERX = 3
CHIDOUT pushed on rising edge; TXBOFF = 1; CETX = 1

7.2.2 Configurations

The programmability of the clock, sync, bit offsets, and number of timeslots allows the CHI Module to support a wide range of configurations. Several of these configurations are commonly utilized as communication interfaces by numerous commercial products, some of which are briefly discussed below. Other configurations are available for application-specific usage.

The K2 Interface is an AT&T standard used as a serial inter-chip digital interface between U-interface transmission circuits and various system-wide interface circuits. The K2 Interface utilizes four pins (clock, sync, data in, and data out) and consists of eight 8-bit timeslots, with a frame rate of 8 KHz and a clock rate of 512 KHz. The SLD Interface utilizes 3 pins (clock, sync, data), with the transmit and receive pins tied together, such that the data is sent bi-directionally in a ping-pong fashion.

The SLD Interface consists of eight 8-bit timeslots, with 4 timeslots reserved for transmit and 4 timeslots reserved for receive. The frame rate is 8 KHz and the clock rate is 512 KHz.

The GCI Interface is a 4-pin variable-speed TDM highway utilizing anywhere from 4 to 48 timeslots. The frame rate is 8 KHz and the clock rate is 2x the data rate and varies from 512 KHz to 6.144 MHz.

The IOM-2 Interface is commonly used to interface to 4-pin ISDN line interface drivers. Each frame consists of twelve 8-bit timeslots, with a frame rate of 8 KHz and a (2x) clock rate of 1.536 MHz.

The CEPT Level-1 PCM Format is a common communications standard used for digital transmission of voice and data. Each frame consists of 32 8-bit timeslots, with a frame rate of 8 KHz and a (1x) clock rate of 2.048 MHz.

Table 7-2 shows a summary matrix of several example CHI configurations and their associated parameters.

Table 7-2. Example CHI Configurations

chiclk divide	timeslots	chiclk mode	fs	chiclk rate	data rate	comments
4.5	64	2x	8 KHz	8.192 MHz	4.096Mbps	slave mode only
6	48	2x	8 KHz	6.144 MHz	3.072Mbps	GCI format
9	64	1x	8 KHz	4.096 MHz	4.096Mbps	CHI format
9	8	1x	64 KHz	4.096 MHz	4.096Mbps	hi-speed mode
10	64	1x	7.2 KHz	3.6864 MHz	3.6864Mbps	
12	24	2x	8 KHz	3.072 MHz	1.536Mbps	
12	48	1x	8 KHz	3.072 MHz	3.072Mbps	
18	32	1x	8 KHz	2.048 MHz	2.048Mbps	CEPT PCM format
24	12	2x	8 KHz	1.536 MHz	768Kbps	IOM-2 format
72	8	1x	8 KHz	512 KHz	512 Kbps	K2, SLD formats

(table values based on CLK = 36.864 MHz)

7.3 Implementation

7.3.1 Block Diagram

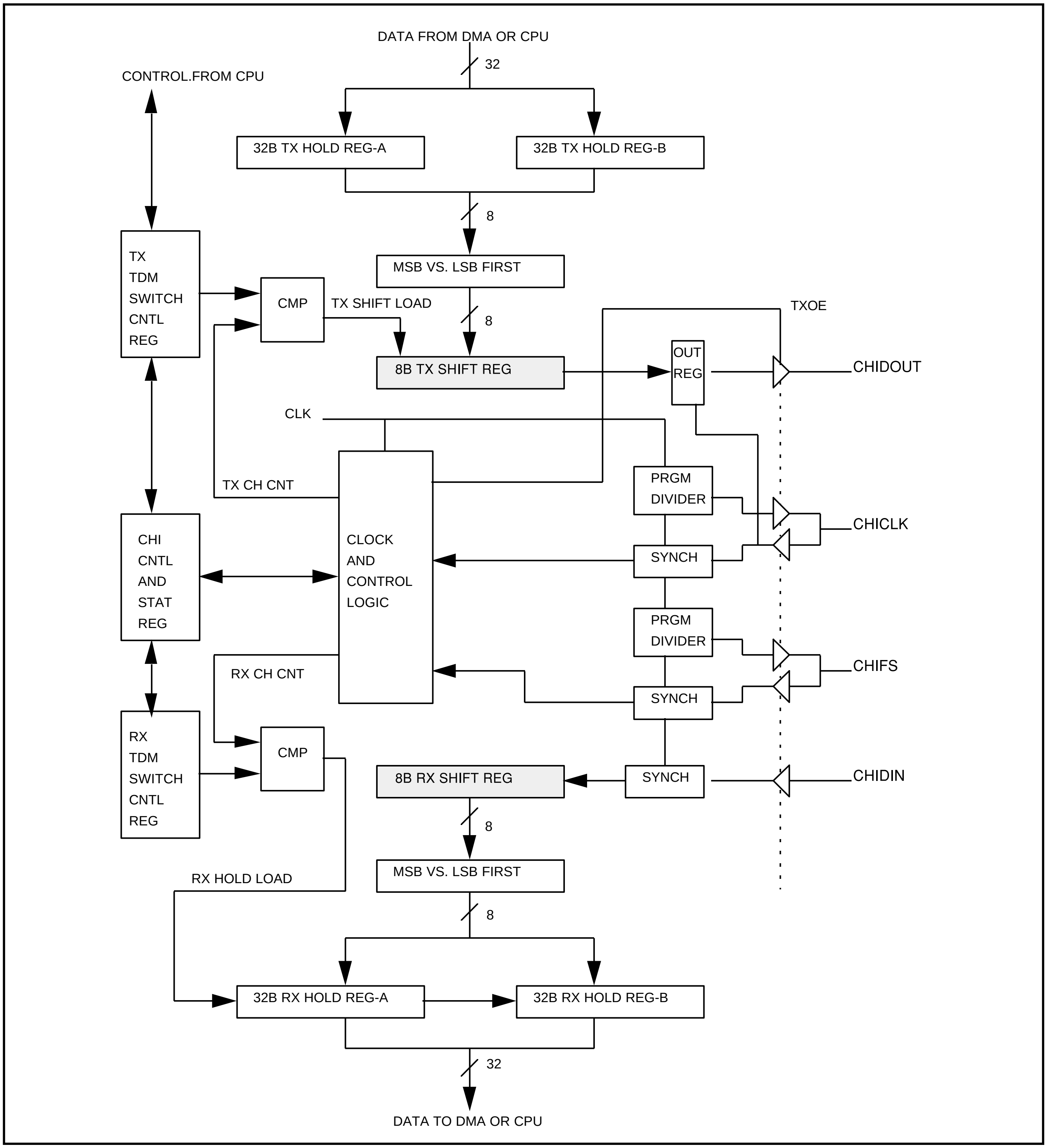


Figure 7-2. CHI Module Block Diagram

The CHI Module consists of holding registers (both transmit and receive), shift registers (both transmit and receive), DMA support, and other logic to support interfacing to various types of TDM highways. See Figure 7-2 for a block diagram of the CHI Module.

7.3.2 Transmitter

For the CHI transmit direction, Buffer-A and Buffer-B transmit holding registers are written either from the DMA circuit or directly from the CPU. Each of these 2 holding registers are 32 bits wide, and CHI control logic determines which byte from which holding register gets loaded at a given time into the 8-bit transmit shift register. In addition, the byte data loaded from the holding register to the shift register can be MSB-first or LSB-first. The reason for having Buffer-A and Buffer-B holding registers is that the CHI Module operates in a ping-pong fashion. Each frame of data is partitioned into 2 buffers (A and B); for example, with 64 timeslots total, the data is partitioned into 32 timeslots per buffer. The ping-pong operation allows one buffer to be updated (via the DMA or CPU) while the other buffer is being loaded into the shift register a byte at a time, depending on which timeslots are active. The ping-pong operation is transparent to the CPU or DMA interface, since the CHI Module automatically points to the correct A or B buffer at a given time and the CPU or DMA always accesses the same 32-bit holding register for all transactions.

The transmit TDM switch control register is used to select ANY 4 channels per buffer to be loaded from the holding register to the shift register. For example, if the CHI Module is configured for 32 timeslots per buffer (64 total timeslots), any 4 channels per buffer (8 total) can be selected out of the 32 available channels. The CHIDOUT signal is tri-stated during any of the non-selected channels. Each of the 8 selected channels also has an individual control bit for enabling/disabling the timeslot.

7.3.3 Receiver

For the CHI receive direction, Buffer-A and Buffer-B receive holding registers are read either by the DMA circuit or directly by the CPU. Each of these 2 holding registers are 32 bits wide, and CHI control logic determines which byte to which holding register gets loaded at a given time from the 8-bit receive shift register. In addition, the byte data loaded from the shift register to the holding register can be MSB-first or LSB-first. Similar to the transmit direction, the receive section also operates in a ping-pong fashion, allowing one buffer to be read (via the DMA or CPU) while the other buffer is being loaded from the shift register a byte at a time, depending on which timeslots are active.

The receive TDM switch control register (independent from the transmit TDM switch control register) is used to select ANY 4 channels per buffer to be loaded from the shift register to the holding register. Each of the 8 selected channels also has an individual control bit for enabling/disabling the timeslot.

An interrupt is available whenever a valid longword is available from the receive data holding register A. This also means a valid CHI output sample can be written to the transmit data holding register B. Similarly, an interrupt is also available whenever a valid longword is available from the receive data holding register B. This also means a valid CHI output sample can be written to the transmit data holding register A.

7.3.4 Clock and Control Generation

The CHI Module contains several programmable counters which are used to generate the various CHI internal and external control signals and clocks. See Figure 7-3 for a block diagram of the CHI clock and control generation circuit. As mentioned previously, CHICLK can be configured as either an output (master mode) or input (slave mode). As an output, CHICLK is derived by dividing down from CLK. In this mode, all CHI clocks are then synchronously locked to the main PR31700 system clock. As an input, CHICLK is generated from an external clock source, which is asynchronous with respect to CLK. The PR31700 CHI Module utilizes a digital-PLL circuit to stayed "locked" to the external source, while still operating internally using CLK. CHIDIN and CHIDOUT are also synchronized between CLK and the externally-supplied CHICLK.

CHIFS can also be configured as either an output (master mode) or input (slave mode). As an output, CHIFS is derived by dividing down from CHICLK. For this mode, the CHIFS pulse width and polarity is also programmable. As an input, CHIFS is generated from an external sync source. The PR31700 CHI Module utilizes a digital-PLL circuit to stayed "locked" to the external sync source, while still operating internally using CLK.

The programmable receive and transmit sync delay counters shown in Figure 7-3 are used to implement the bit offset feature described earlier. The bit offset control bits determine the number of clock cycles between the start of timeslot 0 and CHIFS. The receive and transmit sync delay counters are independent from each other, such that the receive and transmit serial data streams can have different bit offsets.

The programmable receive channel counter output is constantly compared with the receive TDM switch control register values, and whenever a match occurs, the byte of data is loaded from the receive shift register into the correct field within the receive holding register. Similarly, the programmable transmit channel counter output is constantly compared with the transmit TDM switch control register values, and whenever a match occurs, the byte of data is loaded from the correct field within the transmit holding register into the receive shift register.

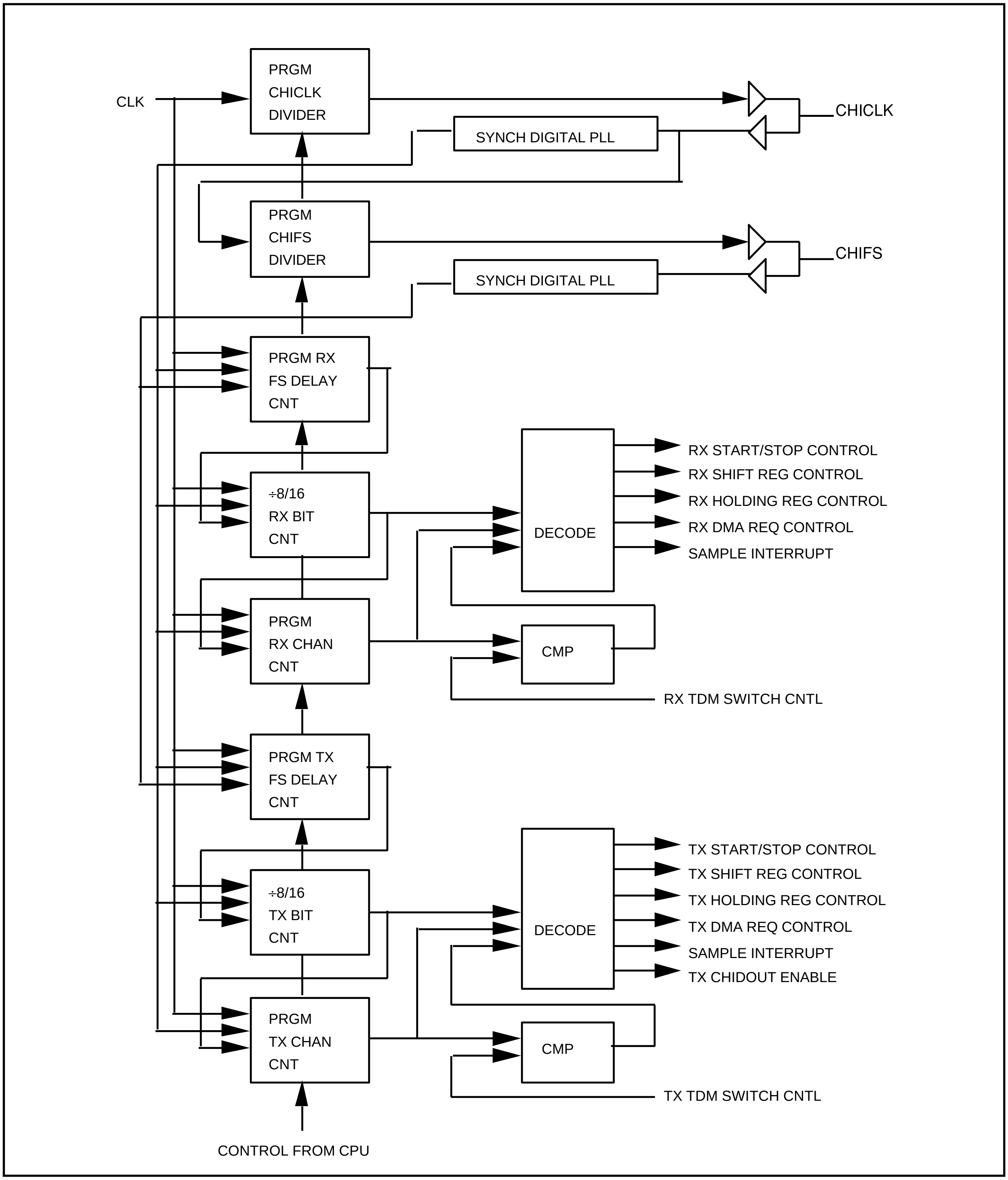


Figure 7-3. CHI Clock and Control Generation

7.3.5 DMA Address Generation

The CHI Module provides support for 2 independent DMA channels: receive and transmit. The circuit used to generate the DMA address, as well as half-buffer and end-of-buffer interrupts is shown in Figure 7-4.

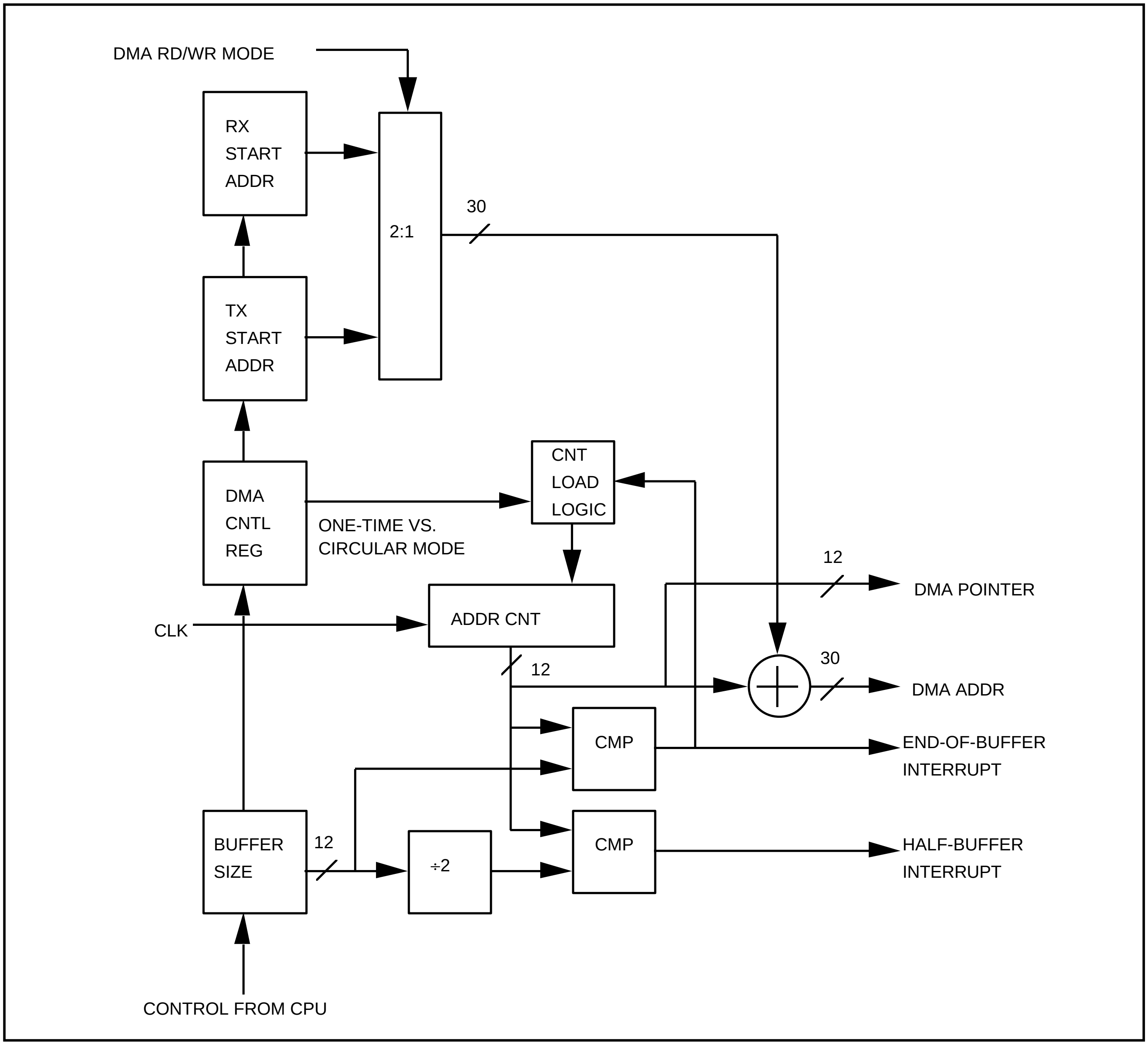


Figure 7-4. CHI DMA Address Generation

The DMA buffer size is programmable (up to a maximum of 16 KBytes) and the receive and transmit buffer start addresses are also programmable (anywhere over the full 32-bit address space). Because there are separate start addresses, the receive and transmit buffers can be configured to either reside in different memory spaces or share the same memory space. The latter setup allows for overlapping buffers for loopback purposes or for optimum memory allocation, for which the DMA logic supports two full-duplex loopback modes. For one mode, receive DMA requests are issued first, followed by transmit DMA requests. This ordering allows a receive-to-transmit immediate loopback via the DMA buffer. For the second mode, transmit DMA requests are issued first, followed by receive DMA requests. Thus, received samples are written to the DMA buffer location immediately after transmit samples were read from that same location (which then became immediately available). This ordering allows a single circular DMA buffer to be used for both transmit and receive samples.

The DMA buffers can be configured in a circular buffer mode or a one-time buffer mode. For the circular mode, the DMA address is continuously incremented (each time a DMA acknowledge is received from PR31700's central DMA controller) and rolls over back to the start address after the end-of-buffer is reached and will continue operating in a continuous and circular manner. For the one-time mode, the DMA logic will stop executing whenever the end-of-buffer is reached.

Because the CHI Module reads and writes a byte at a time between the shift registers and the longword holding registers, the software must pack and unpack these bytes to and from the longwords in memory in order to multiplex and demultiplex each channel for processing. Table 7-3 shows the format and organization of the CHI channels within memory for DMA mode. Consecutive byte samples for a given channel reside in memory every 8th byte.

Table 7-3. CHI DMA Memory Organization

(relative) memory address	bits 31:24	bits 23:16	bits 15:8	bits 7:0
0x0	buffA,ch3 sample0	buffA,ch2 sample0	buffA,ch1 sample0	buffA,ch0 sample0
0x4	buffB,ch3 sample0	buffB,ch2 sample0	buffB,ch1 sample0	buffB,ch0 sample0
0x8	buffA,ch3 sample1	buffA,ch2 sample1	buffA,ch1 sample1	buffA,ch0 sample1
0xC	buffB,ch3 sample1	buffB,ch2 sample1	buffB,ch1 sample1	buffB,ch0 sample1
0x10	buffA,ch3 sample2	buffA,ch2 sample2	buffA,ch1 sample2	buffA,ch0 sample2
etc.				

For a given channel, the minimum input-to-output latency for the CHI data path is
(4 cycles) x (62.5 sec) = 250 sec, assuming an 8 KHz frame rate.
These required 4 cycles are as follows:

- 1 cycle to load receive shift register into receive holding register
- 1 cycle for DMA of receive holding register data into receive memory space
- (insert here any application-specific time required for processing ofreceived data and moving result to transmit memory space)
- 1 cycle for DMA of data in transmit memory space to transmit holding register
- 1 cycle to load transmit holding register data into transmit shift register

Half-buffer and end-of-buffer DMA address counter interrupts are available, allowing the CPU to minimize overhead and utilize the DMA buffer in a ping-pong fashion. For transmit mode, the CPU can use these interrupts to fill or write one half of the buffer while the other half is being emptied by the DMA controller for transmitting out the CHI. Similarly, for receive mode, the CPU can use these interrupts to empty or read one half of the buffer while the other half is being filled by the DMA controller from received CHI input samples.

Also available is a direct CPU read/write mode for bypassing the DMA, allowing the CPU to read or write the CHI data on a sample by sample basis, if so desired. Separate DMA enables for receive and transmit allow DMA to be setup for receive only (transmit via CPU), transmit only (receive via CPU), receive and transmit, or none (receive and transmit via CPU).

The DMA circuit also provides an interrupt each time the DMA buffer pointer is incremented, which occurs whenever a new sample is read from and/or written to the DMA buffer. This interrupt may be useful for triggering a read of the DMA pointer status value, which is the actual 12-bit DMA address counter output. This value indicates exactly where the current address is pointing to in the overall DMA buffer.

7.3.6 Related Interrupts

CHI0_5INT:
Issues an interrupt whenever the CHI DMA buffer pointer has reached the halfway point.

CHI1_0INT:
Issues an interrupt whenever the CHI DMA buffer pointer has reached the end-of-buffer point.

CHIDMACNTINT:
Issues an interrupt each time the CHI DMA buffer pointer is incremented, which occurs whenever a new CHI sample is read from and/or written to the CHI DMA buffer.

CHIININTA:
Issues an interrupt whenever a valid CHI input sample is available from CHI RX Holding Register A; this also means a valid CHI output sample can be written to CHI TX Holding Register B.

CHIININTB:
Issues an interrupt whenever a valid CHI input sample is available from CHI RX Holding Register B; this also means a valid CHI output sample can be written to CHI TX Holding Register A.

CHIACTINT:
Issues an interrupt whenever CHICLK is active. This is used for CHI wakeup purposes.

CHIERRINT:
Issues an interrupt whenever a CHI error is received. This interrupt is triggered if CPU or DMA reading of the CHI RX Holding Registers does not keep up with the hardware filling of the CHI RX Holding Registers or if CPU or DMA writing of the CHI TX Holding Registers does not keep up with the hardware emptying of the CHI TX Holding Registers.

7.4 CHI Registers

7.4.1 CHI Control Register

OFFSET = $1D8:

Bit	Label	RESET	Read/Write
31-30	Reserved		
29	CHILOOP	0	R/W
28	CHIENTEST	0	R/W
27	CHIFSDIR	0	R/W
26-25	CHIFSWIDTH(1:0)	X	R/W
24-20	CHINCHAN(4:0)	X	R/W
19-16	CHITXBOFF(3:0)	X	R/W
15-12	CHIRXBOFF(3:0)	X	R/W
11	TXMSBFIRST	X	R/W
10	RXMSBFIRST	X	R/W
9	CHIRXFSPOL	X	R/W
8	CHITXFSPOL	X	R/W
7	CHIRXEDGE	X	R/W
6	CHITXEDGE	X	R/W
5	CHIFSEDGE	X	R/W
4	CHITXFSEDGE	X	R/W
3	CHICLK2XMODE	0	R/W
2	CHIRXEN	0	R/W
1	CHITXEN	0	R/W
0	ENCHI	0	R/W

CHILOOP:
This bit is used for IC testing and should not be set. Setting this bit to a logic "1" will cause the CHI serial transmitted data to be internally looped back to the CHI serial receive data path. The data is inverted when this mode is selected. Clearing this bit to a logic "0" selects the normal CHIDIN pin as the CHI serial receive data source.

CHIENTEST:
This bit is used for IC testing and should not be set.

CHIFSDIR:
This bit controls the direction of the CHIFS pin. Setting this bit to a logic "1" configures CHIFS to be an output (CHI sync master mode). Clearing this bit to a logic "0" configures CHIFS to be an input (CHI sync slave mode).

CHIFSWIDTH(1:0):
These bits are used to select pulse width for the CHIFS signal, relevant whenever the CHI Module is configured as master mode. The available CHIFS pulse widths are as follows:

CHIFSWIDTH	CHIFS pulse width
0	1 bit wide
1	2 bits wide
2	1 byte wide
3	(CHINCHAN 2) channels wide

CHINCHAN(4:0):
These bits are used to program the number of 8-bit channel timeslots per half-frame, up to 32 total per half-frame. The value loaded for CHINCHAN is the desired number of channels - 1.

CHITXBOFF(3:0):
These bits select the transmit data programmable bit offset, which is related to the number of clocks from the start of timeslot 0 (1st timeslot) transmit data to the CHIFS edge used to trigger the start of each CHI frame. The value loaded for CHITXBOFF is the desired bit offset - 1.

CHIRXBOFF(3:0):
These bits select the receive data programmable bit offset, which is related to the number of clocks from the start of timeslot 0 (1st timeslot) receive data to the CHIFS edge used to trigger the start of each CHI frame. The value loaded for CHIRXBOFF is the desired bit offset - 1.

TXMSBFIRST:
This bit selects between MSB-first and LSB-first serial data formats for each byte of the CHI transmit data. Setting this bit to a logic "1" selects MSB-first. Clearing this bit to a logic "0" selects LSB-first.

RXMSBFIRST:
This bit selects between MSB-first and LSB-first serial data formats for each byte of the CHI receive data. Setting this bit to a logic "1" selects MSB-first. Clearing this bit to a logic "0" selects LSB-first.

CHIRXFSPOL:
This bit selects between positive (active high) or negative (active low) polarity for the received CHIFS signal pulse, relevant whenever the CHI Module is configured as slave mode. Setting this bit to a logic "1" selects negative polarity for the CHIFS pulse. In this case, the falling edge of the CHIFS pulse is used to trigger the start of the CHI frame period. Clearing this bit to a logic "0" selects positive polarity for the CHIFS pulse. In this case, the rising edge of the CHIFS pulse is used to trigger the start of the CHI frame period.

CHITXFSPOL:
This bit selects between positive (active high) or negative (active low) polarity for the transmitted CHIFS signal pulse, relevant whenever the CHI Module is configured as master mode. Setting this bit to a logic "1" selects negative polarity for the CHIFS pulse. Clearing this bit to a logic "0" selects positive polarity for the CHIFS pulse.

CHIRXEDGE:
This bit selects whether to use either the rising edge or falling edge of CHICLK to sample the receive data CHIDIN. Setting this bit to a logic "1" selects rising edge. Clearing this bit to a logic "0" selects falling edge.

CHITXEDGE:
This bit selects whether to use either the rising edge or falling edge of CHICLK to clock out the transmit data CHIDOUT. Setting this bit to a logic "1" selects rising edge. Clearing this bit to a logic "0" selects falling edge.

CHIFSEDGE:
This bit selects whether to use either the rising edge or falling edge of CHICLK to sample the receive frame sync CHIFS. Setting this bit to a logic "1" selects rising edge. Clearing this bit to a logic "0" selects falling edge.

CHITXFSEDGE:
This bit selects whether to use either the rising edge or falling edge of CHICLK to clock out the transmit frame sync CHIFS, relevant whenever the CHI Module is configured as master mode. Setting this bit to a logic "1" selects rising edge. Clearing this bit to a logic "0" selects falling edge.

CHICLK2XMODE:
This bit selects between 1x and 2x clock modes. Setting this bit to a logic "1" selects 2x clock mode, which means that the CHICLK frequency equals twice the serial data bit rate. Clearing this bit to a logic "0" selects 1x clock mode, which means that the CHICLK frequency equals the serial data bit rate.

CHIRXEN:
This bit is used to enable/disable CHI receive processing. Setting this bit to a logic "1" enables CHI receive processing. Clearing this bit to a logic "0" disables CHI receive processing, causing all received data to not be processed by the CHI module. This bit should not be set until after the CHI Module is setup, then ENCHI asserted.

CHITXEN:
This bit is used to enable/disable CHI transmit processing. Setting this bit to a logic "1" enables CHI transmit processing. Clearing this bit to a logic "0" disables CHI transmit processing, causing the CHI serial transmitted data to be tri-stated. This bit should not be set until after the CHI Module is setup, then ENCHI asserted.

ENCHI:
This bit is used to enable/disable the CHI module. Setting this bit to a logic "1" enables the CHI module. Clearing this bit to a logic "0" disables the CHI Module and keeps the module in a reset state.

7.4.2 CHI Pointer Enable Register

OFFSET = $1DC:

Bit	Label	RESET	Read/Write
31	CHITXPTRB3EN	X	R/W
30	CHITXPTRB2EN	X	R/W
29	CHITXPTRB1EN	X	R/W
28	CHITXPTRB0EN	X	R/W
27	CHITXPTRA3EN	X	R/W
26	CHITXPTRA2EN	X	R/W
25	CHITXPTRA1EN	X	R/W
24	CHITXPTRA0EN	X	R/W
23	CHIRXPTRB3EN	X	R/W
22	CHIRXPTRB2EN	X	R/W
21	CHIRXPTRB1EN	X	R/W
20	CHIRXPTRB0EN	X	R/W
19	CHIRXPTRA3EN	X	R/W
18	CHIRXPTRA2EN	X	R/W
17	CHIRXPTRA1EN	X	R/W
16	CHIRXPTRA0EN	X	R/W
15-0	Reserved		

CHITXPTRB3EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRB3. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRB2EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRB2. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRB1EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRB1. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRB0EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRB0. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRA3EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRA3. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRA2EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRA2. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRA1EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRA1. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHITXPTRA0EN:
This bit is used to enable/disable the timeslot for the transmit channel pointed to by the TDM switch pointer CHITXPTRA0. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRB3EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRB3. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRB2EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRB2. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRB1EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRB1. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRB0EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRB0. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRA3EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRA3. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRA2EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRA2. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRA1EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRA1. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

CHIRXPTRA0EN:
This bit is used to enable/disable the timeslot for the receive channel pointed to by the TDM switch pointer CHIRXPTRA0. Setting this bit to a logic "1" enables the timeslot. Clearing this bit to a logic "0" disables the timeslot.

7.4.3 CHI Receive Pointer A Register

OFFSET = $1E0: **write-only**

Bit	Label	RESET	Read/Write
31-29	Reserved		
28-24	CHIRXPTRA3(4:0)	X	W
23-21	Reserved		
20-16	CHIRXPTRA2(4:0)	X	W
15-13	Reserved		
12-8	CHIRXPTRA1(4:0)	X	W
7-5	Reserved		
4-0	CHIRXPTRA0(4:0)	X	W

CHIRXPTRA3(4:0): **write-only**
These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 3 of the CHI receive holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHIRXPTRA2(4:0): **write-only**
These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 2 of the CHI receive holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHIRXPTRA1(4:0): **write-only**
These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 1 of the CHI receive holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHIRXPTRA0(4:0): **write-only**
These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 0 of the CHI receive holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

7.4.4 CHI Receive Pointer B Register

OFFSET = $1E4: write-only

Bit	Label	RESET	Read/Write
31-29	Reserved		
28-24	CHIRXPTRB3(4:0)	X	W
23-21	Reserved		
20-16	CHIRXPTRB2(4:0)	X	W
15-13	Reserved		
12-8	CHIRXPTRB1(4:0)	X	W
7-5	Reserved		
4-0	CHIRXPTRB0(4:0)	X	W

CHIRXPTRB3(4:0): write-only

These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 3 of the CHI receive holding register B; register B handles all timeslots from channel (CHINCHAN 2)to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHIRXPTRB2(4:0): write-only

These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 2 of the CHI receive holding register B; register B handles all timeslots from channel (CHINCHAN 2)to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHIRXPTRB1(4:0): write-only

These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 1 of the CHI receive holding register B; register B handles all timeslots from channel (CHINCHAN 2)to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHIRXPTRB0(4:0): write-only

These bits represent the TDM switch pointer which defines the receive channel timeslot for byte 0 of the CHI receive holding register B; register B handles all timeslots from channel (CHINCHAN 2)to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

7.4.5 CHI Transmit Pointer A Register

OFFSET = $1E8: write-only

Bit	Label	RESET	Read/Write
31-29	Reserved		
28-24	CHITXPTRA3(4:0)	X	W
23-21	Reserved		
20-16	CHITXPTRA2(4:0)	X	W
15-13	Reserved		
12-8	CHITXPTRA1(4:0)	X	W
7-5	Reserved		
4-0	CHITXPTRA0(4:0)	X	W

CHITXPTRA3(4:0): write-only

These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 3 of the CHI transmit holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHITXPTRA2(4:0): write-only

These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 2 of the CHI transmit holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHITXPTRA1(4:0): write-only

These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 1 of the CHI transmit holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

CHITXPTRA0(4:0): write-only

These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 0 of the CHI transmit holding register A; register A handles all timeslots from channel 0 to channel (CHINCHAN 2) - 1.

7.4.6 CHI Transmit Pointer B Register

OFFSET = $1EC: **write-only**

Bit	Label	RESET	Read/Write
31-29	Reserved		
28-24	CHITXPTRB3(4:0)	X	W
23-21	Reserved		
20-16	CHITXPTRB2(4:0)	X	W
15-13	Reserved		
12-8	CHITXPTRB1(4:0)	X	W
7-5	Reserved		
4-0	CHITXPTRB0(4:0)	X	W

CHITXPTRB3(4:0): **write-only**
These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 3 of the CHI transmit holding register B; register B handles all timeslots from channel (CHINCHAN 2) to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHITXPTRB2(4:0): **write-only**
These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 2 of the CHI transmit holding register B; register B handles all timeslots from channel (CHINCHAN 2) to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHITXPTRB1(4:0): **write-only**
These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 1 of the CHI transmit holding register B; register B handles all timeslots from channel (CHINCHAN 2) to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

CHITXPTRB0(4:0): **write-only**
These bits represent the TDM switch pointer which defines the transmit channel timeslot for byte 0 of the CHI transmit holding register B; register B handles all timeslots from channel (CHINCHAN 2) to channel CHINCHAN - 1. The value loaded for this TDM switch pointer is the desired timeslot number minus (CHINCHAN 2).

7.4.7 CHI Size Register

OFFSET = $1F0:

Bit	Label	RESET	Read/Write
31-30	Reserved		
29-18	CHIDMAPTR(31:2)	-	R
17-16	Reserved		
15	CHIBUFF1TIME	0	R/W
14	CHIDMALOOP	0	R/W
13-2	CHISIZE(13:2)	X	W
1	ENDMARXCHI	0	R/W
0	ENDMATXCHI	0	R/W

CHIDMAPTR(13:2): **read-only**
These bits provide the status of the CHI DMA counter.

CHIBUFF1TIME:
The CHI DMA controller supports two buffer addressing modes depending on the state of this bit. When CHIBUFF1TIME is set to a logic "1", the CHI DMA controller will stop executing when it reaches the end of the DMA buffer. When CHIBUFF1TIME is cleared to a logic "0", the CHI DMA controller will loop back to the start of the DMA buffer when the end of the DMA buffer is reached and will continue operating in a continuous and circular manner.

CHIDMALOOP:
The CHI DMA controller supports two full-duplex loopback modes depending on the state of this bit. When CHIDMALOOP is set to a logic "1", the CHI DMA controller issues RX DMA requests first, followed by TX DMA requests. This ordering allows an RX-to-TX immediate loopback via the DMA buffer. When CHIDMALOOP is cleared to a logic "0", the CHI DMA controller issues TX DMA requests first, followed by RX DMA requests. This ordering allows a single circular DMA buffer to be used for both TX and RX, if so desired.

CHISIZE(31:2): write-only

These bits define the size of the CHI DMA buffers (16 KBytes maximum). Both the CHI RX buffer and the CHI TX buffer are the same size. The last address in the CHI RX DMA buffer is given by CHIRXSTART(31:2) + CHISIZE(13:2). The last address in the CHI TX DMA buffer is given by CHITXSTART(31:2) + CHISIZE(13:2). The value loaded into CHISIZE should be equal to the desired buffer length - 1.

ENDMARXCHI:

This bit enables the CHI DMA receive function. Setting this bit to a logic "1"enables the DMA mode. Clearing this bit to a logic "0" disables the DMAmode. This bit should not be set until the CHIRXSTART, CHITXSTART, andCHISIZE registers are setup and the CHI Module is enabled (ENCHI asserted).Either ENDMARXCHI or ENDMATXCHI or both can be set at a time since the CHI DMA controller can support full duplex operation.

ENDMATXCHI:

This bit enables the CHI DMA transmit function. Setting this bit to a logic "1" enables the DMA mode. Clearing this bit to a logic "0" disables the DMA mode. This bit should not be set until the CHIRXSTART, CHITXSTART, and CHISIZE registers are setup and the CHI Module is enabled (ENCHI asserted). Either ENDMARXCHI or ENDMATXCHI or both can be set at a time since the CHI DMA controller can support full duplex operation.

7.4.8 CHI RX Start Register

OFFSET = $1F4: write-only

Bit	Label	RESET	Read/Write
31-2	CHIRXSTART(31:2)	X	W
1-0	Reserved		

CHIRXSTART(31:2): write-only

These bits define the start address for the CHI RX DMA buffer. The CHI RX buffer and CHI TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

7.4.9 CHI TX Start Register

OFFSET = $1F8: write-only

Bit	Label	RESET	Read/Write
31-2	CHITXSTART(31:2)	X	W
1-0	Reserved		

CHITXSTART(31:2): write-only

These bits define the start address for the CHI TX DMA buffer. The CHI RX buffer and CHI TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

7.4.10 CHI TX Holding Register

OFFSET = $1FC: write-only

Bit	Label	RESET	Read/Write
31-0	CHITXHOLD(31:0)	X	W

CHITXHOLD(31:0): write-only

These bits represent the CHI data to be transmitted. CHI data can be either written directly to this register by the CPU or transparently read from the CHI TX DMA buffer to this register. This register should only be loaded by the CPU after the CHIININTA or CHIININTB interrupt is asserted. Transmit data for bytes 3, 2, 1, and 0 are loaded into the 32-bit longword CHITXHOLD at locations (31:24), (23:16), (15:8), and (7:0), respectively. These data bytes correspond to the CHI timeslots as defined by the values in the CHITXPTRA and CHITXPTRB TDM switch registers.

7.4.11 CHI RX Holding Register

OFFSET = $1FC: read-only

Bit	Label	RESET	Read/Write
31-0	CHIRXHOLD(31:0)	-	R

CHIRXHOLD(31:0): read-only

These bits represent the CHI data to be received. CHI data can be either read directly from this register by the CPU or transparently written to the CHI RX DMA buffer from this register. This register should only be read by the CPU after the CHIININTA or CHIININTB interrupt is asserted. Receive data for bytes 3, 2, 1, and 0 are stored into the 32-bit longword CHIRXHOLD at locations (31:24), (23:16), (15:8), and (7:0), respectively. These data bytes correspond to the CHI timeslots as defined by the values in the CHIRXPTRA and CHIRXPTRB TDM switch registers.

Chapter 8
Interrupt Module

This section describes the Interrupt Module, which contains logic for reading, enabling, and clearing all interrupts.

8.1 Overview

The Interrupt Module within the PR31700 contains logic for reading, enabling, and clearing all of the interrupt sources. These interrupts are either generated from internal PR31700 modules or from edge transitions on external signal pins. All of these latter interrupt types generate a separate positive and negative edge interrupt. The status (logic state) of each interrupt is readable, allowing any interrupt event to be polled, if desired. Each interrupt status signal is also gated with the corresponding interrupt enable bit in order to generate the overall IRQ output to the CPU. The interrupt status register is reset using the corresponding clear bit from the corresponding clear interrupt register.

8.2 Implementation

8.2.1 Block Diagram

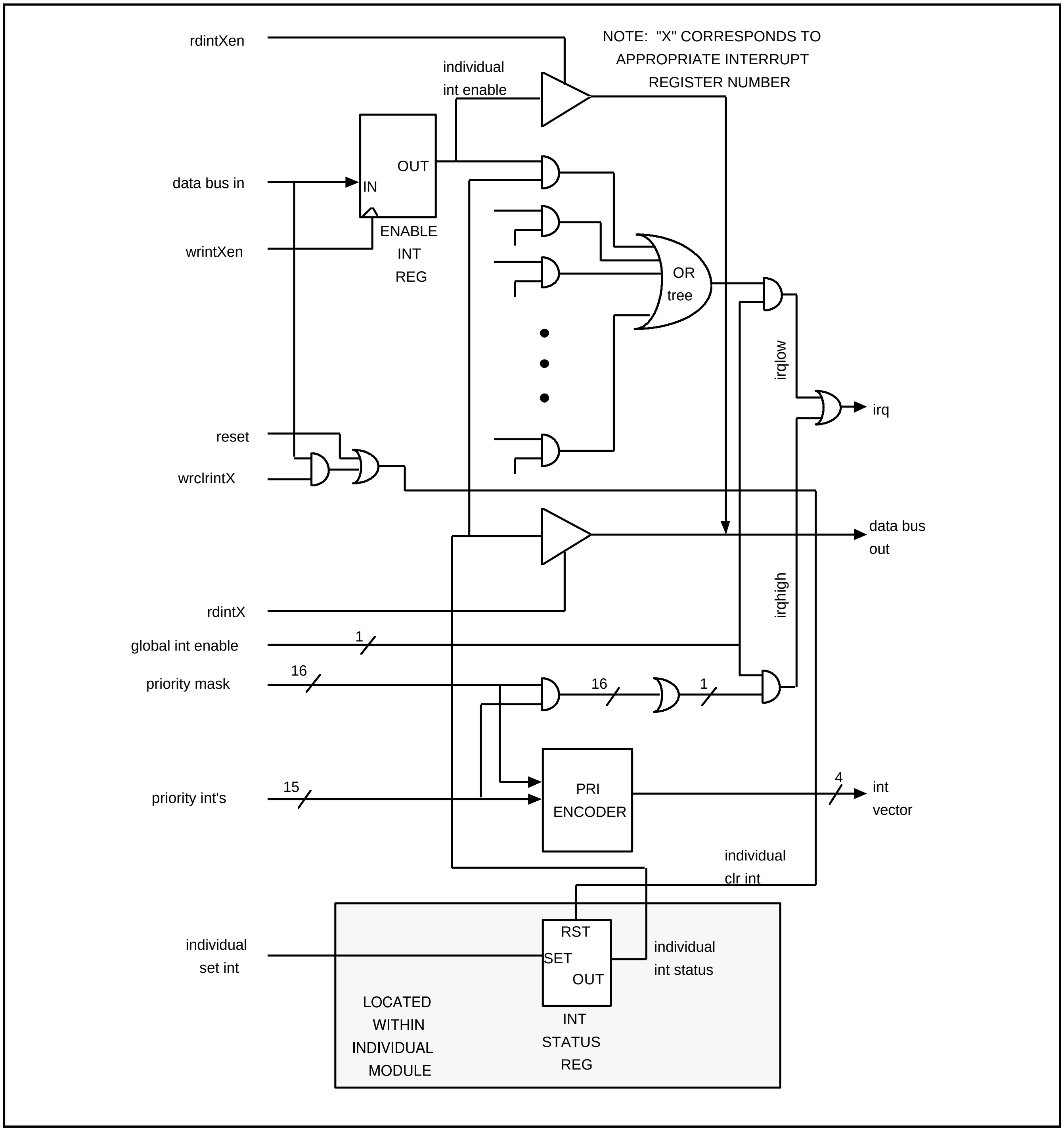


Figure 8-1. Interrupt Circuit Block Diagram

8.2.2 Interrupt Logic Description

Figure 8-1 shows a block diagram of the interrupt circuit for a single interrupt source. Every other interrupt source contains an identical circuit. The overall Interrupt Module contains 5 main sets of registers (numbered 1 through 5), with a 6th register allocated for setting the global interrupt enable and the 16-bit Priority Mask, and for reading the Priority Interrupt Vector and IRQHIGH and IRQLOW status bits. Each bit within each of the 5 main sets then consists of 3 registers: Status, Enable, and Clear.

- Interrupt Status Register read-only
  - allows CPU to read logic state of interrupt status
  - informs the CPU which interrupts are pending
- Enable Interrupt Register read/write
  - allows CPU to enable a given interrupt
  - each individual interrupt has a corresponding enable bit
- Clear Interrupt Register write-only
  - allows CPU to clear a given pending interrupt
  - each individual interrupt has a corresponding clear bit
  - each individual interrupt clear address and bit position maps one-to-one with the interrupt status address and bit position (status transactions are CPU reads, clear transactions are CPU writes)

A given Interrupt Status Register is set by the corresponding individual interrupt event. For example, this event could be a positive (or negative) edge transition on an external PR31700 pin. Another example might be an event triggered by an internal PR31700 module, such as a particular DMA buffer reaching the end-of-buffer point.

The output of a given Interrupt Status Register is readable by the CPU, by reading the address which points to the corresponding Interrupt Status Register. A given interrupt is enabled by writing a "1" to the corresponding bit position in the corresponding Enable Interrupt Register. The Interrupt Status Register output bit is then gated with the enable bit from the corresponding Enable Interrupt Register. The output of this gate is then combined with all the other gated Interrupt Status bits using a wide bitwise "OR" tree in order to generate the IRQLOW status signal. In other words, if ANY of the gated Interrupt Status signals is asserted and the global interrupt enable GLOBALEN is asserted, then IRQLOW will be asserted.

A given pending interrupt is cleared by writing a "1" to the corresponding bit position in the corresponding Clear Interrupt Register. This output of this Clear Interrupt Register is used to reset the corresponding Interrupt Status Register.

On power on reset, the Global Enable is cleared to prevent any interrupts from occuring until the Enable Interrupt Registers are initialized. Also, all Clear Interrupts will be set for the duration of reset (thus interrupts cleared).

The Interrupt Module also contains 15 High Priority Interrupt sources. A 16-bit Priority Mask (located within the Enable Interrupt 6 Register) is used to enable any of these high priority interrupts, by writing a "1" to the corresponding bit position in the Enable Interrupt 6 Register, where bit 15 is the highest priority. Each bit of the 16-bit Priority Mask is gated with the corresponding 15 High Priority Interrupt signals and these gated signals are then combined using a bitwise 16-input "OR" tree in order to generate the IRQHIGH status signal. If IRQHIGH or IRQLOW is asserted, then the overall IRQ will be asserted.

A Priority Encoder circuit also compares the 16-bit Priority Mask with the 15 High Priority Interrupt signals and generates a 4-bit interrupt status vector which points to the highest priority pending interrupt from the set of 16 possible events. Level 15 is the highest priority and Level 1 is the lowest priority, with Level 0 corresponding to the standard interrupt handler.

The IRQHIGH signal is connected to interrupt bit 4 on the R3000 and the IRQLOW signal is connected to interrupt bit 2. The IRQHIGH signal is also connected to CPU Co-Processor Condition bit 3.

8.3 Interrupt Registers

Interrupt Status 1 Register

OFFSET = $100: **read-only**

Bit	Label	RESET	Read/Write
31	LCDINT	–	R
30	DFINT	–	R
29	CHI0_5INT	–	R
28	CHI1_0INT	–	R
27	CHIDMACNTINT	–	R
26	CHIININTA	–	R
25	CHIININTB	–	R
24	CHIACTINT	–	R
23	CHIERRINT	–	R
22	SND0_5INT	–	R
21	SND1_0INT	–	R
20	TEL0_5INT	–	R
19	TEL1_0INT	–	R
18	SNDDMACNTINT	–	R
17	TELDMACNTINT	–	R
16	LSNDCLIPINT	–	R
15	RSNDCLIPINT	–	R
14	VALSNDPOSINT	–	R
13	VALSNDNEGINT	–	R
12	VALTELPOSINT	–	R
11	VALTELNEGINT	–	R
10	SNDININT	–	R
9	TELININT	–	R
8	SIBSF0INT	–	R
7	SIBSF1INT	–	R
6	SIBIRQPOSINT	–	R
5	SIBIRQNEGINT	–	R
4-0	Reserved		

LCDINT:
Issues an interrupt at the end of each video frame.

DFINT:
Issues an interrupt each time the video DF signal toggles.

CHI0_5INT:
Issues an interrupt whenever the CHI DMA buffer pointer has reached the halfway point.

CHI1_0INT:
Issues an interrupt whenever the CHI DMA buffer pointer has reached the end-of-buffer point.

CHIDMACNTINT:
Issues an interrupt each time the CHI DMA buffer pointer is incremented, which occurs whenever a new CHI sample is read from and/or written to the CHI DMA buffer.

CHIININTA:
Issues an interrupt whenever a valid CHI input sample is available from CHI RX Holding Register A; this also means a valid CHI output sample can be written to CHI TX Holding Register B.

CHIININTB:
Issues an interrupt whenever a valid CHI input sample is available from CHI RX Holding Register B; this also means a valid CHI output sample can be written to CHI TX Holding Register A.

CHIACTINT:
Issues an interrupt whenever CHICLK is active. This is used for CHI wakeup purposes.

CHIERRINT:
Issues an interrupt whenever a CHI error is received. This interrupt is triggered if CPU or DMA reading of the CHI RX Holding Registers does not keep up with the hardware filling of the CHI RX Holding Registers or if CPU or DMA writing of the CHI TX Holding Registers does not keep up with the hardware emptying of the CHI TX Holding Registers.

SND0_5INT:
Issues an interrupt whenever the sound DMA buffer pointer has reached the halfway point.

SND1_0INT:
Issues an interrupt whenever the sound DMA buffer pointer has reached the end-of-buffer point.

TEL0_5INT:
Issues an interrupt whenever the telecom DMA buffer pointer has reached the halfway point.

TEL1_0INT:
Issues an interrupt whenever the telecom DMA buffer pointer has reached the end-of-buffer point.

SNDDMACNTINT:
Issues an interrupt each time the sound DMA buffer pointer is incremented, which occurs whenever a new sound sample is read from and/or written to the sound DMA buffer.

TELDMACNTINT:
Issues an interrupt each time the telecom DMA buffer pointer is incremented, which occurs whenever a new telecom sample is read from and/or written to the telecom DMA buffer.

LSNDCLIPINT:
Issues an interrupt whenever the amplitude of the left channel sound data is clipping the codec A/D converter for SIB subframe 1.

RSNDCLIPINT:
Issues an interrupt whenever the amplitude of the right channel sound data is clipping the codec A/D converter for SIB subframe 1.

VALSNDPOSINT:
Issues an interrupt whenever the valid sound status flag transitions from a logic "0" to a logic "1". This valid flag is triggered from SIB subframe 0 (if SELSNDSF1 = "0") or from SIB subframe 1 (if SELSNDSF1 = "1").

VALSNDNEGINT:
Issues an interrupt whenever the valid sound status flag transitions from a logic "1" to a logic "0". This valid flag is triggered from SIB subframe 0 (if SELSNDSF1 = "0") or from SIB subframe 1 (if SELSNDSF1 = "1").

VALTELPOSINT:
Issues an interrupt whenever the valid telecom status flag transitions from a logic "0" to a logic "1". This valid flag is triggered from SIB subframe 0 (if SELTELSF1 = "0") or from SIB subframe 1 (if SELTELSF1 = "1").

VALTELNEGINT:
Issues an interrupt whenever the valid telecom status flag transitions from a logic "1" to a logic "0". This valid flag is triggered from SIB subframe 0 (if SELTELSF1 = "0") or from SIB subframe 1 (if SELTELSF1 = "1").

SNDININT:
Issues an interrupt whenever a valid sound input longword (32 bits) is available from the Sound RX Holding Register; this also means a valid sound output longword can be written to the Sound TX Holding Register.

TELININT:
Issues an interrupt whenever a valid telecom input longword (32 bits) is available from the Telecom RX Holding Register; this also means a valid telecom output longword can be written to the Telecom TX Holding Register.

SIBSF0INT:
Issues an interrupt at the start of every SIB subframe 0. This is used to initiate CPU reading of the SIB Subframe 1 Status Register (SF1STAT Register) and/or CPU writing of the SIB Subframe 0 Control Register (SF0AUX Register).

SIBSF1INT:
Issues an interrupt at the start of every SIB subframe 1. This is used to initiate CPU reading of the SIB Subframe 0 Status Register (SF0STAT Register) and/or CPU writing of the SIB Subframe 1 Control Register (SF1AUX Register).

SIBIRQPOSINT:
Issues an interrupt whenever the SIBIRQ pin transitions from a logic "0" to a logic "1".

SIBIRQNEGINT:
Issues an interrupt whenever the SIBIRQ pin transitions from a logic "1" to a logic "0".

8.3.2 Interrupt Status 2 Register

OFFSET = $104: **read-only**

Bit	Label	RESET	Read/Write
31	UARTARXINT	–	R
30	UARTARXOVERRUNINT	–	R
29	UARTAFRAMEERRINT	–	R
28	UARTABREAKINT	–	R
27	UARTAPARITYERRINT	–	R
26	UARTATXINT	–	R
25	UARTATXOVERRUNINT	–	R
24	UARTAEMPTYINT	–	R
23	UARTADMAFULLINT	–	R
22	UARTADMAHALFINT	–	R
21	UARTBRXINT	–	R
20	UARTBRXOVERRUNINT	–	R
19	UARTBFRAMEERRINT	–	R
18	UARTBBREAKINT	–	R
17	UARTBPARITYERRINT	–	R
16	UARTBTXINT	–	R
15	UARTBTXOVERRUNINT	–	R
14	UARTBEMPTYINT	–	R
13	UARTBDMAFULLINT	–	R
12	UARTBDMAHALFINT	–	R
11	Reserved	–	R
10	Reserved	–	R
9	Reserved	–	R
8	Reserved	–	R
7	Reserved	–	R
6	Reserved	–	R
5	Reserved	–	R
4	Reserved	–	R
3	Reserved	–	R
2	Reserved	–	R
1-0	Reserved		

UARTARXINT:
Issues an interrupt whenever the UARTA Receive Holding Register is loaded with data.

UARTARXOVERRUNINT:
Issues an interrupt if the UARTA Receive Holding Register is loaded twice before the interrupt is service.

UARTAFRAMEERRINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register contains a frame error.

UARTABREAKINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register is a break.

UARTAPARITYERRINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register contains a parity error.

UARTATXINT:
Issues an interrupt if the UARTA Transmit Holding Register is available.

UARTATXOVERRUNINT:
Issues an interrupt if the UARTA Transmit Holding Register is written to when the Transmit Holding Register is not available.

UARTAEMPTYINT:
Issues an interrupt if the UARTA Transmit Holding Register and Transmit Shift Register are both empty.

UARTADMAFULLINT:
Issues an interrupt if the UARTA DMA counter reaches the end of the buffer.

UARTADMAHALFINT:
Issues an interrupt if the UARTA DMA counter reaches the mid point of the buffer.

UARTBRXINT:
Issues an interrupt whenever the UARTB Receive Holding Register is loaded with data.

UARTBRXOVERRUNINT:
Issues an interrupt if the UARTB Receive Holding Register is loaded twice before the interrupt is service.

UARTBFRAMEERRINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register contains a frame error.

UARTBBREAKINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register is a break.

UARTBPARITYERRINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register contains a parity error.

UARTBTXINT:
Issues an interrupt if the UARTB Transmit Holding Register is available.

UARTBTXOVERRUNINT:
Issues an interrupt if the UARTB Transmit Holding Register is written to when the Transmit Holding Register is not available.

UARTBEMPTYINT:
Issues an interrupt if the UARTB Transmit Holding Register and Transmit Shift Register are both empty.

UARTBDMAFULLINT:
Issues an interrupt if the UARTB DMA counter reaches the end of the buffer.

UARTBDMAHALFINT:
Issues an interrupt if the UARTB DMA counter reaches the mid point of the buffer.

8.3.3 Interrupt Status 3 Register

OFFSET = $108: **read-only**

Bit	Label	RESET	Read/Write
31-0	MFIOPOSINT(31:0)	–	R

MFIOPOSINT(31:0):
Issues an interrupt whenever any of the multi-function I/O pin(s) transition from a logic "0" to a logic "1". There are a total of 32 multi-function I/O pins (31 thru 0), each of which corresponds to the respective bit within this Status Register. Each multi-function I/O pin can independently trigger an interrupt.

8.3.4 Interrupt Status 4 Register

OFFSET = $10C: read-only

Bit	Label	RESET	Read/Write
31-0	MFIONEGINT(31:0)	–	R

MFIONEGINT(31:0):
Issues an interrupt whenever any of the multi-function I/O pin(s) transition from a logic "1" to a logic "0". There are a total of 32 multi-function I/O pins (31 thru 0), each of which corresponds to the respective bit within this Status Register. Each multi-function I/O pin can independently trigger an interrupt.

8.3.5 Interrupt Status 5 Register

OFFSET = $110: read-only

Bit	Label	RESET	Read/Write
31	RTCINT	–	R
30	ALARMINT	–	R
29	PERINT	–	R
28	STPTIMERINT	–	R
27	POSPWRINT	–	R
26	NEGPWRINT	–	R
25	POSPWROKINT	–	R
24	NEGPWROKINT	–	R
23	POSONBUTNINT	–	R
22	NEGONBUTNINT	–	R
21	SPIBUFAVAILINT	–	R
20	SPIERRINT	–	R
19	SPIRCVINT	–	R
18	SPIEMPTYINT	–	R
17	IRCONSMINT	–	R
16	CARSTINT	–	R
15	POSCARINT	–	R
14	NEGCARINT	–	R
13-7	IOPOSINT(6:0)	–	R
6-0	IONEGINT(6:0)	–	R

RTCINT:
This interrupt is set whenever all 40 bits of the RTC counter reach a value of "$FFFFFFFFFF" to alert the software that the counter is "rolling over".

ALARMINT:
This interrupt is set whenever the RTC counter reaches a count that is equal to the value of the ALARM(39:0) bits set in the Alarm Register.

PERINT:
This interrupt is set whenever the Periodic Timer is enabled and the Periodic Timer counter reaches a count of zero.

STPTIMERINT:
This interrupt is set whenever the Stop Timer Counter counts up to the value set by the STPTIMERVAL(3:0) control bits.

POSPWRINT:
This interrupt is set when the PWRINT pin transitions from a logic "0" to a logic "1".

NEGPWRINT:
This interrupt is set when the PWRINT pin transitions from a logic "1" to a logic "0".

POSPWROKINT:
Issues an interrupt whenever the PWROK signal transitions from a logic "0" to a logic "1".

NEGPWROKINT:
Issues an interrupt whenever the PWROK signal transitions from a logic "1" to a logic "0".

POSONBUTNINT:
Issues an interrupt whenever the ONBUTN signal transitions from a logic "0" to a logic "1". If the DBNCONBUTN control bit is set then the interrupt will not set until the signal is debounced for 16-24 ms.

NEGONBUTNINT:
Issues an interrupt whenever the ONBUTN signal transitions from a logic "1" to a logic "0". If the DBNCONBUTN control bit is set then the interrupt will not set until the signal is debounced for 16-24 ms.

SPIBUFAVAILINT:
This interrupt is set when the ENSPI bit is first asserted and subsequently when the contents of the SPI Transmitter Holding Register are transferred to the SPI Shift Register. This interrupt is used to indicate that the SPI Transmitter Holding Register is available to be written by the software.

SPIERRINT:
This interrupt is set whenever the SPI Transmitter Holding Register is written, but the SPIBUFAVAILINT has not set to indicate that the register is available. This interrupt serves as a overrun indication for the software.

SPIRCVINT:
This interrupt is whenever the contents of the SPI Shift Register are transferred to the SPI Receiver Holding Register. This interrupt is used to indicate that there is valid data in the SPI Receiver Holding Register to be read by the software.

SPIEMPTYINT:
This interrupt is set whenever the both the SPI Shift Register and the SPI Transmitter Holding Register are empty. This interrupt can be used by the software to determine when the SPI is idle.

IRCONSMINT:
Whenever the upper byte of data is loaded into the 7-bit Period Number Counter, the lower byte of data is loaded into an intermediate holding register. At this time the 16-bit IR Holding Register is empty. The IRCONSMINT interrupt is then set to inform the CPU that the IR Holding Register is available. The CPU must fill the next word of data into the IR Holding Register before the 7-bit Period Number Counter is finished counting the periods for both the upper and lower bytes of data.

CARSTINT:
This interrupt is set whenever the Carrier Detect State Machine samples the CARDET pin = "1" just before turning off the RXPWR pin.

POSCARINT:
This interrupt is set whenever CARDET pin transitions from a logic "0" to a logic "1".

NEGCARINT:
This interrupt is set whenever CARDET pin transitions from a logic "1" to a logic "0".

IOPOSINT(6:0):
Issues an interrupt whenever any of the general purpose I/O pin(s) transition from a logic "0" to a logic "1". There are a total of 7 general purpose I/O pins (6 thru 0), each of which corresponds to the respective bit within this Status Register. Each general purpose I/O pin can independently trigger an interrupt.

IONEGINT(6:0):
Issues an interrupt whenever any of the general purpose I/O pin(s) transition from a logic "1" to a logic "0". There are a total of 7 general purpose I/O pins (6 thru 0), each of which corresponds to the respective bit within this Status Register. Each general purpose I/O pin can independently trigger an interrupt.

8.3.6 Interrupt Status 6 Register

OFFSET = $114: read-only

Bit	Label	RESET	Read/Write
31	IRQHIGH	–	R
30	IRQLOW	–	R
29-6	Reserved		
5-2	INTVECT(3:0)	–	R
1-0	Reserved		

IRQHIGH:
This status bit is the bitwise "OR" of all the 16 possible high priority interrupts.

IRQLOW:
This status bit is the bitwise "OR" of all the interrupts contained in the Interrupt Status 1,2,3,4, and 5 Registers.

INTVECT(3:0):
These 4 status bits are a vector pointing to the highest priority pending interrupt from the set of 16 possible high priority interrupt events. The set of high priority interrupts is listed below, where level 15 is the highest priority and level 1 is the lowest priority. Level 0 corresponds to the standard interrupt handler.

priority level high	priority interrupt source
15	POSPWROKINT or NEGPWROKINT
14	ALARMINT
13	PERINT
12	Reserved
11	UARTARXINT
10	UARTBRXINT
9	Reserved
8	IOPOSINT(6) or IOPOSINT(5)
7	Reserved
6	IONEGINT(6) or IONEGINT(5)
5	Reserved
4	SNDDMACNTINT
3	TELDMACNTINT
2	CHIDMACNTINT
1	IOPOSINT(0) or IONEGINT(0)

8.3.7 Clear Interrupt 1 Register

OFFSET = $100: write-only

The Clear Interrupt 1 Register bit locations are mapped on a one to one basis with the Interrupt Status 1 Register. A logic "1" written to a specific bit location will clear the corresponding bit in the Status Register.

8.3.8 Clear Interrupt 2 Register

OFFSET = $104: write-only

The Clear Interrupt 2 Register bit locations are mapped on a one to one basis with the Interrupt Status 2 Register. A logic "1" written to a specific bit location will clear the corresponding bit in the Status Register.

8.3.9 Clear Interrupt 3 Register

OFFSET = $108: write-only

The Clear Interrupt 3 Register bit locations are mapped on a one to one basis with the Interrupt Status 3 Register. A logic "1" written to a specific bit location will clear the corresponding bit in the Status Register.

8.3.10 Clear Interrupt 4 Register

OFFSET = $10C: write-only

The Clear Interrupt 4 Register bit locations are mapped on a one to one basis with the Interrupt Status 4 Register. A logic "1" written to a specific bit location will clear the corresponding bit in the Status Register.

8.3.11 Clear Interrupt 5 Register

OFFSET = $110: write-only

The Clear Interrupt 5 Register bit locations are mapped on a one to one basis with the Interrupt Status 5 Register. A logic "1" written to a specific bit location will clear the corresponding bit in the Status Register.

8.3.12 Enable Interrupt 1 Register

OFFSET = $118: read/write

The Enable Interrupt 1 Register bit locations are mapped on a one to one basis with the Interrupt Status 1 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register. This register is not cleared upon reset; however, the GLOBALEN global interrupt enable bit is cleared upon reset, therefore disabling all interrupts.

8.3.13 Enable Interrupt 2 Register

OFFSET = $11C: read/write

The Enable Interrupt 2 Register bit locations are mapped on a one to one basis with the Interrupt Status 2 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register. This register is not cleared upon reset; however, the GLOBALEN global interrupt enable bit is cleared upon reset, therefore disabling all interrupts.

8.3.14 Enable Interrupt 3 Register

OFFSET = $120: read/write

The Enable Interrupt 3 Register bit locations are mapped on a one to one basis with the Interrupt Status 3 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register. This register is not cleared upon reset; however, the GLOBALEN global interrupt enable bit is cleared upon reset, therefore disabling all interrupts.

8.3.15 Enable Interrupt 4 Register

OFFSET = $124: read/write

The Enable Interrupt 4 Register bit locations are mapped on a one to one basis with the Interrupt Status 4 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register. This register is not cleared upon reset; however, the GLOBALEN global interrupt enable bit is cleared upon reset, therefore disabling all interrupts.

8.3.16 Enable Interrupt 5 Register

OFFSET = $128: read/write

The Enable Interrupt 5 Register bit locations are mapped on a one to one basis with the Interrupt Status 5 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register. This register is not cleared upon reset; however, the GLOBALEN global interrupt enable bit is cleared upon reset, therefore disabling all interrupts.

8.3.17 Enable Interrupt 6 Register

OFFSET = $12C: read/write

The Enable Interrupt 6 Register bit locations are mapped on a one to one basis with the Interrupt Status 6 Register. A logic "1" written to a specific bit location will enable the corresponding interrupt in the Status Register.

Bit	Label	RESET	Read/Write
31-19	Reserved		
18	GLOBALEN	0	R/W
17	IRQPRITEST	0	R/W
16	IRQTEST	0	R/W
15-0	PRIORITYMASK(15:0)	X	R/W

GLOBALEN:

This is used as a global interrupt enable for all interrupts, and is cleared upon reset, therefore disabling all interrupts.

IRQPRITEST:
This bit is used for IC testing and should not be set.

IRQTEST:
This bit is used for IC testing and should not be set.

PRIORITYMASK(15:0):
These bits are an enable mask for the 16 possible high priority interrupts, where bit 15 is the highest priority. A logic "1" written to a specific bit location will enable the corresponding high priority interrupt.

Chapter 9 I/O Module

PR31700 V0.3

This section describes the Input/Output (IO) Module, which contains logic for reading and/or writing of the bi-directional general purpose IO and multi-function IO pins.

9.1 Overview

The IO Module within the PR31700 contains support for reading and writing the 7 bi-directional general purpose IO pins and the 32 bi-directional multi-function IO pins. Each of the general purpose IO pins can be independently configured as an input or output port. Each port can generate a separate positive and negative edge interrupt and each port can also be independently configured to use a debouncer. Of the 136 signal pins found on the PR31700, 32 of them are multi-function and can be independently programmed either as IO ports or for an alternate standard/ normal function. This allows the PR31700 to support a flexible and wide range of system applications and configurations. As an IO port, any of these pins can be programmed as an input or output port, with the capability of generating a separate positive and negative edge interrupt.

9.1.1 Related Pins

IO(6:0): **INPUT/OUTPUT**

These pins are general purpose input/output ports. Each port can be independently programmed as an input or output port. Each port can generate a separate positive and negative edge interrupt. Each port can also be independently programmed to use a 16 to 24 msec debouncer.

MFIO(1:0): **INPUT/OUTPUT**

These pins are multi-function input/output ports. Each port can be independently programmed as an input or output port, or can be programmed for multi-function use to support vendor-dependent test signals (for debugging purposes only). Each port can generate a separate positive and negative edge interrupt. Note that 30 other multi-function pins are available for usage as multi-function input/output ports. These pins are named after their respective standard/normal function and are not listed here.

9.2 Implementation

9.2.1 Block Diagram

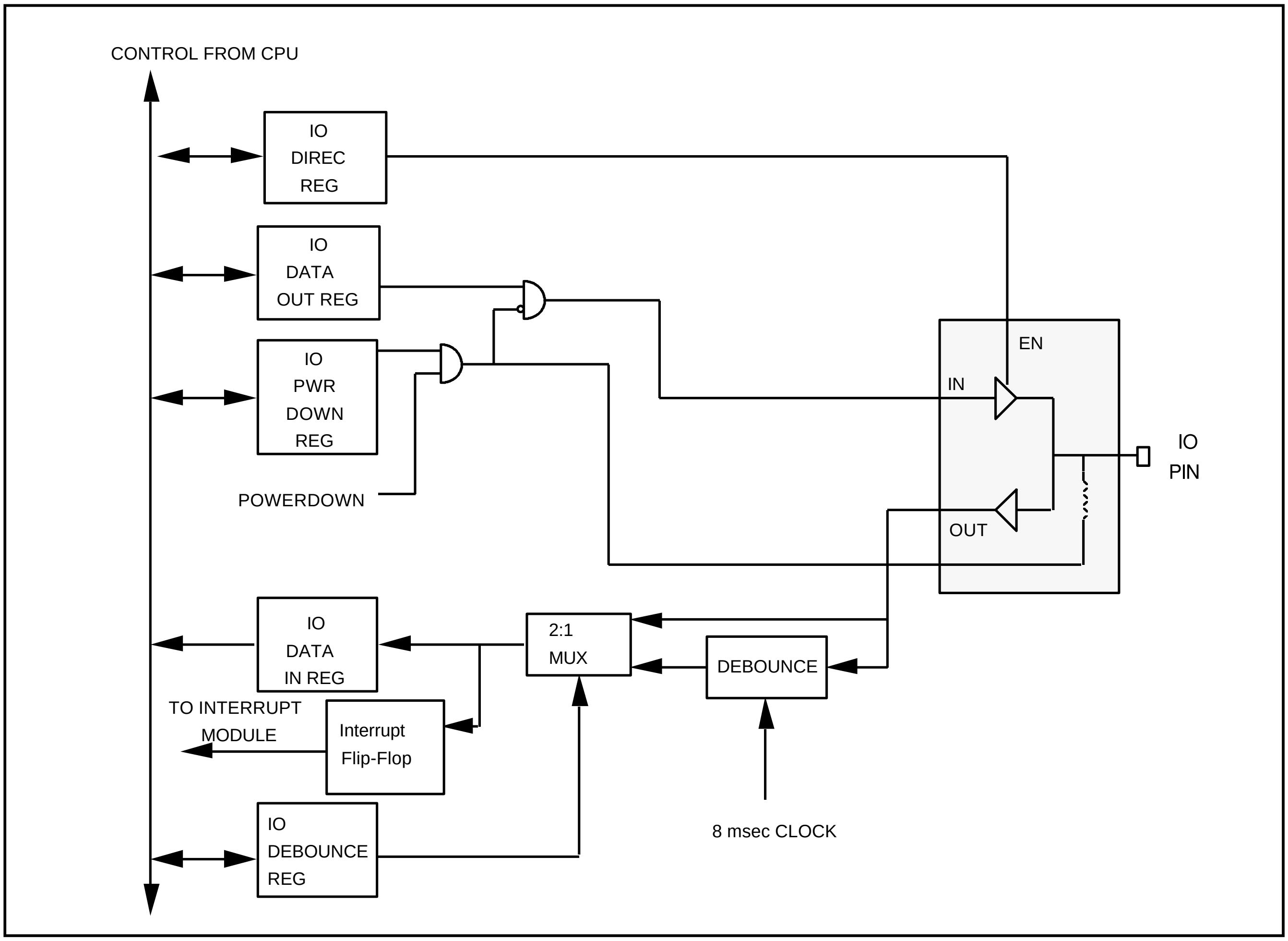


Figure 9-1. General Purpose IO Port Block Diagram

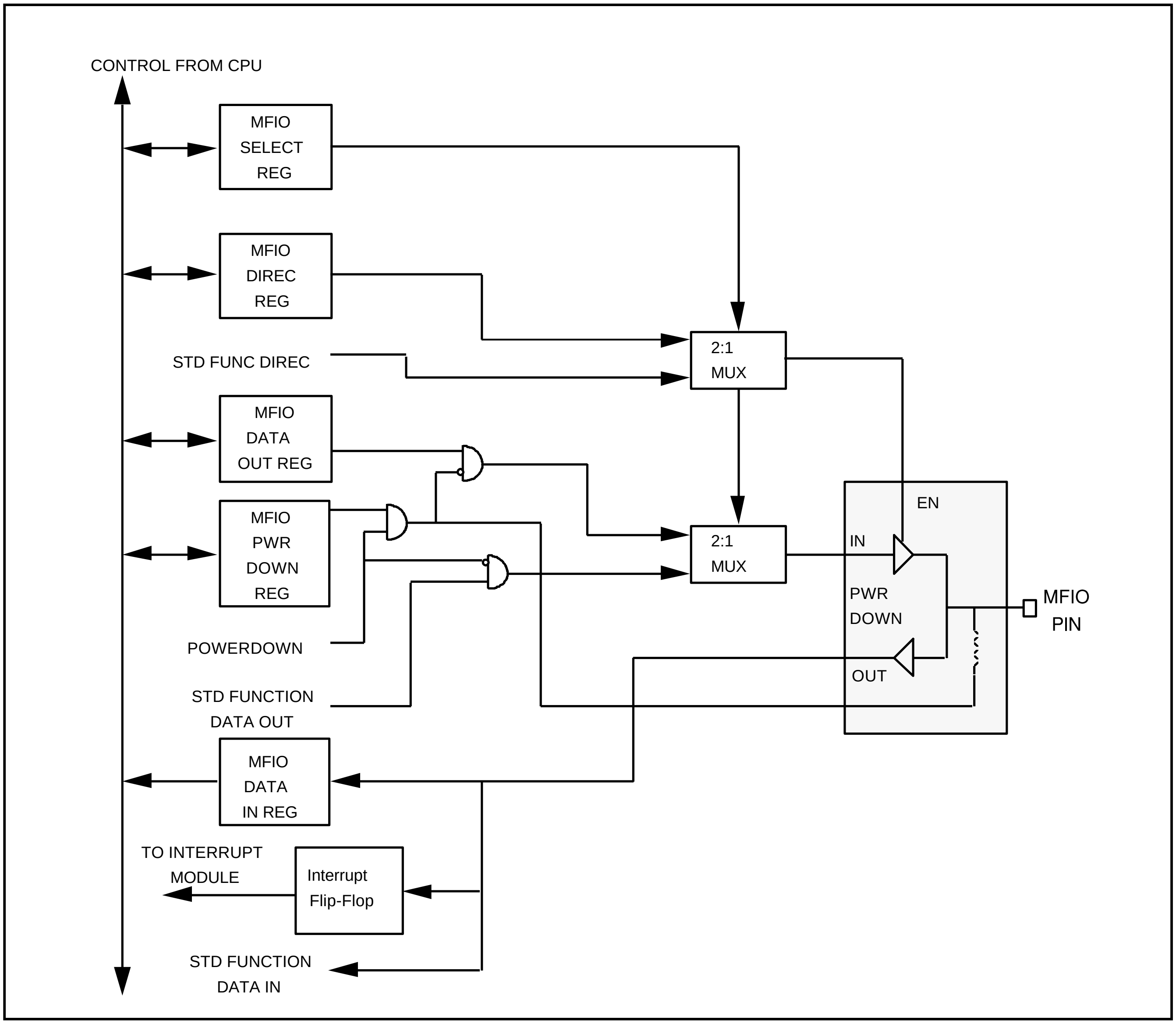


Figure 9-2. Multi-Function IO Port Block Diagram

9.2.2 General Purpose IO Ports

Each of the 7 general purpose IO ports can be independently programmed as an input or output port. Each port can generate a separate positive and negative edge interrupt. Figure 9-1 shows a block diagram of a general purpose IO port.

Each port consists of a bi-directional buffer connected to the appropriate PR31700 pin. For the input direction, the output signal from the input buffer is routed directly to a debounce circuit. This circuit performs a 16 to 24 msec debounce of the input signal. The debounce select control signal from the IO Debounce Select Register is used to select between the debouncer output and the direct signal from the port input buffer, which bypasses the debouncer. This selected signal is then routed to the IO Data Input Register and to the Interrupt Flip-Flop. Reading a specific bit location within the IO Data Input Register returns the logic state of the respective general purpose IO pin (either direct or debounced), regardless of whether that pin is configured as an output or input. If the pin is configured as an input, the value read is the logic state of the pin as driven by an external source. If the pin is configured as an output, the value read is the logic state of the pin as driven by the PR31700.

For the output direction, the input signal to the output buffer is routed from the appropriate bit within the IO Data Output Register. The output enable control signal for the tri-state output buffer is routed from the IO Direction Register. The IO Power-Down Register provides independent control bits for controlling the power-down state for each of the 7 general purpose IO ports. If a particular IO port is configured to power down, then the output will be gated low and a Pull-Down resistor will be enabled whenever the device powers down. Device power down occurs when either VCC3 or VCCON become de-asserted. The Pull-Down will prevent the input from floating if the general purpose IO port is configured as an input.

9.2.3 Multi-function IO Ports

Each of the 32 multi-function IO ports can be independently programmed as an input or output port or can be programmed for an alternate standard/normal function. This allows the PR31700 to support a flexible and wide range of system applications and configurations. As an IO port, any of these pins can generate a separate positive and negative edge interrupt. Figure 9-2 shows a block diagram of a multi-function IO port.

Each port consists of a bi-directional buffer connected to the appropriate PR31700 pin. For the input direction, the output signal from the input buffer is routed directly to the MFIO Data Input Register, to the Interrupt Flip-Flop, and to the appropriate module corresponding to the standard/normal function of the particular port, if this standard use is as an input signal. Reading a specific bit location within the MFIO Data Input Register returns the logic state of the respective multi-function IO pin, regardless of whether that pin is configured as an output or input. If the pin is configured as an input, the value read is the logic state of the pin as driven by an external source. If the pin is configured as an output, the value read is the logic state of the pin as driven by the PR31700.

For the output direction, the input signal to the output buffer is routed from a multiplexer which selects between the appropriate bit within the MFIO Data Output Register and the signal driven by the appropriate module corresponding to the standard/normal function of the particular port, if this standard use is as an output signal. The output enable control signal for the tri-state output buffer is also routed from a multiplexer which selects between the appropriate bit within the MFIO Direction Register and the direction signal driven by the appropriate module corresponding to the standard use of the particular port, if this standard use is as an output signal.

The MFIO Power-Down Register provides independent control bits for controlling the power-down state for each of the 32 multi-function IO ports. If a particular multi-function IO port is configured to power down, then the output will be gated low and a Pull-Down resistor will be enabled whenever the device powers down. Device power down occurs when either VCC3 or VCCON become de-asserted. The Pull-Down will prevent the input from floating if the multi-function IO port is configured as an input.

As mentioned previously, each of the multi-function IO ports can be independently programmed for routing of signals for a standard/normal function to or from any given multi-function IO port. Table 9-1 lists each of these standard functions for each multi-function IO port. Note that depending on the reset state for the respective MFIO Select Register bits, the pin function is configured either as a multi-function port or as a standard function port.

Table 9-1. Multi-Function IO Ports Versus Standard Function

PR31700 pin	standard function (I = input, O = output)	multi-function IO port	multi-function select (reset state)	Power Down Control Powerdown = /(vccon & vcc3) (reset state)
chifs	chifs (I/O)	mio [31]	miosel [31] (1)	Powerdown & miopd [31] (1)
chiclk	chiclk (I/O)	mio [30]	miosel [30] (1)	Powerdown & miopd [30] (1)
chidout	chidout (O)	mio [29]	miosel [29] (1)	Powerdown & miopd [29] (1)
chidin	chidin (I)	mio [28]	miosel [28] (1)	Powerdown & miopd [28] (1)
/dreq	/dreq (I)	mio [27]	miosel [27] (0)	Powerdown & miopd [27] (1)
/dgrnt	/dgrnt (O)	mio [26]	miosel [26] (0)	Powerdown & miopd [26] (0)
bc32k	bc32k (O)	mio [25]	miosel [25] (1)	Powerdown & miopd [25] (1)
txd	txd (O)	mio [24]	miosel [24] (0)	Powerdown & miopd [24] (0)
rxd	rxd (I)	mio [23]	miosel [23] (0)	Powerdown & miopd [23] (1)
/cs1	/cs1 (O)	mio [22]	miosel [22] (0)	Powerdown & miopd [22] (1)
/cs2	/cs2 (O)	mio [21]	miosel [21] (0)	Powerdown & miopd [21] (1)
/cs3	/cs3 (O)	mio [20]	miosel [20] (0)	Powerdown & miopd [20] (1)
/mccs0	/mccs0 (O)	mio [19]	miosel [19] (1)	Powerdown & miopd [19] (0)
/mccs1	/mccs1(O)	mio [18]	miosel [18] (1)	Powerdown & miopd [18] (0)
/mccs2	/mccs2 (O)	mio [17]	miosel [17] (1)	Powerdown & miopd [17] (0)
/mccs3	/mccs3 (O)	mio [16]	miosel [16] (1)	Powerdown & miopd [16] (0)
spiclk	spiclk (O)	mio [15]	miosel [15] (0)	Powerdown & miopd [15] (0)
spiout	spiout (O)	mio [14]	miosel [14] (0)	Powerdown & miopd [14] (0)
spiin	spiin (I)	mio [13]	miosel [13] (0)	Powerdown & miopd [13] (1)
sibmclk	sibmclk (I/O)	mio [12]	miosel [12] (0)	Powerdown & miopd [12] (1)
/cardreg	/cardreg (O)	mio [11]	miosel [11] (1)	Powerdown & miopd [11] (1)
/cardiowr	/cardiowr (O)	mio [10]	miosel [10] (1)	Powerdown & miopd [10] (1)
/cardiord	/cardiord (O)	mio [9]	miosel [9] (1)	Powerdown & miopd [9] (1)
/card1csl	/card1csl (O)	mio [8]	miosel [8] (1)	Powerdown & miopd [8] (1)
/card1csh	/card1csh (O)	mio [7]	miosel [7] (1)	Powerdown & miopd [7] (1)
/card2csl	/card2csl (O)	mio [6]	miosel [6] (1)	Powerdown & miopd [6] (1)
/card2csh	/card2csh (O)	mio [5]	miosel [5] (1)	Powerdown & miopd [5] (1)
/card1wait	/card1wait (I)	mio [4]	miosel [4] (1)	Powerdown & miopd [4] (1)
/card2wait	/card2wait (I)	mio [3]	miosel [3] (1)	Powerdown & miopd [3] (1)
/carddir	/carddir (O)	mio[2]	miosel [2] (1)	Powerdown & miopd [2] (1)
mfio[1]	(master)	mio[1]	miosel [1] (1)	Powerdown & miopd [1] (1)
mfio [0]	(cpu_data_cyc)	mio [0]	miosel [0] (1)	Powerdown & miopd [0] (1)

9.2.4 Related Interrupts

MFIOPOSINT(31:0):
Issues an interrupt whenever any of the multi-function I/O pin(s) transition from a logic "0" to a logic "1". There are a total of 32 multi-function I/O pins (31 thru 0), each of which corresponds to the respective bit within the Interrupt Status 3 Register. Each multi-function I/O pin can independently trigger an interrupt.

MFIONEGINT(31:0):
Issues an interrupt whenever any of the multi-function I/O pin(s) transition from a logic "1" to a logic "0". There are a total of 32 multi-function I/O pins (31 thru 0), each of which corresponds to the respective bit within the Interrupt Status 4 Register. Each multi-function I/O pin can independently trigger an interrupt.

IOPOSINT(6:0):
Issues an interrupt whenever any of the general purpose I/O pin(s) transition from a logic "0" to a logic "1". There are a total of 7 general purpose I/O pins (6 thru 0), each of which corresponds to the respective bit within the Interrupt Status 5 Register. Each general purpose I/O pin can independently trigger an interrupt.

IONEGINT(6:0):
Issues an interrupt whenever any of the general purpose I/O pin(s) transition from a logic "1" to a logic "0". There are a total of 7 general purpose I/O pins (6 thru 0), each of which corresponds to the respective bit within the Interrupt Status 5 Register. Each general purpose I/O pin can independently trigger an interrupt.

9.3 IO Registers

9.3.1 IO Control Register

OFFSET = $180:

Bit	Label	RESET	Read/Write
31	Reserved		
30-24	IODEBSEL(6:0)	X	R/W
23	Reserved		
22-16	IODIREC(6:0)	0	R/W
15	Reserved		
14-8	IODOUT(6:0)	X	R/W
7	Reserved		
6-0	IODIN(6:0)	-	R

IODEBSEL(6:0):

These bits select the debounce mode for the 7 general purpose IO pins. Setting a specific bit location to a logic "1" causes the respective general purpose IO port signal to be filtered by a debounce circuit for the input signal direction. Clearing a specific bit location to a logic "0" causes the respective general purpose IO port signal to bypass the input debounce circuit.

IODIREC(6:0):

These bits control the direction of the 7 general purpose IO pins. Setting a specific bit location to a logic "1" configures the respective general purpose IO pin to be an output. Clearing a specific bit location to a logic "0" configures the respective general purpose IO pin to be an input.

IODOUT(6:0):

These bits correspond to the data output values for the 7 general purpose IO pins. Setting or clearing a specific bit location controls the logic state of the respective general purpose IO pin, if that pin is configured as an output.

IODIN(6:0): read-only

These bits correspond to the data input values for the 7 general purpose IO pins. Reading a specific bit location returns the logic state of the respective general purpose IO pin, regardless of whether that pin is configured as an output or input.

9.3.2 MFIO Data Output Register

OFFSET = $184:

Bit	Label	RESET	Read/Write
31-0	MFIODOUT(31:0)	X	R/W

MFIODOUT(31:0):

These bits correspond to the data output values for the 32 multi-function IO ports. Setting or clearing a specific bit location controls the logic state of the respective multi-function IO port, if that pin is configured as an output.

9.3.3 MFIO Direction Register

OFFSET = $188:

Bit	Label	RESET	Read/Write
31-0	MFIODIREC(31:0)	0	R/W

MFIODIREC(31:0):

These bits control the direction of the 32 multi-function IO ports. Setting a specific bit location to a logic "1" configures the respective multi-function IO port to be an output. Clearing a specific bit location to a logic "0" configures the respective multi-function IO port to be an input.

9.3.4 MFIO Data Input Register

OFFSET = $18C: read-only

Bit	Label	RESET	Read/Write
31-0	MFIODIN(31:0)	-	R

MFIODIN(31:0): read-only

These bits correspond to the data input values for the 32 multi-function IO pins. Reading a specific bit location returns the logic state of the respective multi-function IO pin, regardless of whether that pin is configured as an output or input.

9.3.5 MFIO Select Register

OFFSET = $190:

Bit	Label	RESET	Read/Write
31-0	MFIOSEL(31:0)	$F20F0FFF	R/W

MFIOSEL(31:0):

These bits correspond to the select mode for the 32 multi-function IO pins. Setting a specific bit location to a logic "1" configures the respective multi-function IO pin to be selected as a bi-directional IO port. Clearing a specific bitlocation to a logic "0" configures the respective multi-function IO pin to be selected as a standard/normal pin function.

9.3.6 IO Power-Down Register

OFFSET = $194:

Bit	Label	RESET	Read/Write
31-7	Reserved		
6-0	IOPD(6:0)	$7F	R/W

IOPD(6:0):

These bits control the power-down state for the 7 general purpose IO pins. Setting a specific bit location to a logic "1" configures the respective general purpose IO pin to power down when the PR31700 powers down. Clearing a specific bit location to a logic "0" configures the respective general purpose IO pin to be in an active state at all times.

9.3.7 MFIO Power-Down Register

OFFSET = $198:

Bit	Label	RESET	Read/Write
31-0	MFIOPD(31:0)	$FAF03FFC	R/W

MFIOPD(31:0):

These bits control the power-down state for the 32 multi-function IO ports. Setting a specific bit location to a logic "1" configures the respective multi-function IO port to power down when the PR31700 powers down. Clearing a specific bit location to a logic "0" configures the respective multi-function IO port to be in an active state at all times.

Chapter 10 IR Module

The Infrared (IR) Module contains logic and control registers used to interface to external analog circuitry that is used to either communicate via IR between devices or send IR pulses to a consumer device such as a TV or a VCR.

10.1 Overview

The IR Module consists of three parts. The first is the Consumer Interface, which provides an easy mechanism for generating pulses to either turn on or turn off a light emitting diode (LED). The pattern in which the diode is turned on and off is used to encode commands which emulate remote control key presses that the consumer device recognizes. The software is responsible for programming the IR Module parameters to emulate desired consumer remote control protocols and key presses. The Consumer Interface provides universal control of common consumer devices such as televisions, VCR's, hi-fi equipment, appliances, etc.

The second part of the IR Module is the Communication Interface. The IR Module can be used in one of two data communication modes: using a frequency shift keyed (FSK) modulation scheme or using a scheme based on the protocol standardized by the InfraRed Data Association (IRDA). The FSK modulation scheme currently supports data communication between any device or peripheral supporting the IRDA standard. This allows communication with IRDA devices such as FAX machines, copiers, printers, etc. The Communication Interface is implemented using a standard UART protocol with an adjustable baud rate. This interface operates half-duplex in one direction at a time. The UART-B Module is used for both the FSK and IRDA data communication modes (see Section 16 for a detailed description of the UART Module).

For the FSK mode transmit direction, external communication IR analog circuitry encodes the UART output data using an FSK modulation scheme. For the receive direction, external communication IR analog circuitry amplifies the received photo-diode signal and demodulates the FSK signal before feeding the UART input.

For the IRDA mode transmit direction, the UART output data directly drives the external analog LED circuit. For the receive direction, the received photo-diode signal is externally amplified before feeding the UART input. Additional control bits in the UART Module also allow for various narrow pulse options to support the IRDA standard.

The final part of the IR Module is the Carrier Detect State Machine Interface. This interface is used to periodically enable the receiver in the external communication IR analog circuitry to detect whether there is a valid carrier present. If a valid carrier is present then an interrupt will be issued so the system can begin receiving data.

10.1.1 Related Pins

IROUT: **OUTPUT**

This pin is the UART transmit signal from the UARTB Module or the Consumer IR output signal if Consumer IR mode is enabled.

IRIN: **INPUT**

This pin is the UART receive signal to the UARTB Module.

RXPWR: **OUTPUT**

This pin is the receiver power output control signal to the external communication IR analog circuitry.

CARDET: **INPUT**

This pin is the carrier detect input signal from the external communication IR analog circuitry.

10.2 Consumer IR

10.2.1 Requirements

Consumer IR remote controllers send a burst of pulses whenever a button is pressed. The number of bursts along with the length of each burst define the key that has been pressed.

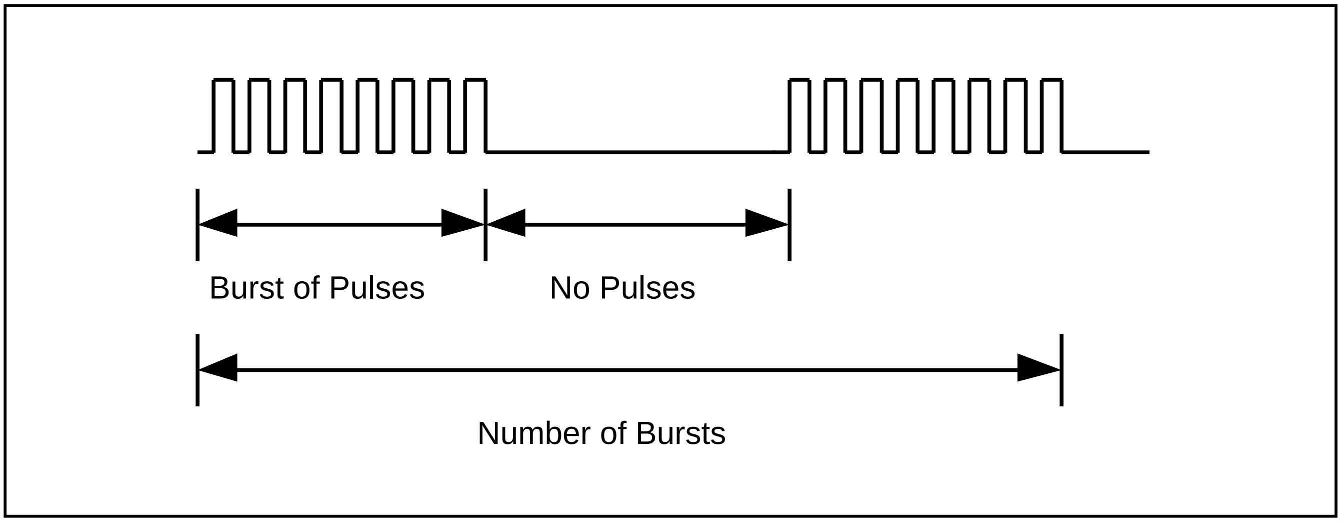


Figure 10-1. Consumer IR Pulses

The frequency of the pulses and the pulse width is fixed for a given remote control but can vary between different remote controls from different manufacturers. The number of pulses in a burst, the amount of time when there are no pulses, and the number of times that the burst is repeated are used to distinguish which key has been pressed for a given remote control.

10.2.2 Implementation

The Consumer IR Interface is designed to assist the CPU in shaping the pulses that are required to emulate a key press on a remote control. The logic has been implemented such that the frequency (typical consumer remote controllers use a carrier frequency in the 30 to 50 KHz range) and width of a pulse can be programmed by the CPU for a given remote control. The processor can then vary the number of bursts in a pulse and the amount of time when there are no pulses on a byte by byte basis in order to emulate the various key presses.

10.2.3 Block Diagram

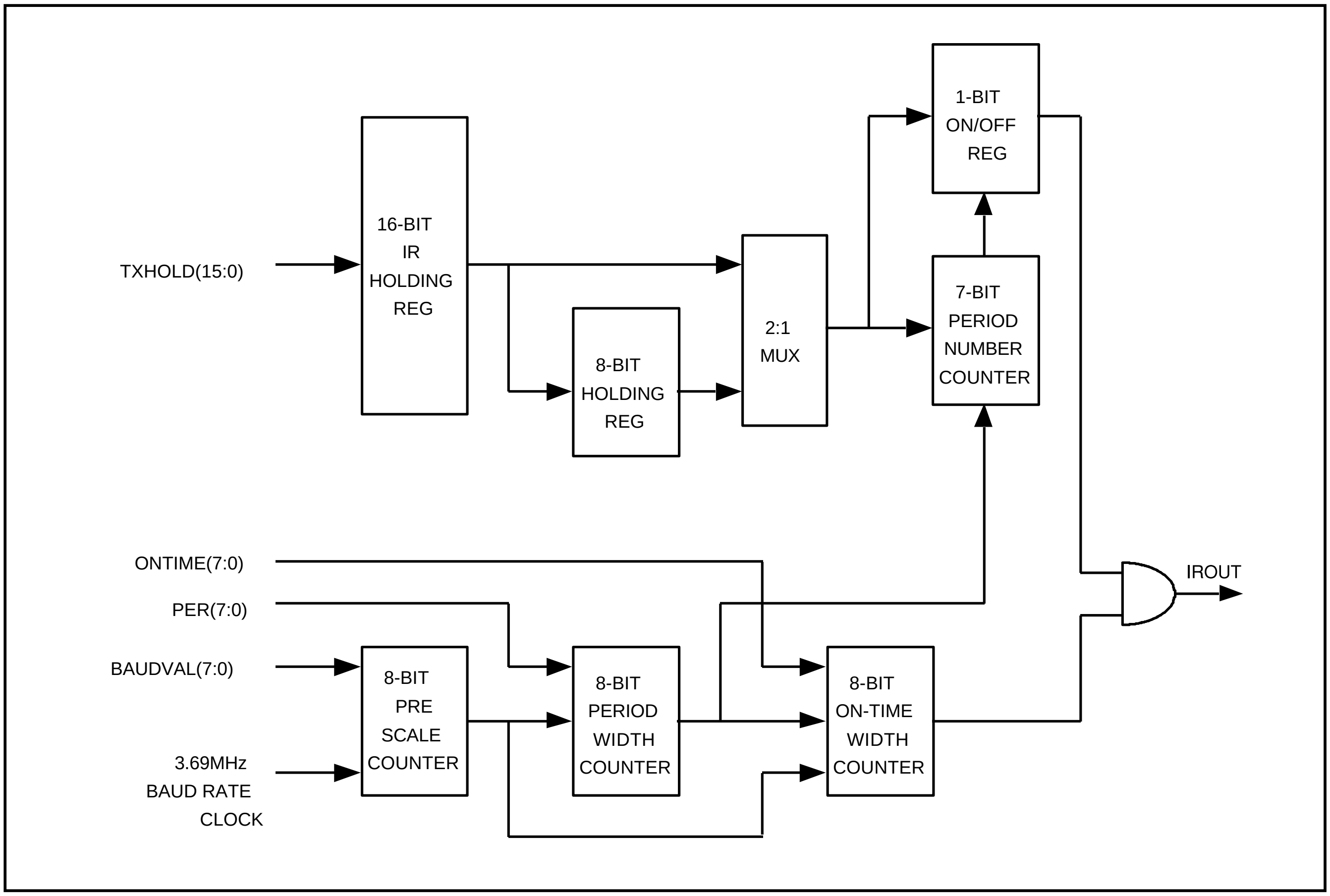


Figure 10-2. Functional Description

10.2.4 Functional Description

The Consumer IR logic remains idle until the CPU activates the IR logic by writing to the IR Control 1 Register to enable the Consumer mode (ENCONSM = "1"). Prior to enabling the Consumer mode, the CPU must configure the Baud Value (BAUDVAL), Period Value (PER), and OnTime Value (ONTIME) fields located in the IR Control 2 Register. BAUDVAL(7:0) is used to pre-scale the 3.6864 MHz baud rate clock since typical consumer IR data rates are in the KHz range instead of the MHz range. PER(7:0) is used to define the period of the pulses for a burst, representing the number of pre-scaled counts per period. ONTIME(7:0) defines the amount of time that the pulse is on for the given period, representing the number of pre-scaled counts for which the pulse is on. ONTIME should always be less than PER or the IROUT signal will never transition low.

Transmit data is written by the CPU into a 16-bit holding register. When the 7-bit Period Number Counter (down-counter) reaches zero it will load the lower 7 bits of the holding register's upper byte into the Period Number Counter and the upper bit into the 1-bit ON/OFF Register. If the upper bit is set to a "1" then the IROUT signal will be a "1" until the OnTime Width Counter reaches it's terminal count. The OnTime Width Counter freezes once it reaches it's terminal count in order to prevent the signal from transitioning until the counter is loaded again. The IROUT signal will continue to transition according to the Period Value and OnTime Value until the 7-bit Period Number Counter reaches zero. At this time the lower 7 bits of the holding register's lower byte is loaded into the Period Number Counter and the upper bit of the lower byte is loaded into the 1-bit ON/OFF Register. If the upper bit is set to a "0" this time, then the IROUT signal will remain low for as many period counts as are defined by the value loaded into the 7-bit Period Number Counter. Thus the CPU controls the pulse generation by defining a number of pulses to burst (upper bit set to "1", lower 7-bit data defines the number of pulses) or an amount of time when there are no pulses (upper bit set to "0", lower 7-bit data defines the number of pulse times when IROUT is "0") on a byte by byte basis. The OnTime Value and Period Value timing relationships are shown in Figure 10-3.

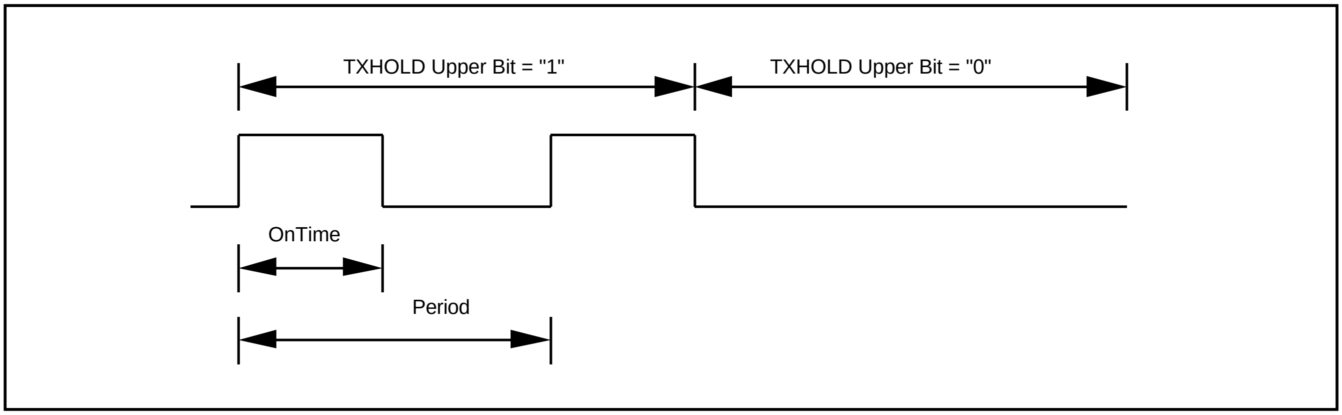


Figure 10-3. Consumer IR Signal Generation

10.2.5 Related Interrupts

IRCONSMINT:

Whenever the upper byte of data is loaded into the 7-bit Period Number Counter, the lower byte of data is loaded into an intermediate holding register. At this time the 16-bit IR Holding Register is empty. The IRCONSMINT interrupt is then set to inform the CPU that the IR Holding Register is available. The CPU must fill the next word of data into the IR Holding Register before the 7-bit Period Number Counter is finished counting the periods for both the upper and lower bytes of data.

10.3 Two-Way Communication Via IR

10.3.1 Requirements

The Communication Interface is used for two-way data communication between any device or peripheral supporting the IRDA standard.

The Communication Interface is implemented using a standard UART protocol, consisting of a start bit, 8 bits of data (LSB first), then a stop bit. The data rate is adjustable using a programmable baud rate counter. The Communication Interface consists of both a transmitter and a receiver and the UART is operated half-duplex in one direction at a time. The UART-B Module is used for both the FSK and IRDA data communication modes and contains DMA support for transmit and receive data (see Section 16 for a detailed description of the UART Module).

For the FSK mode transmit direction, external communication IR analog circuitry encodes the UART-B output data using an FSK modulation scheme, which modulates at two different frequencies depending on whether the data is a "1" or a "0". The carrier frequency is 1375 KHz and the mark and space frequencies are 1325 KHz and 1425 KHz, respectively. For the receive direction, external communication IR analog circuitry amplifies the received photo-diode signal and demodulates and decodes the FSK signal before feeding the UART-B input. The transmitter and receiver portions of the external communication IR analog circuitry contain individual power-down control for power management purposes. The FSK communication mode operates at a data rate of 2400 to 36000 bps at 3 meters.

For the IRDA mode transmit direction, the UART-B output data directly drives the external analog LED circuit. For the receive direction, the received photo-diode signal is externally amplified before feeding the UART-B input. Additional control bits in the UART Module also allow for various narrow pulse options to support the IRDA standard. When configured for these options, the UART transmit output pulses are narrower than normal and the UART receiver circuit also expects the same narrow pulse format for the input data. The IRDA communication mode operates at a data rate up to 115 Kbps at 1 meter.

10.3.2 Block Diagram

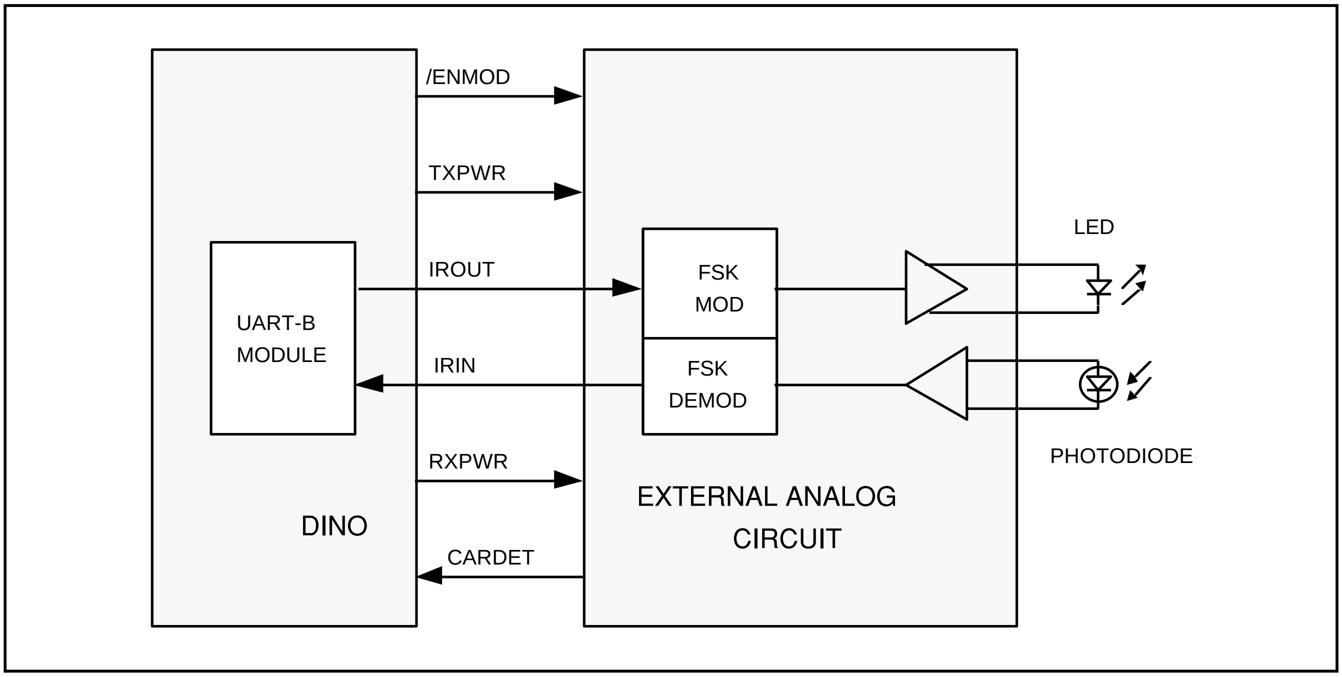


Figure 10-4. FSK Communication Circuit

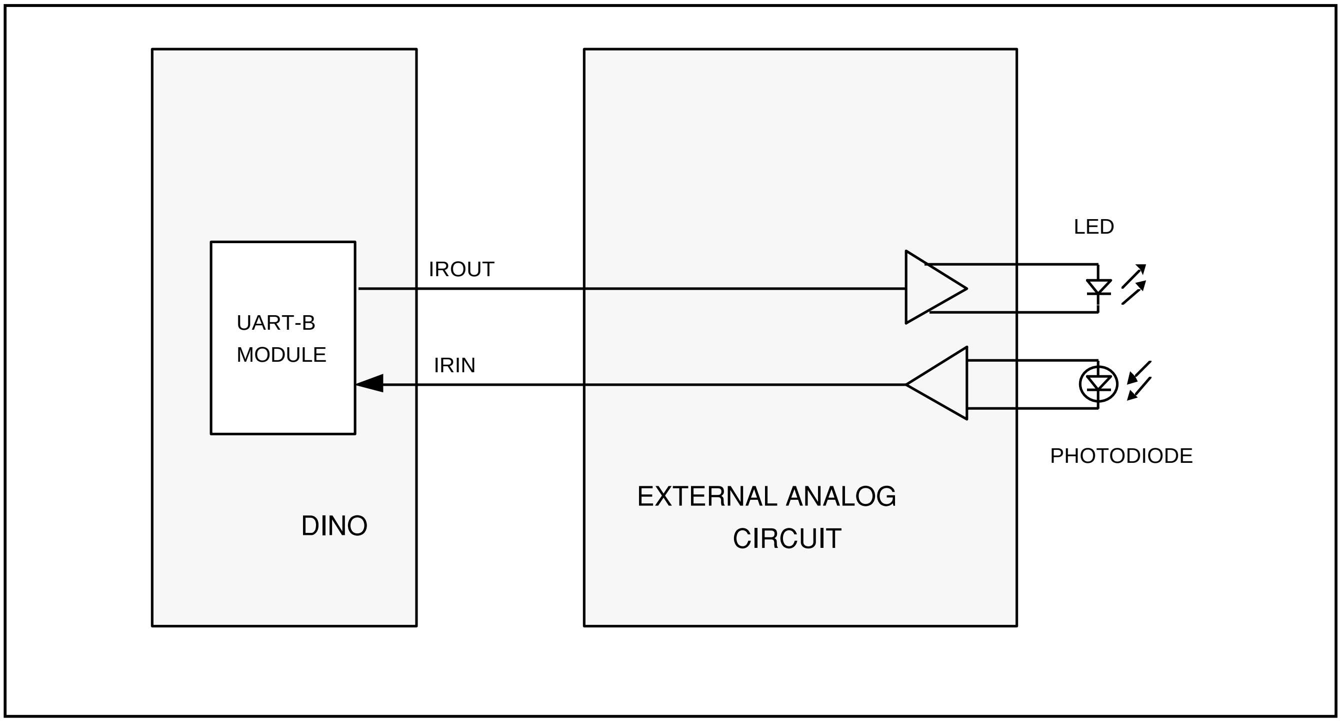


Figure 10-5. IRDA Communication Circuit

10.4 Carrier Detect State Machine

10.4.1 Requirements

The external communication IR analog circuitry contains a circuit which detects if there is a valid carrier present. It is not desirable to leave this circuit enabled all the time because it dissipates too much power. Thus the circuit is powered on and off with the RXPWR pin. It is possible for the CPU to periodically enable the RXPWR pin, then wait to see if a carrier is present (CARDET pin = "1"), but this requires a lot of processor overhead. In order to alleviate the overhead, the IR Module contains a circuit that periodically enables the RXPWR pin, waits a specified period of time, samples the CARDET pin, then disables the RXPWR pin. If the CARDET pin is a "1" when sampled then an interrupt is issued to the CPU indicating that there is a valid carrier present. After enabling the RXPWR pin the CARDET pin cannot be sampled for a specified amount of time because the Carrier Detect Logic in the external communication IR analog circuitry takes time to settle. The time between successive enabling of the RXPWR pin is programmable depending on the desired sample rate.

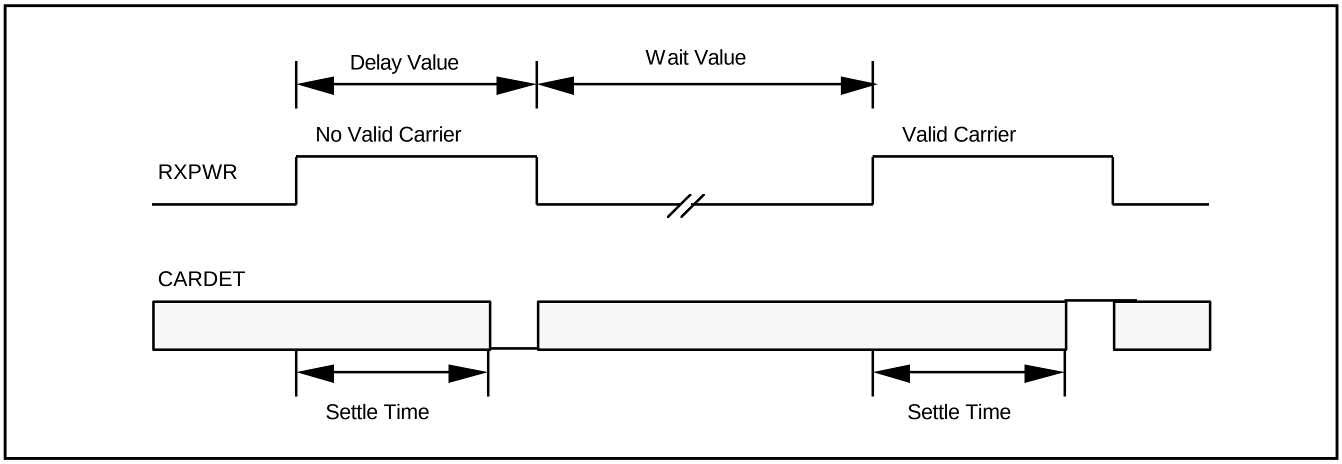


Figure 10-6. Carrier Detect State Timing

10.4.2 Block Diagram

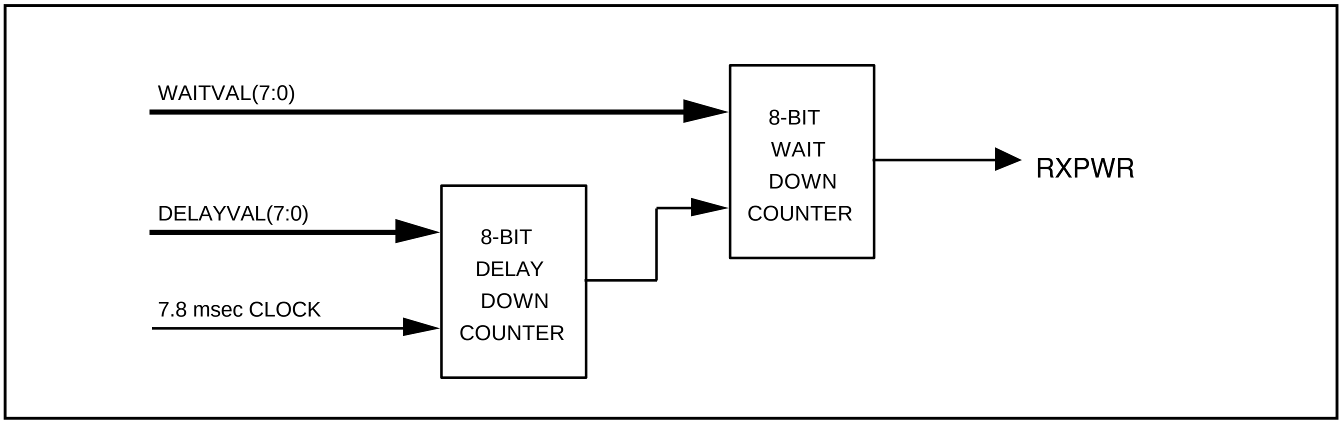


Figure 10-7. Carrier Detect Block Diagram

10.4.3 Functional Description

DELAYVAL(7:0) is loaded into the 8-bit Delay Down Counter whenever the counter reaches a count of zero. The terminal count of the Delay Down Counter enables the 8-bit Wait Down Counter which loads WAITVAL(7:0) whenever the Wait Down Counter reaches a count of zero. The RXPWR pin is asserted whenever the Wait Down Counter reaches a count of zero. Thus the period at which RXPWR is periodically asserted is derived by:

(DELAYVAL + 1) * (WAITVAL + 1) * 7.8 msec

The RXPWR pin will remain asserted until the Wait Down Counter counts again which will occur at:

(DELAYVAL + 1) * 7.8 msec

At the same time the RXPWR pin is deasserted the CARDET pin is sampled and an interrupt is set if a valid carrier is present.

10.4.4 Related Interrupts

CARSTINT:
This interrupt is set whenever the Carrier Detect State Machine samples the CARDET pin = "1" just before turning off the RXPWR pin.

POSCARINT:
This interrupt is set whenever CARDET pin transitions from a logic "0" to a logic "1".

NEGCARINT:
This interrupt is set whenever CARDET pin transitions from a logic "1" to a logic "0".

10.5 IR Registers

10.5.1 IR Control 1 Register

OFFSET = $0A0:

Bit	Label	RESET	Read/Write
31-25	Reserved		
24	CARDET	–	R
23-16	BAUDVAL(7:0)	X	R/W
15-5	Reserved		
4	TESTIR	0	R/W
3	DTINVERT	0	R/W
2	RXPWR	0	R/W
1	ENSTATE	0	R/W
0	ENCONSM	0	R/W

CARDET: **read-only**
This bit provides the status of the CARDET (carrier detect) input pin.

BAUDVAL(7:0):
These bits are used to pre-scale the 3.6864 MHz baud rate clock. The bits are loaded into an 8-bit down-counter that counts at the 3.6864 MHz clock rate. The resulting baud clock is used to clock the Period Width Counter and OnTime Width Counter.

TESTIR:
This bit is used to speed up the counters for IC testing and should not be set.

DTINVERT:
Setting this bit to a logic "1" will cause the IROUT signal to be active low instead of active high.

RXPWR:
This bit is connected to the RXPWR pin.

ENSTATE:
Setting this bit to a logic "1" enables the Communication IR carrier detect state machine.

ENCONSM:
Setting this bit to a logic "1" enables the Consumer IR function.

10.5.2 IR Control 2 Register

OFFSET = $0A4: write-only

Bit	Label	RESET	Read/Write
31-24	PER(7:0)	X	W
23-16	ONTIME(7:0)	X	W
15-8	DELAYVAL(7:0)	X	W
7-0	WAITVAL(7:0)	X	W

PER(7:0): **write-only**
These bits are used to define the period of the Consumer IR pulses. The bits are loaded into an eight bit down-counter that counts at the rate defined by the Baud Value. The resulting period is given by the following equation.

Period = (PER + 1) * (BAUDVAL + 1) * (1/3.6864 MHz)

ONTIME(7:0): **write-only**
These bits are used to define the On Time of the Consumer IR pulses. These bits are loaded into an eight bit down-counter that counts at the rate defined by the Baud Value. The resulting On Time is given by the following equation.

On Time = ONTIME * (BAUDVAL + 1) * (1/3.6864 MHz)

DELAYVAL(7:0): **write-only**

These bits are used to define the amount of time that the RXPWR pin will be a logic "1" when the Carrier State Machine logic is enabled. The bits are loaded into an 8-bit down-counter that counts at a 7.8 ms rate. Thus the time is programmable from 7.8 ms to 2.0 seconds.

Delay Time = (DELAYVAL + 1) * 7.8 ms

WAITVAL(7:0): **write-only**

These bits are used to define the amount of time to wait before asserting the RXPWR pin to a logic "1" when the Carrier State Machine logic is enabled. The bits are loaded into an 8-bit down-counter that counts whenever the DELAYVAL counter reaches a count of zero. Thus the time is programmable from 7.8 ms to 511 seconds.

Wait Time = (DELAYVAL + 1) * (WAITVAL + 1) * 7.8 ms

10.5.3 IR Holding Register

OFFSET = $0A8: write-only

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	TXHOLD(15:0)	X	W

TXHOLD(15:0): **write-only**

These bits are used to define the number of pulses in a burst or the amount of time to keep the IROUT pin low. The lower seven bits of the upper byte and the lower seven bits of the lower byte are used as values to load into a 7-bit counter that counts the number of periods as defined by the Period Value. The upper bit in each byte if set to a logic "1" causes pulses on the IROUT pin and if set to a logic "0" causes the IROUT pin to remain low for a given count. Before setting the ENCONSM bit the TXHOLD register must be pre-loaded with the first valid word of data.

Chapter 11
Power Module

This section describes the Power Module logic used to interface to the System Power Supply and to control internal PR31700 power functions.

11.1 Overview

The System Power Supply must provide several power supplies to various parts of the system. The software can control turning on or off the various supplies along with the PWRCS signal. The software will only turn on the supplies that are required to perform a particular task. PR31700 must always be provided power as long as there is a good Main or Backup battery in the system, or if a Battery Charger is plugged in. Similarly, there may be other components in the system that must always be powered. These components, along with the PR31700, should be supplied power from the VSTANDBY signal.

DRAM or SDRAM should be supplied power from the VCCDRAM signal. The VCCDRAM signal should remain low when VSTANDBY is first asserted, until PWRCS is asserted to turn on the system for the first time. Once the system boots for the first time, the software can decide not to turn off VCCDRAM when the system is subsequently turned off, in order to preserve the contents of memory.

Much of the rest of the system, including ROM, the UCB1200, and system buffers are powered by VCC3. The VCC3 signal is only on when the system is on. Other switchable power supplies are provided for components such as the LCD, PCMCIA cards, etc. Each of these supplies will normally be turned off by the software when powering down the system. When the system is on, these supplies will only be turned on if the particular module is being used.

In addition to the System Power Supply control, the Power Module also contains logic to support the PR31700 initialization, Clock Enable function, CPU Stop Mode, Stop Timer, and power failure logic. Each of these are described in more detail in the following sub-sections.

11.1.1 Power Signals

VSTANDBY:
This signal provides power for the PR31700 and other components in the system that must never lose power. This signal should always be asserted if there is either a good Main Battery or Backup Battery, or if a Battery Charger is plugged in.

VCCDRAM:
This signal provides power for the DRAM and/or SDRAM. This supply must be off when VSTANDBY is first asserted, and remain off until the system is powered up by the assertion of PWRCS. When the software subsequently powers down the system it may choose to keep this supply on to preserve the contents of memory.

VCC3:
This signal provides power for the ROM, the UCB1200, system buffers, and other transient components in the system. This signal will be asserted by the System Power Supply when PWRCS is asserted and will always be turned off when the system is powered down.

In addition to these supplies, the System Power Supply will generate individual switchable power signals for the LCD, PCMCIA cards, and other devices used in the system. These different supplies will vary according to particular system implementations.

11.1.2 Related Pins

/PON: INPUT
This pin serves as the Power On Reset signal for the PR31700. This signal must remain low when VSTANDBY is asserted until VSTANDBY is stable. Once VSTANDBY is asserted, this signal should never go low unless all power is lost in the system.

ONBUTN: INPUT
This pin is used as the On Button for the system. Asserting this signal will cause PWRCS to set to turn the System Power Supply power on to the system. PWRCS will not assert if the PWROK signal is low.

PWRCS: OUTPUT
This pin is used as the chip select or the control signal for the System Power Supply. When the system is off, the assertion of this signal will cause the System Power Supply to turn VCCDRAM and VCC3 on to power up the system.

PWROK: INPUT
This pin provides a status from the System Power Supply that there is a good source of power in the system. This signal typically will be asserted if there is a Battery Charger supplying current or if the Main Battery is good and the Battery Door is closed. If PWROK is low when the system is powered off, PWRCS will not assert as a result of the user pressing the ONBUTN or an interrupt attempting to wake up the system. If the device is on when the PWROK signal goes low, the software will immediately shut down the system since power is about to be lost. When PWROK goes low, there must be ample warning so that the software can shut down the system before power is actually lost.

PWRINT: INPUT
This pin is used by the System Power Supply to alert the software that some status has changed in the System Power Supply and the software should read the status from the System Power Supply to find out what has changed. These will be low priority events, unlike the PWROK status, which is a high priority emergency case.

VCC3: INPUT
This pin provides the status of the power supply for the ROM, the UCB1200, system buffers, and other transient components in the system. This signal will be asserted by the System Power Supply when PWRCS is asserted, and will always be turned off when the system is powered down.

11.2 Description

11.2.1 Power On Reset

When VSTANDBY is asserted, the /PON signal must be kept low until VSTANDBY is stable, in order to provide a good Power On Reset to the PR31700. The assertion of /PON will set a RESET flip-flop, as shown in Figure11-1. The RESET flip-flop will in remain set until the ONBUTN is pressed for the first time. This functionality provides a long reset time to the PR31700 when VSTANDBY is first asserted. The long reset time is needed to allow the 32 KHz oscillator to stabilize. The RESET signal is used to initialize all registers and logic in the PR31700.

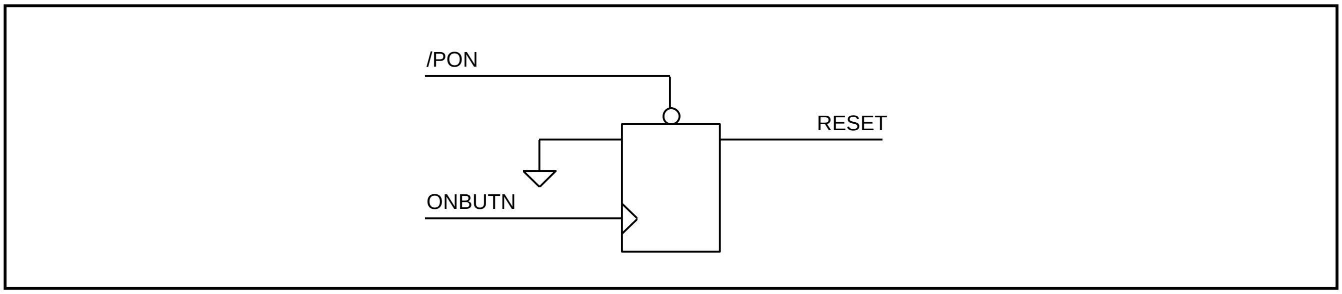


Figure 11-1. Reset Flip-Flop

11.2.2 Power Up

The Power Up sequence timing is shown in Figure11-2. When the ONBUTN is pressed, the PWRCS signal will be set if the PWROK signal is high. Once the PWRCS signal is set, the System Power Supply must assert VCCDRAM and VCC3 in order to power up the system. Once VCC3 is asserted, the high speed oscillator and PLL that generates the clock signals (nominally 36.864 MHz) to the CPU and BIU is asserted. When the oscillator is first enabled, the oscillation can vary wildly until the oscillator and PLL become stable. The stabilization time can vary between 1 ms and 10 ms, depending on the oscillator and PLL of a particular PR31700 implementation. In order to prevent the PR31700 from using the clock before the clock becomes stable, a circuit in the Power Module will provide a signal ENSYSCLK that will disable the clock until the oscillator and PLL become stable. The circuit uses the 32 KHz signal as its reference, since this is known to be good because of the long Power On Reset time. ENSYSCLK is asserted 16-24 ms after VCC3 is asserted or 4-6 ms after VCC3 is asserted, depending on the state of the SELC2MS control bit in the Power Control Register.

Once the ENSYSCLK signal asserts, the CPU will start up. If the PWROK signal is low when the ONBUTN is pressed, the PWRCS signal will not be asserted. The PWRCS signal will also be asserted when the system is off if an enabled interrupt is set and the PWROK signal is asserted. If an enabled interrupt is set when the system is off, but the PWROK signal is low, then the system will power up once the PWROK signal is asserted since the interrupt is latched. On the other hand, the ONBUTN is not a latched event unless the POSONBUTNINT or NEGONBUTNINT interrupts are enabled. Therefore, the ONBUTN will not latch and wake up the system once PWROK asserts.

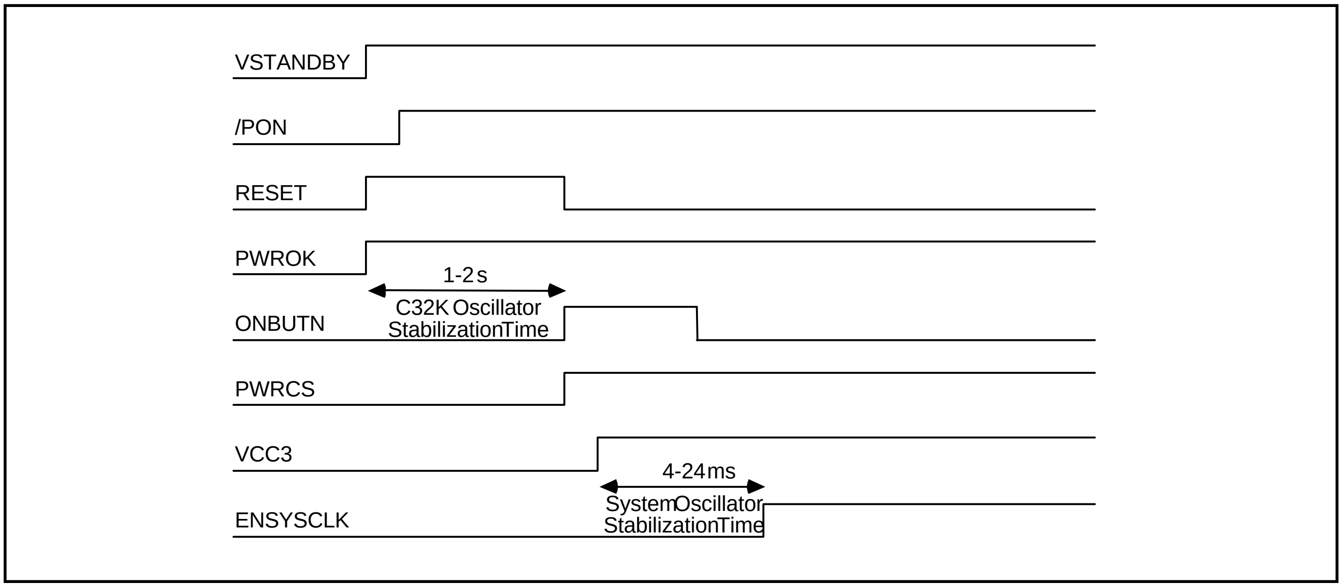


Figure 11-2. Power Up

11.2.3 Force Shut Down

The Power Module provides a feature called Force Shut Down that will force the system to power down if the PWROK signal goes low prior to the software clearing a bit. The timing for Force Shut Down is shown in Figure11-3. This feature can be enabled or disabled by setting the ENFORCESHUTDWN bit in the Power Control Register. This feature is useful if a particular battery will provide a PWROK indication when there is no load on the battery, but will quickly de-assert PWROK once VCC3 is asserted.

A signal called FORCESHUTDWN is asserted whenever PWRCS is asserted as a result of the ONBUTN being pressed or an enabled interrupt being set when the device is off. If the PWROK signal goes low prior to the software clearing the FORCESHUTDWN bit, the Power Module will simply deassert the PWRCS signal and the System Power Supply will turn off the VCC3 power supply. The instruction to clear the FORCESHUTDWN bit must be executed prior to bringing the memory interface out of self refresh mode. This implies that the instruction has to be pre-loaded into the cache before the previous system shut down, since the memory interface will exit self refresh mode once an address location is accessed in the system that is not in the cache or not to a PR31700 register. The very first time that the system is turned on after the assertion of VSTANDBY, it is not possible to support this function since the cache is not initialized. This is not a problem because the main purpose of this feature is to protect persistent memory, which will not exist when the device is turned on for the first time.

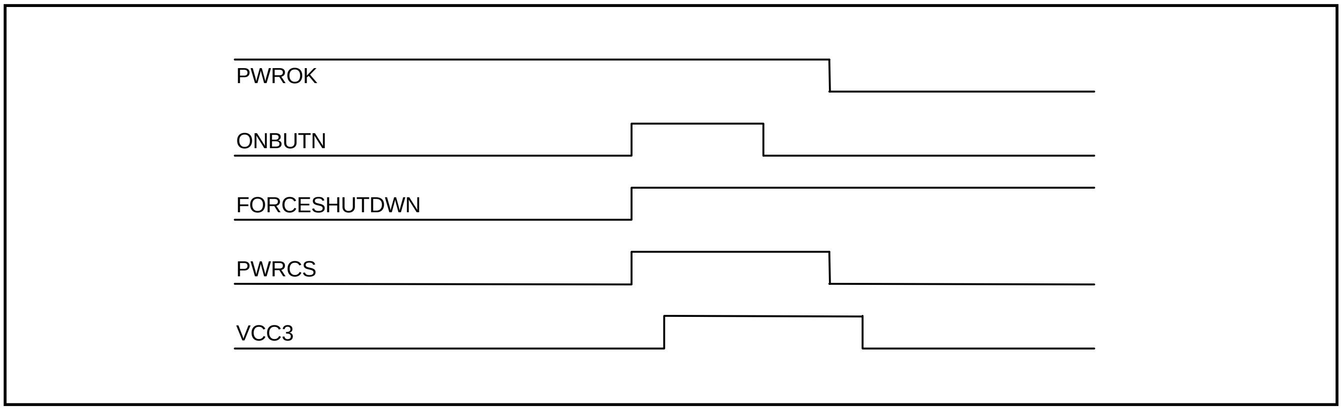


Figure 11-3. Force Shut Down

11.2.4 Power Down

The timing for Power Down is shown in Figure11-4. Prior to shut down the System Power Supply, all modules in PR31700 should be disabled and all clocks in the Clock Module should be turned off. Next, any interrupts that will wake up the system should be enabled. The software must then set the MEMPOWERDOWN bit in the Memory Configuration 4 Register in order to place the DRAM and/or SDRAM into self refresh mode and power down the memory interface. Next, the PWRCS and VCCON bits in the Power Control Register should be set low to inform PR31700 that the system is powering off. The VCCON signal is used by the PR31700 as an early warning that VCC3 is about to go off. The Power Module will not allow PWRCS and VCCON to clear if an enabled interrupt is set.

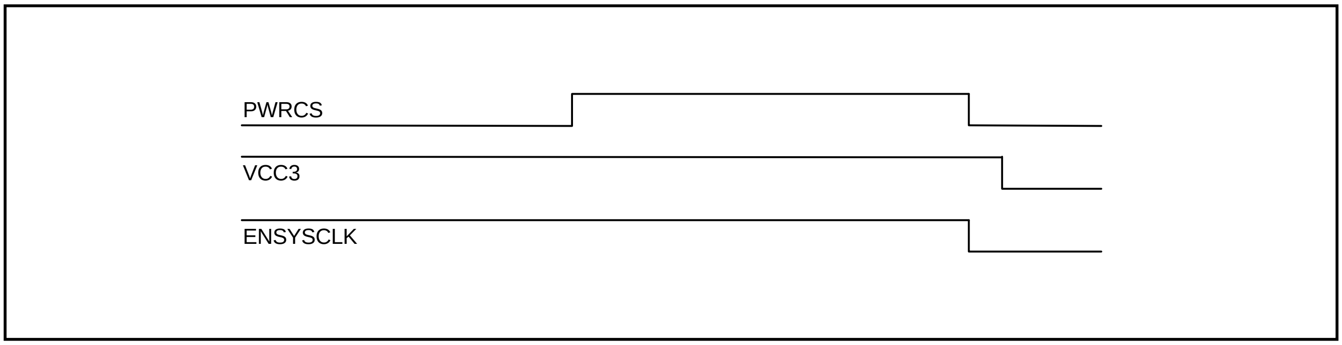


Figure 11-4. Power Down

11.2.5 Software Considerations

Once the memory interface is powered down by asserting the MEMPOWERDOWN bit in the Memory Configuration 4 Register, the software can no longer access address space other than the cache or PR31700 registers. Thus it is necessary that all instructions following the assertion of the MEMPOWERDOWN bit need to be executed from cache. Once the ENSYSCLK signal goes low, the clock to the CPU is stopped and the CPU will remain suspended at its last state. When the device powers back up, the ENSYSCLK will assert and the CPU will resume executing from where it left off. It is probably desirable to loop for some time to make sure that the battery voltage has stabilized and the PWROK signal will not suddenly go low as a result of a substantial load added to the battery. Once the loop has completed, the software must clear the FORCESHUTDWN bit prior to accessing any address that is not in the cache or PR31700 registers.

11.2.6 Stop Mode

In order to reduce the power consumption in the PR31700, it is desirable to stop the clock to the CPU when the CPU is inactive. This is possible since the CPU core is fully static and will simply re-start from where it left off when the clock is re-enabled. The software can stop the clock to the CPU by asserting the STOPCPU bit in the Power Control Register. Once this bit is set, the clock to the CPU will be stopped and remain that way until an enabled interrupt clears the STOPCPU bit. This bit should not be set when powering down the system because the ENSYSCLK signal is a more dominant clock shut down for the entire PR31700 chip.

11.2.7 Stop Timer

The Stop Timer is used as a coarse timer to bring the CPU out of Stop Mode. The Stop Timer consists of a 4-bit up counter and a comparator to generate the STPTIMERINT interrupt. The Stop Timer is enabled using the ENSTPTIMER control bit in the Power Control Register. When the ENSTPTIMER control bit is not set the counter will be reset to zero. Once the ENSTPTIMER bit is set the counter will count up using an 8 ms clock as the input. Once the counter reaches a value that is equal to the STPTIMERVAL(3:0) control bits, the STPTIMERINT interrupt is set. The Stop Timer will provide a maximum length of 120 ms, in steps of 8 ms.

11.2.8 Power Module Interrupts

STPTIMERINT:
This interrupt is set whenever the Stop Timer Counter counts up to the value set by the STPTIMERVAL(3:0) control bits.

POSPWRINT:
This interrupt is set when the PWRINT pin transitions from a logic "0" to a logic "1".

NEGPWRINT:
This interrupt is set when the PWRINT pin transitions from a logic "1" to a logic "0".

POSPWROKINT:
Issues an interrupt whenever the PWROK signal transitions from a logic "0" to a logic "1".

NEGPWROKINT:
Issues an interrupt whenever the PWROK signal transitions from a logic "1" to a logic "0".

POSONBUTNINT:
Issues an interrupt whenever the ONBUTN signal transitions from a logic "0" to a logic "1". If the DBNCONBUTN control bit is set then the interrupt will not set until the signal is debounced for 16-24 ms.

NEGONBUTNINT:
Issues an interrupt whenever the ONBUTN signal transitions from a logic "1" to a logic "0". If the DBNCONBUTN control bit is set then the interrupt will not set until the signal is debounced for 16-24 ms.

11.3 Power Registers

11.3.1 Power Control Register

OFFSET = $1C4:

Bit	Label	RESET	Read/Write
31	ONBUTN	–	R
30	PWRINT	–	R
29	PWROK	–	R
28-27	VIDRF[1:0]	0	R/W
26	SLOWBUS	0	R/W
25	DIVMOD	0	R/W
24-16	Reserved		
15-12	STPTIMERVAL(3:0)	X	R/W
11	ENSTPTIMER	0	R/W
10	ENFORCESHUTDWN	0	R/W
9	FORCESHUTDWN	0	R/W
8	FORCESHUTDWNOCC	0	R/W
7	SELC2MS	0	R/W
6	Reserved		
5	BPDBVCC3	0	R/W
4	STOPCPU	0	R/W
3	DBNCONBUTN	0	R/W
2	COLDSTART	1	R/W
1	PWRCS	0	R/W
0	VCCON	0	R/W

ONBUTN: **read-only**
This bit provides the status of the ONBUTN signal.

PWRINT: **read-only**
This bit provides the status of the PWRINT signal.

PWROK: **read-only**
This bit provides the status of the PWROK signal.

VIDRF(1:0):
Reduce Frequency control bits for Video clock for power management.

VIDRF[1:0]	VIDCLK
00	F/2
01	F/4
10	F/8
11	F/16

SLOWBUS:
An extra control bit is added to reduce system clock frequency further down to 1/16 for effective power management. Please refer to table in 6.2.2 Clock Module Description section.

DIVMOD:
Setting this bit will cause the internal clocks for each functional module to be independent of RF[1:0] . If this bit is set high, the source of internal clocks would be switched to FREECLK/XHFREE (independent system clocks) from CLK2X/CLK.

STPTIMERVAL(3:0):
When the Stop Timer is enabled the STPTIMERINT interrupt will be set when the counter is equal to the value set by these bits.

ENSTPTIMER:
This bit is used to enable the Stop Timer.

ENFORCESHUTDWN:
The FORCESHUTDWN signal is set whenever VCCON and PWRCS are asserted by either the ONBUTN being pressed or an enabled interrupt being set. If the PWROK signal goes low prior to the processor clearing the FORCESHUTDWN signal, then VCCON and PWRCS will go low if the ENFORCESHUTDWN bit is set. If the ENFORCESHUTDWN is not set, then the FORCESHUTDWN signal will have no affect.

FORCESHUTDWN:
If the ENFORCESHUTDWN bit is set, then this bit must be cleared after the initial boot, but prior to the memory interface waking up. This can only be guaranteed if the processor executes out of cache until this bit is cleared. This is not necessary for first time Power On Reset, since the cache has not been initialized.

FORCESHUTDWNOCC:
This bit is set if a force shut down occurs due to the PWROK signal going low prior to the processor clearing the FORCESHUTDWN bit. This bit provides status to the processor to indicate that the event has occurred. The bit should be cleared once the status has been checked, so it can be used again for the next power up.

SELC2MS:
The assertion of PWRCS will cause the System Power Supply to assert the VCC3 signal. Once VCC3 is asserted, the Oscillator and PLL inside of the PR31700 will wake up and generate the main system clock, but for a certain period of time the value of the clock will be unstable while the Oscillator and PLL lock to the correct frequency. The Power Module will prevent the clock from driving any logic until the clock is stable. The reference for the debounce delay time is provided by the C2MS (2 ms clock) and C8MS (8 ms clock) signals generated by the RTC from the 32 KHz Oscillator. If SELC2MS is set to a logic "1", the clock will be deasserted for a minimum of 4 ms and a maximum of 6 ms after VCC3 is turned on. If SELC2MS is set to a logic "0", the clock will be deasserted for a minimum of 16 ms and a maximum of 24 ms after VCC3 is turned on.

BPDBVCC3:
If this bit is set, the clock will be asserted immediately following the assertion of VCC3, instead of waiting for the delay mentioned in the SELC2MS bit description.

STOPCPU:
Setting this bit will cause the clock to the CPU core to be disabled for low power operation. The bit is cleared whenever an enabled interrupt is set. This bit should not be set when powering down the system. It is only used when the device is powered up, but the processor is not in use.

DBNCONBUTN:
If this bit is set, then the POSONBUTNINT and NEGONBUTNINT interrupts and the ONBUTN wake up function will only occur after a minimum of 16 ms and a maximum of 24 ms of debounce of the ONBUTN signal has taken place. If this bit is low, then the interrupts and power up will occur immediately upon transition of the ONBUTN signal.

COLDSTART:
This bit is set by RESET and provides status to the processor that a Power On Reset has occurred.

PWRCS:
This signal is set whenever the ONBUTN is pressed or an enabled interrupt occurs if the PWROK signal is high. The software clears this signal to latch commands into the System Power Supply. When powering down the system, this bit must be cleared simultaneously with the VCCON bit. All enabled interrupts must be cleared prior to powering down the system, or the PWRCS and VCCON signals will not go low.

VCCON:
This signal is asserted whenever the ONBUTN is pressed or an enabled interrupt occurs if the PWROK signal is high. When powering down the system, this bit must be cleared simultaneously with the PWRCS bit. All enabled interrupts must be cleared prior to powering down the system, or the PWRCS and VCCON signals will not go low.

Chapter 12 SIB Module

This section describes the Serial Interconnect Bus (SIB) Module, which contains logic for interfacing to the UCB1200 analog IC, as well as other optional external sound and/or telecom codec devices.

12.1 Overview

The SIB Module within the PR31700 contains holding registers, shift registers, and other logic to support interfacing to the UCB1200 analog IC and/or other optional external codec devices.

The overall sound subsystem allows playing and recording of sounds, and consists of a single-channel 12-bit codec within the UCB1200, the UCB1200 interface circuits for direct connection to an external microphone (MIC) and speaker, and DMA support within the PR31700.

Similarly, the overall telecom subsystem allows support of high-performance modems (up to V.34bis data rates or possibly even faster in future versions), and consists of a single-channel 14-bit codec within UCB1200, which also includes optional echo cancellation and interface circuits for connection to an external Data Access Arrangement (DAA), as well as an auxiliary input/output port for supporting future wireless interfaces. The DAA provides the front-end interface circuitry needed between the analog telephone network and the telecom codec for wireline configurations. The front-end circuit can also include ring detect, off-hook detect, and connect detect functions.

Audio and telecom data and control/status information (such as sampling rate, gain control, muting, clip detect, etc.) is passed between the PR31700 and the UCB1200 (and/or other external codec devices) via the SIB. Within the PR31700, the SIB Module logic provides DMA support and control/status registers for data transfers between external system memory, the CPU core, and the SIB. SIB data transfer is always synchronous and is frame-based, with the PR31700 side always the master source of the clock and frame frequency and phase.

For the sound subsystem, the UCB1200 allows the system to digitize (using the sound codec's ADC) the MIC input, as well as to convert (using the sound codec's DAC) sampled sounds (either pre-stored, synthesized, or previously recorded) stored in system memory to an analog audio output, for routing to either a speaker and/or headphone output. For the telecom subsystem, the UCB1200 allows the system to digitize (using the telecom codec's ADC) the telecom analog baseband receive input, and the subsequent modem baseband signal processing can be implemented using entirely a software-based approach. The UCB1200 then allows the system to convert (using the telecom codec's DAC) the processed baseband transmit output to an analog telecom output for routing back out the wireline or wireless port.

The SIB Module provides independent DMA support for sound receive and transmit, as well as independent DMA support for telecom receive and transmit (four total independent DMA channels). The DMA buffers can be configured in a continuous (circular) buffer mode or a one-time (empty or fill, then stop) buffer mode. Half-buffer and end-of-buffer DMA address counter interrupts are available, allowing the CPU to minimize overhead and efficiently empty or fill half of the DMA buffer in a ping-pong fashion. The DMA buffer size is programmable (up to a maximum of 16 KBytes) and the receive and transmit buffers can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation). Also available is a direct CPU read/write mode for bypassing the DMA, allowing the CPU to read or write the sound or telecom data on a sample by sample basis, if so desired.

12.1.1 Related Pins

SIBDIN: INPUT

This pin contains the input data shifted from the UCB1200 and/or external codec device.

SIBDOUT: OUTPUT

This pin contains the output data shifted to the UCB1200 and/or external codec device.

SIBSCLK: OUTPUT

This pin is the serial clock sent to the UCB1200 and/or external codec device. The programmable SIBSCLK rate is derived by dividing down from SIBMCLK.

SIBSYNC: OUTPUT
This pin is the frame synchronization signal sent to the UCB1200 and/or external codec device. This frame sync is asserted for one clock cycle immediately before each frame starts and all devices connected to the SIB monitor SIBSYNC to determine when they should transmit or receive data.

SIBIRQ: INPUT
This pin is a general purpose input port used for the SIB interrupt source from the UCB1200. This interrupt source can be configured to generate an interrupt on either a positive and/or negative edge.

SIBMCLK: INPUT/OUTPUT
This pin is the master clock source for the SIB logic. This pin is available for use in one of two modes. First, SIBMCLK can be configured as a high-rate output master clock source required by certain external codec devices. In this mode all SIB clocks are synchronously slaved to the main PR31700 system clock CLK2X. Conversely, SIBMCLK can be configured as an input slave clock source. In this mode, all SIB clocks are derived from an external SIBMCLK oscillator source, which is asynchronous with respect to CLK2X. Also, for this mode, SIBMCLK can still be optionally used as a high-rate master clock source required by certain external codec devices.

12.2 Interface Requirements

12.2.1 Frame Structure

Each SIB frame consists of 128 clock cycles, further divided into 2 subframes or words of 64 bits each. One subframe or word is allocated for each of up to 2 devices residing on the SIB, with the UCB1200 hard-coded to be the first subframe or word. The second subframe or word can support optional external audio or telecom codec devices. The SIB frame structure is shown in Figure 12-1. Currently, the SIB subframe 0 is dedicated to the UCB1200 and subframe 1 is not used .

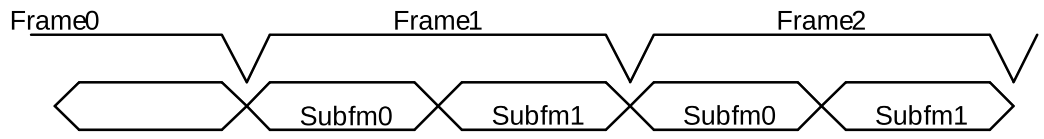


Figure 12-1. SIB Framing Structure

12.2.2 Timing Requirements

The PR31700 always samples receive data (SIBDIN) on the falling edge of SIBSCLK, whereas this data transmitted from the UCB1200 is always pushed on the rising edge of SIBSCLK. Similarly, the PR31700 always pushes transmit data (SIBDOUT) on the rising edge of SIBSCLK, whereas this data received by the UCB1200 is always sampled on the falling edge of SIBSCLK. The SIBSYNC frame sync signal is asserted for one clock cycle immediately before each frame starts and all devices connected to the SIB monitor SIBSYNC to determine when they should transmit or receive data. SIBSYNC is always sampled by the UCB1200 on the falling edge of SIBSCLK. The timing relationships for the SIB clock, sync, and data signals are shown in Figure 12-2.

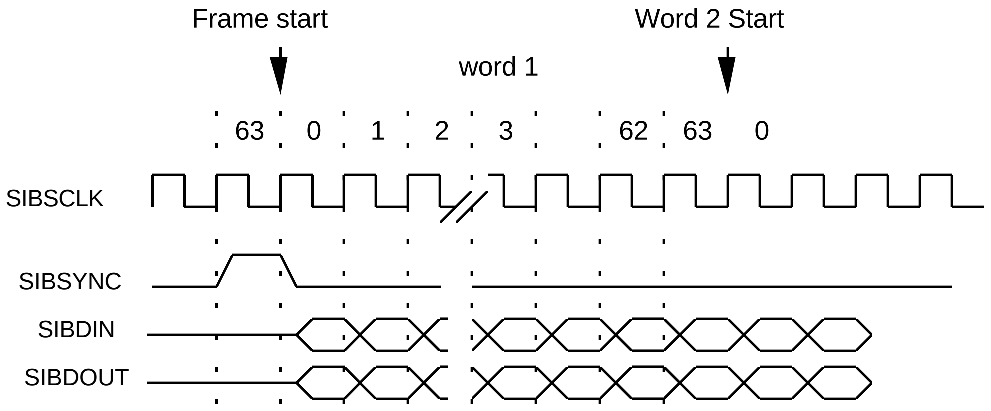


Figure 12-2. SIB Timing Relationships

12.2.3 Configurations

The SIB Module control bits SELSNDSF1 and SELTELSF1 are used to determine the desired SIB configuration of two subframes. Currently, only one possible configuration is supported, the UCB1200 assigned to subframe 0 of the SIB, while subframe 1 is not used. For this configuration, the system utilizes the audio and telecom codecs within the UCB1200. The SIBSCLK rate is fixed (at nominally CLK2X 8) and the independent sound and telecom sample rates are controlled via the PR31700 and the UCB1200 programmable counters. Other configuration will be supported in the future.

(Note: numbers in square brackets refer to notes/comments at bottom of table)

Table 12-1. SIB Configuration

config	sound codec	teleco codec	PR31700 clk2x [1]	sibsclk	sibclk out or sound OSC in [2]	SIB frame rate [3]	comments

1	UCB1200 ; Fs = 19.2K, 22.154K, or 24K	UCB1200 ; Fs = 7.2K, 8K, or 9.6K	73.728M	9.216M (8)	18.432M out (4)	72K	UCB1200 only; lowest cost;
							sibsclk = fixed; sound & telecom Fs control via UCB1200 prgm [4];
							SIB subframe0 = UCB1200;
							SIB subframe1 = not used;
							SELSNDSF1=0;
							SELTELSF1=0;

2							TBD
3							TBD

NOTES:

[1] The PR31700 clk2x is internal high-rate system-wide clock (which is 2x the CPU clock rate, clk)

[2] optional external sound OSC can be connected to the PR31700 sibmclk (to allow decoupling of SIB clocks from clk2x; else this pin can be configured as buffered sibmclk output which is synchronously divided down from clk2x (sibmclk allows potential support of codecs requiring a high-rate fixed master clock + a sample-rate-dependent serial clock); sibsclk serial clock is always synchronously divided down from sibmclk

[3] SIB frame rate = sibsclk 128;ratio for average number of SIB-frames per valid-data-frame is dependent upon sound & telecom Fs

[4] UCB1200 audio/telecom Fs divider programmability: SIB-frame/valid-data-frame RATIO

(sibsclk x 2)	(40 x 64) = 7.2K		5
(sibsclk x 2)	(36 x 64) = 8K		4.5
(sibsclk x 2)	(30 x 64) = 9.6K		3.75
(sibsclk x 2)	(15 x 64) = 19.2K	(1.6% error vs. 18.9K for CD-XA)	1.875
(sibsclk x 2)	(13 x 64) = 22.154K	(0.47% error vs. 22.050K for CD-Audio)	1.625
(sibsclk x 2)	(12 x 64) = 24K		1.5

12.2.4 Sample Rates

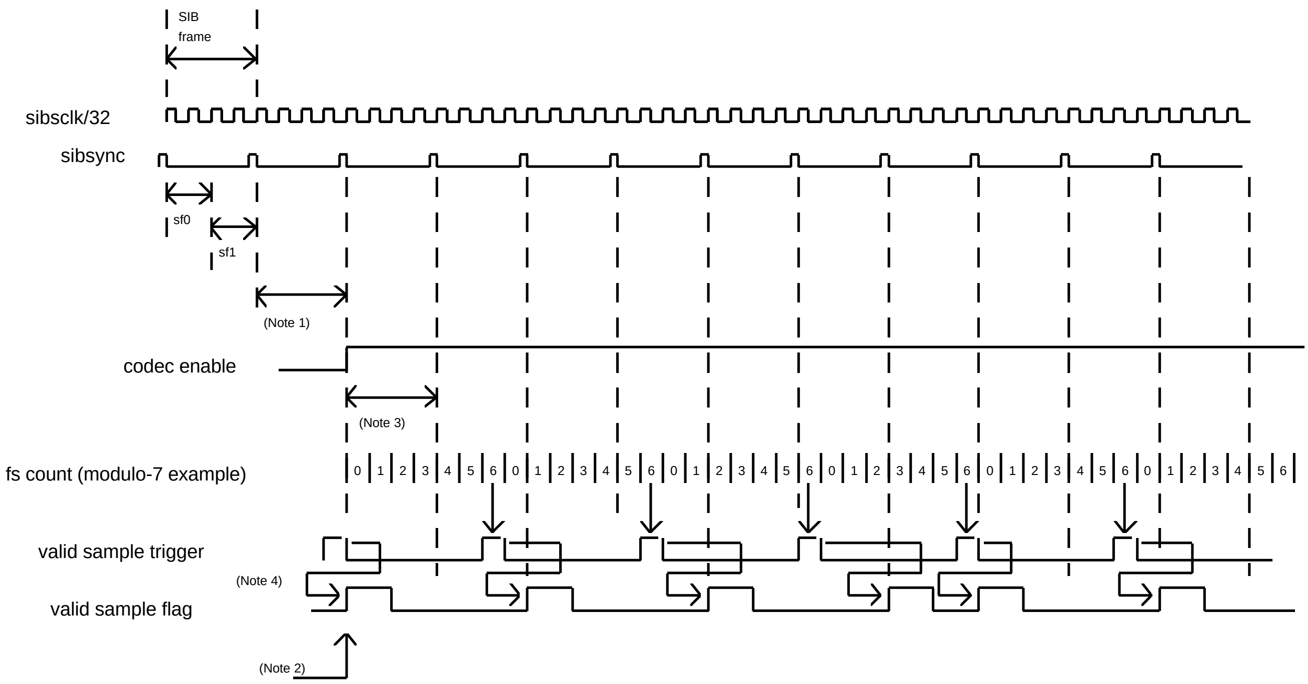
The SNDFSDIV and TELFSDIV control bits are used to independently configure the sampling rate for the audio and telecom codecs, respectively. These control bits set the modulus of the audio and telecom sample rate counters within the SIB Module. The UCB1200 also contains its own set of audio and telecom sample rate counters for its internal use and these counters must be both frequency and phase synchronized to the sample rate counters within the PR31700 in order to guarantee correct operation. The synchronization of these counters is triggered from the codec enable command transmitted by the PR31700 and received by the UCB1200, as shown in Figure 12-3.

The sample rate is calculated using the following (note that the FSDIV value loaded into the counter is the desired count modulus -1):

Fs = (SIBSCLK x 2)   ((FSDIV+1) x 64)

For example, if SIBSCLK = 9.216 MHz and FSDIV = 39 (module 40), then:

Fs = (9.216 MHz x 2)   ((39+1) x 64)
   = 7.2 KHz



Note 1: during this frame, audio (and/or telecom) codec input or output enable command is transmitted by the PR31700 and received by UCB1200

Note 2: at start of first frame after codec is enabled, audio (and/or telecom) counters are reset; this ensures that the PR31700 and UCB1200 fs counters are immediately phase-synchronized

Note 3: during this frame, first valid DAC sample is transmitted by DINO (received by the UCB1200) or first valid ADC sample is transmitted by UCB1200 (received by DINO); valid DAC and ADC samples are always transmitted during the same subframe (full duplex)

Note 4: audio (and/or telecom) vflag bits are asserted to UCB1200 for one subframe each time a new DAC sample is being transmitted (the PR31700 will also latch a new ADC sample from the UCB1200 during the same subframe); since the audio (and/or telecom) programmable sample rate is derived from sibsclk/32 (= 4x frame rate), valid samples are synchronized from the periodic fs count rollover to the next available SIB subframe

Above timing diagram shows fs count and valid sample flag generation for example configuration where frequency divisor = 6 (modulo-7 divide).

Figure 12-3. SIB Sample Rate Setup Timing

12.2.5 Enable/Disable Sequencing

As mentioned in the previous section, the synchronization of the sample rate counters within the PR31700 and the UCB1200 is triggered from the codec enable command transmitted by the PR31700 and received by the UCB1200. The following steps illustrate the full sequence of events needed to enable and disable the audio and/or telecom processing paths, while ensuring synchronization of these sample rate counters (the example below shows a configuration with the UCB1200 for sound and telecom, using subframe 0 only):

INITIAL ENABLE SEQUENCE:

1. SIB is initially disabled (ENSIB = 0)
2. setup SIB control registers to desired setttings (SIB clock rates, sample rate dividers, data formats (8-bit vs. 16-bit, etc.), DMA settings, subframe 0 enable = ENSF0
3. write initial data and control values to transmit holding registers (SNDTXHOLD, TELTXHOLD, SF0AUX); SF0AUX control register sets up the UCB1200 audio and telecom codecs, including sample rate dividers (to match PR31700 sample rates), gains, etc. -- codecs are not enabled at this point
4. enable SIB (assert ENSIB = 1)
5. SIB then begins transmitting and receiving data; the UCB1200 control registers get configured with desired settings; codecs still not enabled at this point SND (or TEL) ENABLE SEQUENCE:
6. setup software to trigger from SIBSF0INT
7. immediately after SIBSF0INT trigger, software must write to SF0AUX to assert Audio (or Telecom) Codec Enable, and during same subframe (i.e., before SIBSF1INT event occurs), software must assert ENSND (or ENTEL)
8. after next SIBSF0INT and during the subframe 0 period, the Codec Enable Command is transmitted from the PR31700 and received by the UCB1200
9. the Audio (or Telecom) Codec within the UCB1200 is then enabled at the start of the next frame after the Enable Command is sent, at which time the sample rate counters within both the PR31700 and the UCB1200 begin counting in a synchronized fashion
10. during this first frame after codec is enabled, the first DAC sample is transmitted by the PR31700 (received by the UCB1200) and/or the first ADC sample is transmitted by the UCB1200(received by the PR31700); DAC and ADC samples are always transmitted during the same subframe in a full-duplex manner (as mentioned in the previous section, DAC and ADC samples are synchronized from the periodic sample rate counter rollover to the next available SIB subframe)
11. check valid status from the UCB1200 (indicates whether ADC's are settled, all turn-on delays have been met, etc.) which indicates valid samples SND (or TEL) DISABLE SEQUENCE:
12. software writes to SF0AUX to de-assert Audio (or Telecom) Codec Enable, in order to disable codec
13. software then de-asserts ENSND (or ENTEL) to disable sound (or telecom) processing on the PR31700 side

12.2.6 Data formats

The PR31700's SIB Module logic contains transmit and receive holding registers which are used by the DMA circuit or CPU to write and read sound and telecom data and SIB control and status register data. The SNDTXHOLD and SNDRXHOLD registers are used for transmit and receive sound data corresponding to the appropriate subframe (0 or 1), according to the configuration as determined by the SELSNDSF1 control bit.

Similarly, the TELTXHOLD and TELRXHOLD registers are used for transmit and receive telecom data corresponding to the appropriate subframe (0 or 1), according to the configuration as determined by the SELTELSF1 control bit. All of these holding registers are 32 bits wide, and the appropriate shift register fields for each subframe are written to or read from based on the programmed sound and telecom samples rates, mono versus stereo mode, and 8-bit versus 16-bit modes.

The SF0AUX and SF1AUX control holding registers and SF0STAT and SF1STAT status holding registers are also 32 bits wide. Each of these control holding registers updates the control fields for the respective subframe once per frame. These transmit holding registers load the same control data into the shift register until the CPU updates the contents of the holding register. Similarly, the status holding registers are updated from the received status data in the shift register once per frame. The contents of the receive holding registers are available to be read by the CPU at any time.

The SIB data format for transfers to and from the UCB1200 is shown in Figure 12-4. Each word consists of fields containing audio data, telecom data, and control register address and data. During each frame, data is transferred bi-directionally (using SIBDIN and SIBDOUT) for all of these fields. For instance, the audio data field is written once every frame (from the PR31700 to the UCB1200) via SIBDOUT and also returns audio data (from the UCB1200 to the PR31700) at the same time via SIBDIN.

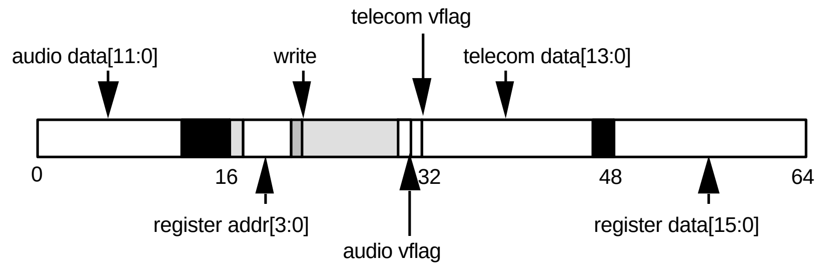


Figure 12-4. Data Format for UCB1200 Word Interface

The SF0AUX control holding register and SF0STAT status holding register map directly to bits 16-31 and 48-63 of the subframe 0 word format. See Table 12-2 which shows the mapping of SF0AUX and SF0STAT bit positions to the respective the UCB1200 fields. For the UCB1200, the audio and telecom "vflag" bits (valid flags) in the transmit direction are dynamically asserted to the UCB1200 for one subframe each time a new sample is being transmitted. The audio and telecom "vflag" bits in the receive direction are statically asserted from the UCB1200 to the PR31700 only after the ADC's are settled, all turn-on delays have been met, etc. The PR31700 then uses these valid flags to determine when the receive data from the UCB1200 is truly valid.

Table 12-2. SIB Subframe 0 Control/Status Bits and Fields

SF0AUX/SF0STAT bits	UCB1200 word bits	field definition
---	0-15	audio data
31	16	reserved for register extension; must write to 0 if not used
30-27	17-20	register address[3:0]
26	21	write bit
25-18	22-29	reserved, must write to 0
17	30	audio valid flag
16	31	telecom valid flag
---	32-47	telecom data
15-0	48-63	register data[15:0]

The last 16 bits of the UCB1200 word is made up of control register data. The exact contents and definition of this field is defined by the register address field and the "write" bit. For a read cycle ("write" bit = 0), the address bits determine which the UCB1200 register to read and this read data is sent by UCB1200 within the control register data field of SIBDIN during the same frame as the read request occurred. The separation between the address and data fields should make this possible. In addition, during a read cycle, the control register data field of SIBDOUT is ignored by the UCB1200. For a write cycle ("write" bit = 1), the control register data contents of SIBDOUT are written to the UCB1200 register pointed to by the register address field.

12.3 Implementation

12.3.1 Block Diagram

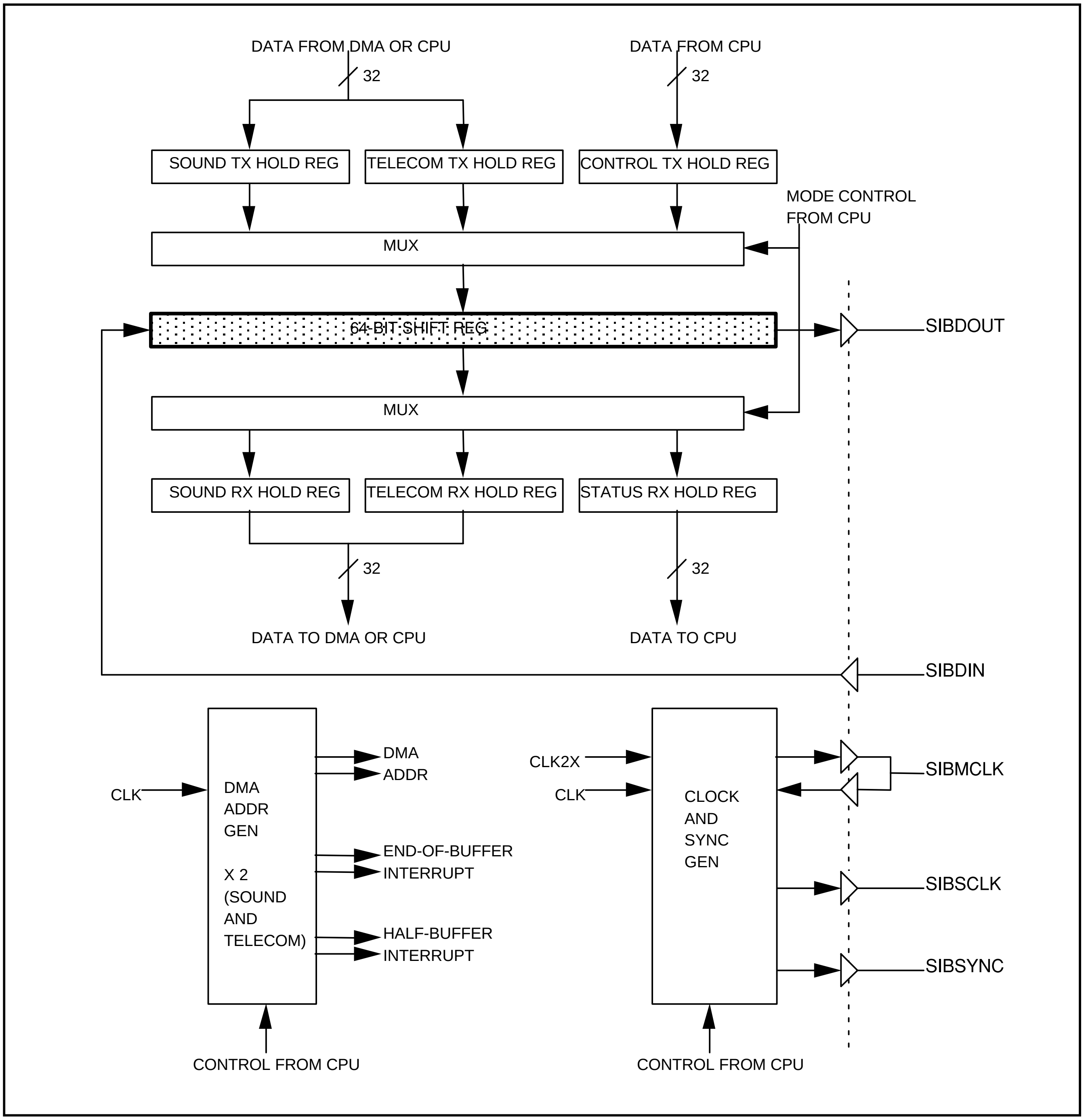


Figure 12-5. SIB Module Block Diagram

12.3.2 Holding and Shift Registers

The SIB Module logic consists of holding registers (both transmit and receive) and a serial shift register to support the transfer of sound and telecom data and control/status information between the PR31700 and the UCB1200 (and/or other external codec devices) via the SIB (see Figure 12-5 for a block diagram of the SIB Module). For the SIB transmit direction, the sound and telecom transmit holding registers are written either from the DMA circuit or directly from the CPU, while the subframe 0 and subframe 1 control holding registers are always written directly from the CPU. These holding registers are processed by an array of multiplexers which select which fields of the various 32-bit transmit holding registers are loaded into the 64-bit shift register, such that an entire subframe is loaded in parallel and then shifted out the serial output SIBDOUT. The sequence of loading the shift register from the various holding register fields is determined by the programmed sound and telecom sample rates, mono versus stereo mode, and 8-bit versus 16-bit modes. Conversely, for the SIB receive direction, the sound and telecom receive holding registers are read either by the DMA circuit or directly by the CPU, while the subframe 0 and subframe 1 status holding registers are always read directly by the CPU. The 64-bit shift register containing an entire subframe of received data from the serial input SIBDIN is processed by an array of multiplexers which select which fields of the various 32-bit receive holding registers are loaded from the shift register. The sequence of loading the various fields of the receive holding registers from the shift register is determined by the programmed sound and telecom sample rates, mono versus stereo mode, left versus right mono source, and 8-bit versus 16-bit modes.

12.3.3 Subframe Formats

SIB subframe 0 is nominally reserved for interfacing to the UCB1200. Four possible sample-size modes are supported for sound and telecom data (these are all mono sound and telecom formats):

sound	**telecom**
16-bit	16-bit
16-bit	8-bit
8-bit	16-bit
8-bit	8-bit

See Table 12-3 for a summery matrix of the subframe 0 and 1 holding register and shift register formats. The subframe 1 holding register and shift register is for future reference, not currently used.

Table 12-3a. SIB Holding and Shift Register Formats (Subframe 0)

Subframe 0 Sound 16 Bit, Telecom 16 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:16]	sf0aux[31:18],VS,VT	teltxhold[31:16]	sf0aux[15:0]
1	sndtxhold[15:0]	sf0aux[31:18],VS,VT	teltxhold[15:0]	sf0aux[15:0]
2	sndtxhold[31:16]	sf0aux[31:18],VS,VT	teltxhold[31:16]	sf0aux[15:0]
3	sndtxhold[15:0]	sf0aux[31:18],VS,VT	teltxhold[15:0]	sf0aux[15:0]

Subframe 0 Sound 16 Bit, Telecom 8 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:16]	sf0aux[31:18],VS,VT	teltxhold[31:24],8b0	sf0aux[15:0]
1	sndtxhold[15:0]	sf0aux[31:18],VS,VT	teltxhold[23:16],8b0	sf0aux[15:0]
2	sndtxhold[31:16]	sf0aux[31:18],VS,VT	teltxhold[15:8],8b0	sf0aux[15:0]
3	sndtxhold[15:0]	sf0aux[31:18],VS,VT	teltxhold[7:0],8b0	sf0aux[15:0]

Subframe 0 Sound 8 Bit, Telecom 16 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:24],8b0	sf0aux[31:18],VS,VT	teltxhold[31:16]	sf0aux[15:0]
1	sndtxhold[23:16],8b0	sf0aux[31:18],VS,VT	teltxhold[15:0]	sf0aux[15:0]
2	sndtxhold[15:8],8b0	sf0aux[31:18],VS,VT	teltxhold[31:16]	sf0aux[15:0]
3	sndtxhold[7:0],8b0	sf0aux[31:18],VS,VT	teltxhold[15:0]	sf0aux[15:0]

Subframe 0 Sound 8 Bit, Telecom 8 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:24],8b0	sf0aux[31:18],VS,VT	teltxhold[31:24],8b0	sf0aux[15:0]
1	sndtxhold[23:16],8b0	sf0aux[31:18],VS,VT	teltxhold[23:16],8b0	sf0aux[15:0]
2	sndtxhold[15:8],8b0	sf0aux[31:18],VS,VT	teltxhold[15:8],8b0	sf0aux[15:0]
3	sndtxhold[7:0],8b0	sf0aux[31:18],VS,VT	teltxhold[7:0],8b0	sf0aux[15:0]

NOTE: "VS" = valid sound flag bit; "VT" = valid telecom flag bit
"8b0" = 8-bit field of zero values
"SR" = Shift Register

Table 12-3b. SIB Holding and Shift Register Formats (Subframe 1 Sound)

Subframe 1 Sound 16 Bit Stereo:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]
1	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]
2	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]
3	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]

Subframe 1 Sound 16 Bit Mono:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[31:16]	sf1aux[15:0]
1	sndtxhold[15:0]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]
2	sndtxhold[31:16]	sf1aux[31:16]	sndtxhold[31:16]	sf1aux[15:0]
3	sndtxhold[15:0]	sf1aux[31:16]	sndtxhold[15:0]	sf1aux[15:0]

Subframe 1 Sound 8 Bit Stereo:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:24],8b0	sf1aux[31:16]	sndtxhold[23:16],8b0	sf1aux[15:0]
1	sndtxhold[15:8],8b0	sf1aux[31:16]	sndtxhold[7:0],8b0	sf1aux[15:0]
2	sndtxhold[31:24],8b0	sf1aux[31:16]	sndtxhold[23:16],8b0	sf1aux[15:0]
3	sndtxhold[15:8],8b0	sf1aux[31:16]	sndtxhold[7:0],8b0	sf1aux[15:0]

Subframe 1 Sound 8 Bit Mono:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	sndtxhold[31:24],8b0	sf1aux[31:16]	sndtxhold[31:24],8b0	sf1aux[15:0]
1	sndtxhold[23:16],8b0	sf1aux[31:16]	sndtxhold[23:16],8b0	sf1aux[15:0]
2	sndtxhold[15:8],8b0	sf1aux[31:16]	sndtxhold[15:8],8b0	sf1aux[15:0]
3	sndtxhold[7:0],8b0	sf1aux[31:16]	sndtxhold[7:0],8b0	sf1aux[15:0]

NOTE: "8b0" = 8-bit field of zero values
"SR" = Shift Register

Table 12-3c. SIB Holding and Shift Register Formats (Subframe 1 Telecom)

Subframe 1 Telecom 16 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	16b0	sf1aux[31:17],VT	teltxhold[31:16]	sf1aux[15:0]
1	16b0	sf1aux[31:17],VT	teltxhold[15:8]	sf1aux[15:0]
2	16b0	sf1aux[31:17],VT	teltxhold[31:16]	sf1aux[15:0]
3	16b0	sf1aux[31:17],VT	teltxhold[15:8]	sf1aux[15:0]

Subframe 1 Telecom 16 Bit:

Frame	SR[64:49]	SR[48:33]	SR[32:17]	SR[16:1]
0	16b0	sf1aux[31:17],VT	teltxhold[31:24],8b0	sf1aux[15:0]
1	16b0	sf1aux[31:17],VT	teltxhold[23:16],8b0	sf1aux[15:0]
2	16b0	sf1aux[31:17],VT	teltxhold[15:8],8b0	sf1aux[15:0]
3	16b0	sf1aux[31:17],VT	teltxhold[7:0],8b0	sf1aux[15:0]

NOTE: "VS" = valid sound flag bit; "VT" = valid telecom flag bit
"8b0" = 8-bit field of zero values; "16b0" = 16-bit field of zero values
"SR" = Shift Register

12.3.4 Clock and Sync Generation

The SIB Module logic contains several programmable counters which are used to generate the various SIB internal and external control signals and clocks. See Figure 12-7 for a block diagram of the SIB clock and sync generation circuit. As mentioned previously, SIBMCLK can be configured as either an input or output. As an output, SIBMCLK is derived by dividing down from CLK2X. In this mode, all SIB clocks are then synchronously slaved to the main the PR31700 system clock and only one main XTAL source is required for the entire system (besides the 32.768 KHz RTC XTAL), resulting in the lowest cost system configuration with regards to clock sources. As an input, SIBMCLK is generated from an external oscillator source, which is asynchronous with respect to CLK2X. This more expensive configuration allows the SIB (and also potentially the CHI and dual-UART circuits) to operate independent of the frequency used for the CPU core. This allows flexible system design options with respect to the CPU operating frequency, along with the ability to generate the required or desired UART baud rates and audio/telecom sampling frequencies.

The programmable SIBSCLK rate is derived by dividing down from SIBMCLK. Nominal rates for a UCB1200-only configuration are SIBMCLK = 18.432 MHz (equals CLK2X   4) and SIBSCLK = 9.216 MHz (equals SIBMCLK   2). The SIBSCLK rate is then used to enable a set of counters (word counter and subframe/frame counter) to generate the SIBSYNC frame rate of 72 KHz (equals SIBSCLK   128), as well as various internal SIB clocks and control signals.

The subframe rate is used to enable independent sets of programmable sound and telecom sample rate counters which generate various control signals based on the desired sample rates. For instance, this logic correctly generates the sound and telecom valid flag bits which are asserted within the SIBDOUT serial output stream every time a valid sample is being transmitted.

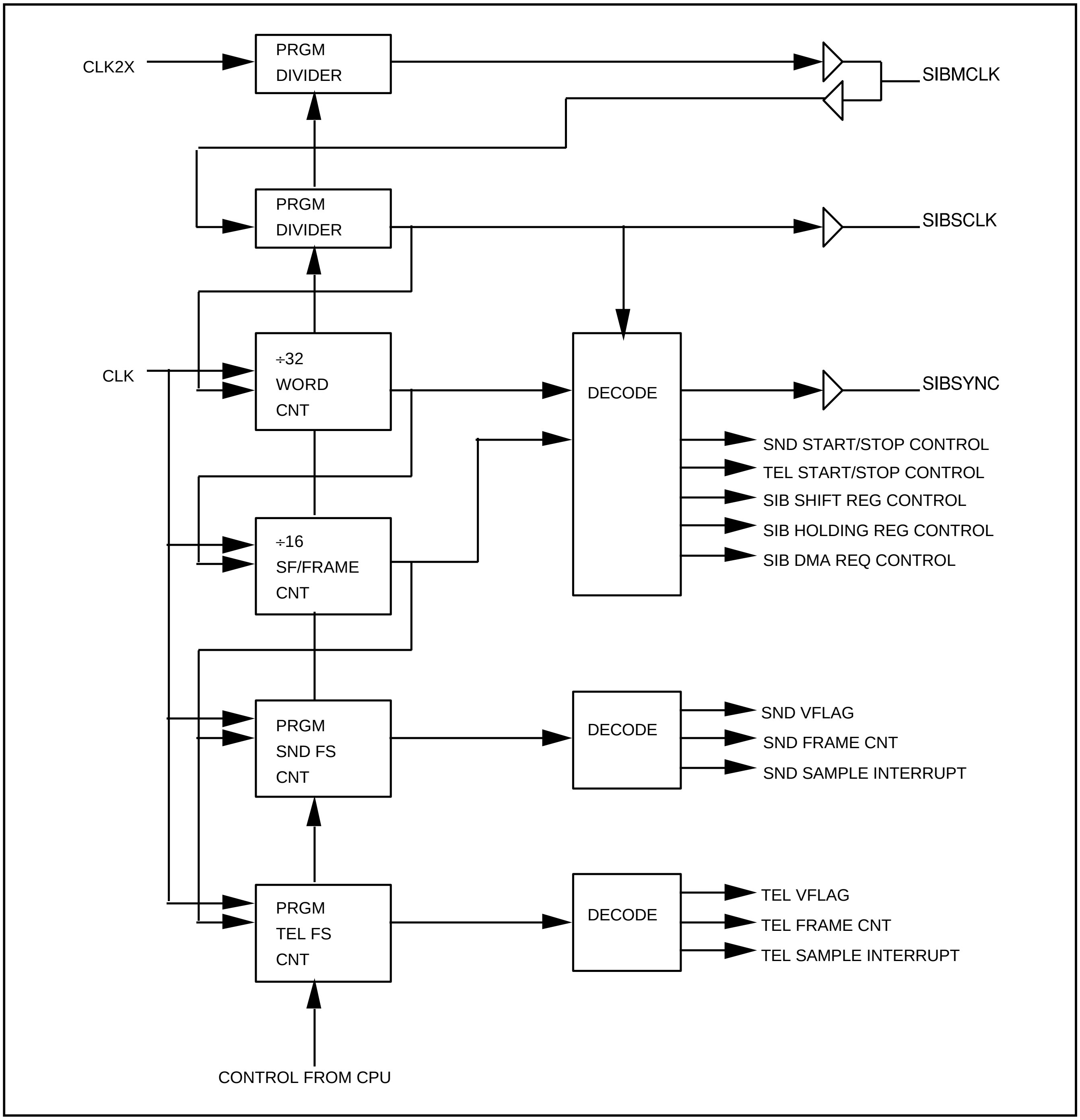


Figure 12-6. SIB Clock and Sync Generation

12.3.5 DMA Address Generation

The SIB Module provides support for four independent DMA channels: sound receive and transmit and telecom receive and transmit. Two identical circuits (one for sound DMA and one for telecom DMA) are used to generate the DMA address, as well as half-buffer and end-of-buffer interrupts (see Figure 12-7).

The DMA buffer size is programmable (up to a maximum of 16 KBytes) and the receive and transmit buffer start addresses are also programmable (anywhere over the full 32-bit address space). Because there are separate start addresses, the receive and transmit buffers can be configured to either reside in different memory spaces or share the same memory space. The latter setup allows for overlapping buffers for loopback purposes or for optimum memory allocation, for which the DMA logic supports two full-duplex loopback modes. For one mode, receive DMA requests are issued first, followed by transmit DMA requests. This ordering allows a receive-to-transmit immediate loopback via the DMA buffer. For the second mode, transmit DMA requests are issued first, followed by receive DMA requests. Thus, received samples are written to the DMA buffer location immediately after transmit samples were read from that same location (which then became immediately available). This ordering allows a single circular DMA buffer to be used for both transmit and receive samples.

The DMA buffers can be configured in a circular buffer mode or a one-time buffer mode. For the circular mode, the DMA address is continuously incremented (each time a DMA acknowledge is received from the PR31700's central DMA controller) and rolls over back to the start address after the end-of-buffer is reached and will continue operating in a continuous and circular manner. For the one-time mode, the DMA logic will stop executing whenever the end-of-buffer is reached.

Half-buffer and end-of-buffer DMA address counter interrupts are available, allowing the CPU to minimize overhead and utilize the DMA buffer in a ping-pong fashion. For transmit mode, the CPU can use these interrupts to fill or write one half of the buffer while the other half is being emptied by the DMA controller for transmitting out the SIB. Similarly, for receive mode, the CPU can use these interrupts to empty or read one half of the buffer while the other half is being filled by the DMA controller from received SIB input samples.

Also available is a direct CPU read/write mode for bypassing the DMA, allowing the CPU to read or write the sound or telecom data on a sample by sample basis, if so desired. Separate DMA enables for receive and transmit allow DMA to be setup for receive only (transmit via CPU), transmit only (receive via CPU), receive and transmit, or none (receive and transmit via CPU).

The sound and telecom DMA circuits also provide an interrupt each time the respective DMA buffer pointer is incremented, which occurs whenever a new sample is read from and/or written to the DMA buffer. This interrupt may be useful for triggering a read of the DMA pointer status value, which is the actual 12-bit DMA address counter output. This value indicates exactly where the current address is pointing to in the overall DMA buffer.

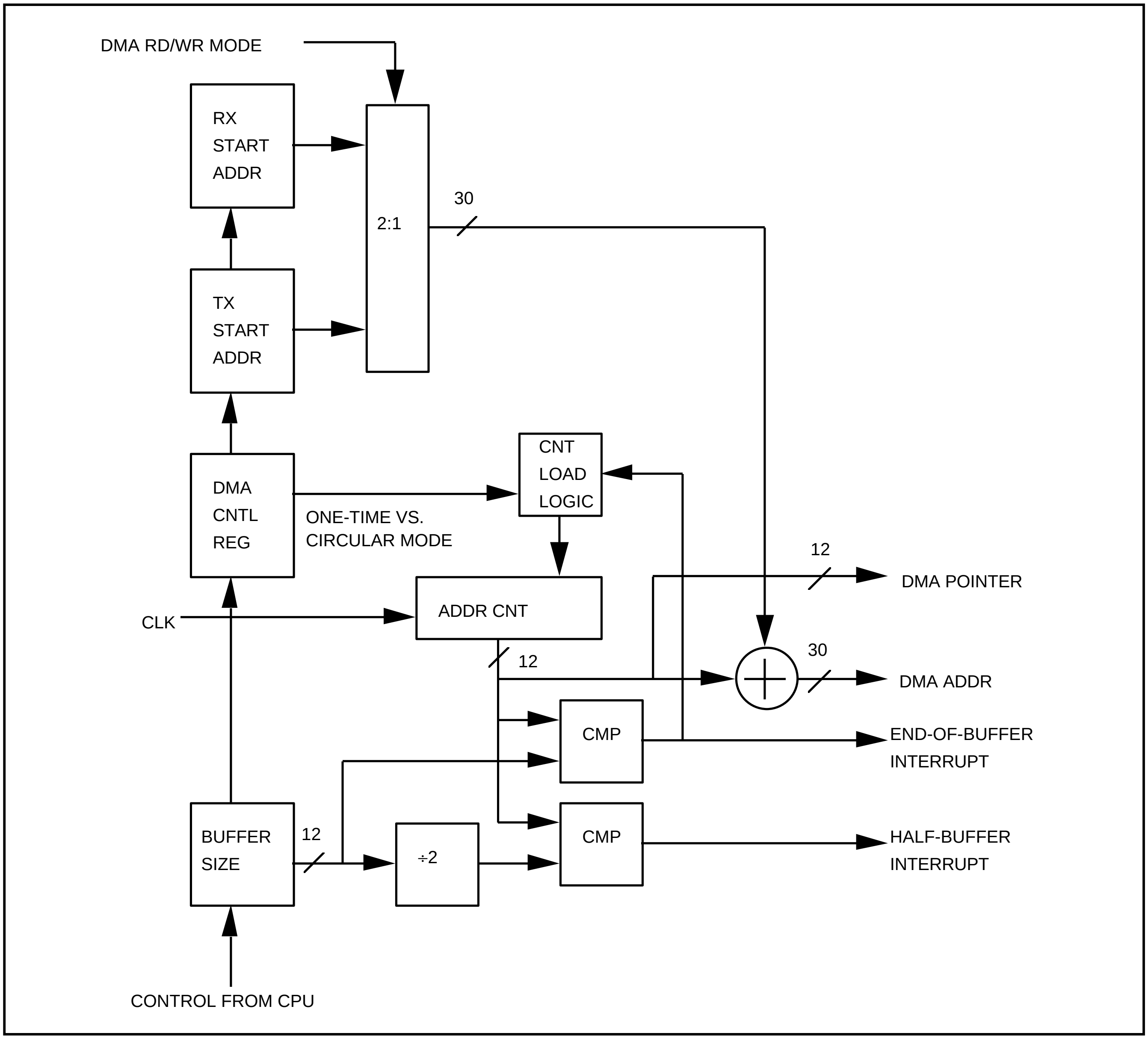


Figure 12-7. SIB DMA Address Generation

12.3.6 Related Interrupts

SND0_5INT:
Issues an interrupt whenever the sound DMA buffer pointer has reached the halfway point.

SND1_0INT:
Issues an interrupt whenever the sound DMA buffer pointer has reached the end-of-buffer point.

TEL0_5INT:
Issues an interrupt whenever the telecom DMA buffer pointer has reached the halfway point.

TEL1_0INT:
Issues an interrupt whenever the telecom DMA buffer pointer has reached the end-of-buffer point.

SNDDMACNTINT:
Issues an interrupt each time the sound DMA buffer pointer is incremented, which occurs whenever a new sound sample is read from and/or written to the sound DMA buffer.

TELDMACNTINT:
Issues an interrupt each time the telecom DMA buffer pointer is incremented, which occurs whenever a new telecom sample is read from and/or written to the telecom DMA buffer.

LSNDCLIPINT:
Issues an interrupt whenever the amplitude of the left channel sound data is clipping the codec A/D converter for SIB subframe 1.

RSNDCLIPINT:
Issues an interrupt whenever the amplitude of the right channel sound data is clipping the codec A/D converter for SIB subframe 1.

VALSNDPOSINT:
Issues an interrupt whenever the valid sound status flag transitions from a logic "0" to a logic "1". This valid flag is triggered from SIB subframe 0 (if SELSNDSF1 = "0") or from SIB subframe 1 (if SELSNDSF1 = "1").

VALSNDNEGINT:
Issues an interrupt whenever the valid sound status flag transitions from a logic "1" to a logic "0". This valid flag is triggered from SIB subframe 0 (if SELSNDSF1 = "0") or from SIB subframe 1 (if SELSNDSF1 = "1").

VALTELPOSINT:
Issues an interrupt whenever the valid telecom status flag transitions from a logic "0" to a logic "1". This valid flag is triggered from SIB subframe 0 (if SELTELSF1 = "0") or from SIB subframe 1 (if SELTELSF1 = "1").

VALTELNEGINT:
Issues an interrupt whenever the valid telecom status flag transitions from a logic "1" to a logic "0". This valid flag is triggered from SIB subframe 0 (if SELTELSF1 = "0") or from SIB subframe 1 (if SELTELSF1 = "1").

SNDININT:
Issues an interrupt whenever a valid sound input longword (32 bits) is available from the Sound RX Holding Register; this also means a valid sound output longword can be written to the Sound TX Holding Register.

TELININT:
Issues an interrupt whenever a valid telecom input longword (32 bits) is available from the Telecom RX Holding Register; this also means a valid telecom output longword can be written to the Telecom TX Holding Register.

SIBSF0INT:
Issues an interrupt at the start of every SIB subframe 0. This is used to initiate CPU reading of the SIB Subframe 1 Status Register (SF1STAT Register) and/or CPU writing of the SIB Subframe 0 Control Register (SF0AUX Register).

SIBSF1INT:
Issues an interrupt at the start of every SIB subframe 1. This is used to initiate CPU reading of the SIB Subframe 0 Status Register (SF0STAT Register) and/or CPU writing of the SIB Subframe 1 Control Register (SF1AUX Register).

SIBIRQPOSINT:
Issues an interrupt whenever the SIBIRQ pin transitions from a logic "0" to a logic "1".

SIBIRQNEGINT:
Issues an interrupt whenever the SIBIRQ pin transitions from a logic "1" to a logic "0".

12.4 SIB Registers

12.4.1 SIB Size Register

OFFSET = $060: **write-only**

Bit	Label	RESET	Read/Write
31-30	Reserved		
29-18	SNDSIZE(13:2)	X	W
17-14	Reserved		
13-2	TELSIZE(13:2)	X	W
1-0	Reserved		

SNDSIZE(31:2): **write-only**

These bits define the size of the sound DMA buffers (16 KBytes maximum). Both the sound RX buffer and the sound TX buffer are the same size. The last address in the sound RX DMA buffer is given by SNDRXSTART(31:2) + SNDSIZE(13:2). The last address in the sound TX DMA buffer is given by SNDTXSTART(31:2) + SNDSIZE(13:2). The value loaded into SNDSIZE should be equal to the desired buffer length - 1.

TELSIZE(31:2): **write-only**

These bits define the size of the telecom DMA buffers (16 KBytes maximum). Both the telecom RX buffer and the telecom TX buffer are the same size. The last address in the telecom RX DMA buffer is given by TELRXSTART(31:2) + TELSIZE(13:2). The last address in the telecom TX DMA buffer is given by TELTXSTART(31:2) + TELSIZE(13:2). The value loaded into TELSIZE should be equal to the desired buffer length - 1.

12.4.2 SIB Sound RX Start Register

OFFSET = $064: **write-only**

Bit		Label	RESET Read/Write
31-2	SNDRXSTART(31:2)	X	W
1-0	Reserved		

SNDRXSTART(31:2): **write-only**

These bits define the start address for the sound RX DMA buffer. The sound RX buffer and sound TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

12.4.3 SIB Sound TX Start Register

OFFSET = $068: **write-only**

Bit	Label	RESET	Read/Write
31-2	SNDTXSTART(31:2)	X	W
1-0	Reserved		

SNDTXSTART(31:2): **write-only**

These bits define the start address for the sound TX DMA buffer. The sound RX buffer and sound TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

12.4.4 SIB Telecom RX Start Register

OFFSET = $06C: **write-only**

Bit	Label	RESET	Read/Write
31-2	TELRXSTART(31:2)	X	W
1-0	Reserved		

TELRXSTART(31:2): **write-only**

These bits define the start address for the telecom RX DMA buffer. The telecom RX buffer and telecom TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

12.4.5 SIB Telecom TX Start Register

OFFSET = $070: **write-only**

Bit	Label	RESET	Read/Write
31-2	TELTXSTART(31:2)	X	W
1-0	Reserved		

TELTXSTART(31:2): **write-only**

These bits define the start address for the telecom TX DMA buffer. The telecom RX buffer and telecom TX buffer can be configured to either reside in different memory spaces or share the same memory space (overlapping buffers for loopback purposes or for optimum memory allocation).

12.4.6 SIB Control Register

OFFSET = $074:

Bit	Label	RESET	Read/Write
31	SIBIRQ	-	R
30	ENCNTTEST	0	R/W
29	ENDMATEST	0	R/W
28	SNDMONO	X	R/W
27	RMONOSNDIN	X	R/W
26-24	SIBSCLKDIV(2:0)	X	R/W
23	TEL16	X	R/W
22-16	TELFSDIV(6:0)	X	R/W
15	SND16	X	R/W
14-8	SNDFSDIV(6:0)	X	R/W
7	SELTELSF1	X	R/W
6	SELSNDSF1	X	R/W
5	ENTEL	0	R/W
4	ENSND	0	R/W
3	SIBLOOP	0	R/W
2	ENSF1	0	R/W
1	ENSF0	0	R/W
0	ENSIB	0	R/W

SIBIRQ: **read-only**

This bit provides the logic state of the SIBIRQ input pin. The SIBIRQ pin is active-high from the UCB1200.

ENCNTTEST:

This bit is used for IC testing and should not be set.

ENDMATEST:

This bit is used for IC testing and should not be set.

SNDMONO:

This bit is used to configure the sound input and output format for SIB subframe 1 as mono versus stereo. Setting this bit to a logic "1" selects mono mode. Clearing this bit to a logic "0" selects stereo mode.

RMONOSNDIN:

This bit is used to select left versus right channel as the mono sound source for SIB subframe 1, whenever the sound is configured as mono mode. Setting this bit to a logic "1" selects the right channel as the mono sound source. Clearing this bit to a logic "0" selects the left channel as the mono sound source.

SIBSCLKDIV(2:0):
These bits select the start count value and the stop count value for the 4-bit programmable counter used to generate SIBSCLK, which is derived by dividing down SIBMCLK; SIBSCLK is the serial bit clock used for the SIB. Since the MSB of the counter output is used for SIBSCLK, the start count and stop count values are chosen to provide (as close as possible) a 50% duty cycle SIBSCLK. The table used to compute these counter start and stop values are as follows:

SIBSCLKDIV	start value	stopvalue	divide-modulus
0	7	8	2
1	6	8	3
2	6	9	4
3	5	9	5
4	5	10	6
5	4	11	8
6	3	12	10
7	2	13	12

TEL16:
This bit is used to configure the SIB telecom input and output format as 8-bit versus 16-bit mode. Setting this bit to a logic "1" selects 16-bit mode. Clearing this bit to a logic "0" selects 8-bit mode.

TELFSDIV(6:0):
These bits select the divider modulus for the 7-bit programmable counter used to generate the telecom sample rate clock, which is derived by dividing down an internal SIB clock of rate equal to the 4 times the SIB frame rate or 2 times the SIB subframe rate (where each frame consists of 128 serial data bits). This programmable divider consists of a down-counter which counts down from TELFSDIV to zero, so the value loaded for TELFSDIV should be the desired divider modulus - 1.

For example, if SIBMCLK = 18.432 MHz and SIBSCLK = 9.216 MHz (= SIBMCLK 2), in order to generate a telecom sample rate of 7.2 KHz, the value loaded for TELFSDIV should = 39, since 9.216 MHz (40 x 32) = 7.2 KHz.

SND16:
This bit is used to configure the SIB sound input and output format as 8-bit versus 16-bit mode. Setting this bit to a logic "1" selects 16-bit mode. Clearing this bit to a logic "0" selects 8-bit mode.

SNDFSDIV(6:0):
These bits select the divider modulus for the 7-bit programmable counter used to generate the sound sample rate clock, which is derived by dividing down an internal SIB clock of rate equal to the 4 times the SIB frame rate or 2 times the SIB subframe rate (where each frame consists of 128 serial data bits). This programmable divider consists of a down-counter which counts down from SNDFSDIV to zero, so the value loaded for SNDFSDIV should be the desired divider modulus - 1.

For example, if SIBMCLK = 18.432 MHz and SIBSCLK = 9.216 MHz (= SIBMCLK 2), in order to generate a sound sample rate of 24 KHz, the value loaded for SNDFSDIV should = 11, since 9.216 MHz (12 x 32) = 24 KHz.

SELTELSF1:
This bit is used to select between SIB subframe 0 and subframe 1 for the telecom data source and destination. Setting this bit to a logic "1" selects SIB subframe 1 as the telecom source. Clearing this bit to a logic "0" selects SIB subframe 0 as the telecom source.

SELSNDSF1:
This bit is used to select between SIB subframe 0 and subframe 1 for the sound data source and destination. Setting this bit to a logic "1" selects SIB subframe 1 as the sound source. Clearing this bit to a logic "0" selects SIB subframe 0 as the sound source. The subframe 1 is not currently supported.

ENTEL:
This bit is used to enable/disable telecom processing for the SIB module. Setting this bit to a logic "1" enables telecom processing. Clearing this bit to a logic "0" disables telecom processing. This bit should not be set until after the SIB module is setup, then ENSIB asserted.

Special timing restrictions are required whenever an enable/disable command is sent to the UCB1200 and/or other external codec device (see Figure 14-3). First, after SIBSF0INT (if SELTELSF1 = "0") or SIBSF1INT (if SELTELSF1 = "1") is asserted, the CPU then writes a telecom codec enable command to the UCB1200 via the SF0AUX (if SELTELSF1 = "0") or SF1AUX (if SELTELSF1 = "1") register. The CPU then asserts ENTEL. Both of these CPU transactions must occur within the same SIB frame period, which is before the next SIBSF0INT (if SELTELSF1 = "0") or SIBSF1INT (if SELTELSF1 = "0"). This ensures that the sample rate counters within the PR31700 and the UCB1200 are fully phase-synchronized.

ENSND:
This bit is used to enable/disable sound processing for this SIB module. Setting this bit to a logic "1" enables sound processing. Clearing this bit to a logic "0" disables sound processing. This bit should not be set until after the SIB module is setup, then ENSIB asserted.

Special timing restrictions are required whenever an enable/disable command is sent to the UCB1200 and/or other external codec device (see Figure 14-3). First, after SIBSF0INT (if SELSNDSF1 = "0") or SIBSF1INT (if SELSNDSF1 = "1") is asserted, the CPU then writes a sound codec enable command to the UCB1200 via the SF0AUX (if SELSNDSF1 = "0") or SF1AUX (if SELSNDSF1 = "1") register. The CPU then asserts ENSND. Both of these CPU transactions must occur within the same SIB frame period, which is before the next SIBSF0INT (if SELSNDSF1 = "0") or SIBSF1INT (if SELSNDSF1 = "0"). This ensures that the sample rate counters within the PR31700 and the UCB1200 are fully phase-synchronized.

SIBLOOP:
This bit is used for IC testing and should not be set. Setting this bit to a logic "1" will cause the SIB serial transmitted data to be internally looped back to the SIB serial receive data path. The data is inverted when this mode is selected. Setting this bit to a logic "0" selects the normal SIBDIN pin as the SIB serial receive data source.

ENSF1:
This bit is used to enable/disable SIB subframe 1 processing. Setting this bit to a logic "1" enables SIB subframe 1. Clearing this bit to a logic "0" disables SIB subframe 1, causing the SIB serial transmitted data during this subframe to be zeroed and all received data during this subframe to not be processed by the SIB module. This bit should be set before ENSIB is asserted.

ENSF0:
This bit is used to enable/disable SIB subframe 0 processing. Setting this bit to a logic "1" enables SIB subframe 0. Clearing this bit to a logic "0" disables SIB subframe 0, causing the SIB serial transmitted data during this subframe to be zeroed and all received data during this subframe to not be processed by the SIB module. This bit should be set before ENSIB is asserted.

ENSIB:
This bit is used to enable/disable the SIB module. Setting this bit to a logic "1" enables the SIB module. Clearing this bit to a logic "0" disables the SIB module and keeps the module in a reset state.

12.4.7 SIB Sound TX Holding Register

OFFSET = $078: **write-only**

Bit	Label	RESET	Read/Write
31-0	SNDTXHOLD(31:0)	X	W

SNDTXHOLD(31:0): **write-only**
These bits represent the sound data to be transmitted. Sound data can be either written directly to this register by the CPU or transparently read from the sound TX DMA buffer to this register. This register should only be loaded by the CPU after the SNDININT interrupt is asserted. The sound and telecom data processing can be configured from among several possible formats (mono versus stereo and 8-bit versus 16-bit data formats); see Table 12-4 for a summary of these formats.

12.4.8 SIB Sound RX Holding Register

OFFSET = $078: **read-only**

Bit	Label	RESET	Read/Write
31-0	SNDRXHOLD(31:0)	-	R

SNDRXHOLD(31:0): **read-only**

These bits represent the sound data to be received. Sound data can be either read directly from this register by the CPU or transparently written to the sound RX DMA buffer from this register. This register should only be read by the CPU after the SNDININT interrupt is set. The sound and telecom data processing can be configured from among several possible formats (mono versus stereo and 8-bit versus 16-bit data formats); see Table 12-4 for a summary of these formats.

12.4.9 SIB Telecom TX Holding Register

OFFSET = $07C: **write-only**

Bit	Label	RESET	Read/Write
31-0	TELTXHOLD(31:0)	X	W

TELTXHOLD(31:0): **write-only**

These bits represent the telecom data to be transmitted. Telecom data can be either written directly to this register by the CPU or transparently read from the telecom TX DMA buffer to this register. This register should only be loaded by the CPU after the TELININT interrupt is asserted. The sound and telecom data processing can be configured from among several possible formats (mono versus stereo and 8-bit versus 16-bit data formats); see Table 12-4 for a summary of these formats.

12.4.10 SIB Telecom RX Holding Register

OFFSET = $07C: **read-only**

Bit	Label	RESET	Read/Write
31-0	TELRXHOLD(31:0)	-	R

TELRXHOLD(31:0): **read-only**

These bits represent the telecom data to be received. Telecom data can be either read directly from this register by the CPU or transparently written to the telecom RX DMA buffer from this register. This register should only be read by the CPU after the TELININT interrupt is set. The sound and telecom data processing can be configured from among several possible formats (mono versus stereo and 8-bit versus 16-bit data formats); see Table 12-4 for a summary of these formats.

12.4.11 SIB Subframe 0 Control Register

OFFSET = $080:

Bit	Label	RESET	Read/Write
31-0	SF0AUX(31:0)	X	R/W

SF0AUX(31:0):

These bits represent the control data to be transmitted during SIB subframe 0. This register can be loaded by the CPU asynchronously with respect to the SIB frame timing (although special timing restrictions are required whenever an enable/disable command is sent to the UCB1200 and/or other external codec device). Whenever this register is not updated, the previous contents are transmitted during consecutive SIB frames.

12.4.12 SIB Subframe 1 Control Register

OFFSET = $084:

Bit	Label	RESET	Read/Write
31-0	SF1AUX(31:0)	X	R/W

SF1AUX(31:0):

These bits represent the control data to be transmitted during SIB subframe 1. This register can be loaded by the CPU asynchronously with respect to the SIB frame timing (although special timing restrictions are required whenever an enable/disable command is sent to the UCB1200 and/or other external codec device). Whenever this register is not updated, the previous contents are transmitted during consecutive SIB frames.

12.4.13 SIB Subframe 0 Status Register

OFFSET = $088: **read-only**

Bit	Label	RESET	Read/Write
31-0	SF0STAT(31:0)	-	R

SF0STAT(31:0): **read-only**

These bits represent the status data to be read during SIB subframe 0. This register can be read by the CPU asynchronously with respect to the SIB frame timing. Whenever this register is not updated from the UCB1200 and/or other external codec device, the previous contents are transmitted during consecutive SIB frames.

12.4.14 SIB Subframe 1 Status Register

OFFSET = $08C: **read-only**

Bit	Label	RESET	Read/Write
31-0	SF1STAT(31:0)	-	R

SF1STAT(31:0): **read-only**

These bits represent the status data to be read during SIB subframe 1. This register can be read by the CPU asynchronously with respect to the SIB frame timing. Whenever this register is not updated from the UCB1200 and/or other external codec device, the previous contents are transmitted during consecutive SIB frames.

12.4.15 SIB DMA Control Register

OFFSET = $090:

Bit	Label	RESET	Read/Write
31	SNDBUFF1TIME	0	R/W
30	SNDDMALOOP	0	R/W
29-18	SNDDMAPTR(13:2)	-	R
17	ENDMARXSND	0	R/W
16	ENDMATXSND	0	R/W
15	TELBUFF1TIME	0	R/W
14	TELDMALOOP	0	R/W
13-2	TELDMAPTR(13:2)	-	R
1	ENDMARXTEL	0	R/W
0	ENDMATXTEL	0	R/W

SNDBUFF1TIME:

The sound DMA controller supports two buffer addressing modes depending on the state of this bit. When SNDBUFF1TIME is set to a logic "1", the sound DMA controller will stop executing when it reaches the end of the DMA buffer. When SNDBUFF1TIME is cleared to a logic "0", the sound DMA controller will loop back to the start of the DMA buffer when the end of the DMA buffer is reached and will continue operating in a continuous and circular manner.

SNDDMALOOP:

The sound DMA controller supports two full-duplex loopback modes depending on the state of this bit. When SNDDMALOOP is set to a logic "1", the sound DMA controller issues RX DMA requests first, followed by TX DMA requests. This ordering allows an RX-to-TX immediate loopback via the DMA buffer. When SNDDMALOOP is cleared to a logic "0", the sound DMA controller issues TX DMA requests first, followed by RX DMA requests. This ordering allows a single circular DMA buffer to be used for both TX and RX, if so desired.

SNDDMAPTR(13:2): **read-only**

These bits provide the status of the sound DMA counter.

ENDMARXSND:

This bit enables the sound DMA receive function. Setting this bit to a logic "1" enables the DMA mode. Clearing this bit to a logic "0" disables the DMA mode. This bit should not be set until the SNDRXSTART, SNDTXSTART, and SIBSIZE registers are setup and the SIB module is enabled (ENSIB asserted). Either ENDMARXSND or ENDMATXSND or both can be set at a time since the sound DMA controller can support full duplex operation.

ENDMATXSND:
This bit enables the sound DMA transmit function. Setting this bit to a logic "1" enables the DMA mode. Clearing this bit to a logic "0" disables the DMA mode. This bit should not be set until the SNDRXSTART, SNDTXSTART, and SIBSIZE registers are setup and the SIB module is enabled (ENSIB asserted). Either ENDMARXSND or ENDMATXSND or both can be set at a time since the sound DMA controller can support full duplex operation.

TELBUFF1TIME:
The telecom DMA controller supports two buffer addressing modes depending on the state of this bit. When TELBUFF1TIME is set to a logic "1", the telecom DMA controller will stop executing when it reaches the end of the DMA buffer. When TELBUFF1TIME is cleared to a logic "0", the telecom DMA controller will loop back to the start of the DMA buffer when the end of the DMA buffer is reached and will continue operating in a continuous and circular manner.

TELDMALOOP:
The telecom DMA controller supports two full-duplex loopback modes depending on the state of this bit. When TELDMALOOP is set to a logic "1", the telecom DMA controller issues RX DMA requests first, followed by TX DMA requests. This ordering allows an RX-to-TX immediate loopback via the DMA buffer. When TELDMALOOP is cleared to a logic "0", the telecom DMA controller issues TX DMA requests first, followed by RX DMA requests. This ordering allows a single circular DMA buffer to be used for both TX and RX, if so desired.

TELDMAPTR(13:2): read-only
These bits provide the status of the telecom DMA counter.

ENDMARXTEL:
This bit enables the telecom DMA receive function. Setting this bit to a logic "1" enables the DMA mode. Clearing this bit to a logic "0" disables the DMA mode. This bit should not be set until the TELRXSTART, TELTXSTART, and SIBSIZE registers are setup and the SIB module is enabled (ENSIB asserted). Either ENDMARXTEL or ENDMATXTEL or both can be set at a time since the telecom DMA controller can support full duplex operation.

ENDMATXTEL:
This bit enables the telecom DMA transmit function. Setting this bit to a logic "1" enables the DMA mode. Clearing this bit to a logic "0" disables the DMA mode. This bit should not be set until the TELRXSTART, TELTXSTART, and SIBSIZE registers are setup and the SIB module is enabled (ENSIB asserted). Either ENDMARXTEL or ENDMATXTEL or both can be set at a time since the telecom DMA controller can support full duplex operation.

Chapter 13 SPI Module

PR31700 V0.3

This section describes the Serial Peripheral Interface (SPI) logic used to interface to devices in the system that support the SPI protocol or similar protocols.

13.1 Overview

The SPI is a serial interface consisting of clock, data out, and data in. The SPI is used to interface to devices such as serial power supplies, serial A/D converters, and other devices that contain simple serial clock and data interfaces. The PR31700 is always the master and generates the clock. Multiple slave devices can share the SPI by using a unique chip select for each slave device. The chip select can be generated using one of the general purpose I/O ports on the PR31700 or the UCB1200, or using some other output port available in the system. When a device is selected by asserting it's chip select, the device will shift data in using the SPICLK and SPIOUT signals and the device will shift data out using the SPIIN signal. When a device is not selected then the data output connected to SPIIN must be tri-stated so other devices can share the SPIIN signal. The SPI Module contains registers which provide programmability for SPICLK rate, MSB first versus LSB first, clock polarity, data phase polarity, and byte mode versus word mode operation.

13.1.1 Related Pins

SPICLK: **OUTPUT**

This pin is used to clock data in and out of the slave device.

SPIOUT: **OUTPUT**

This pin contains the data that is shifted into the slave device.

SPIIN: **INPUT**

This pin contains the data that is shifted out of the slave device.

13.2 Description

13.2.1 Block Diagram

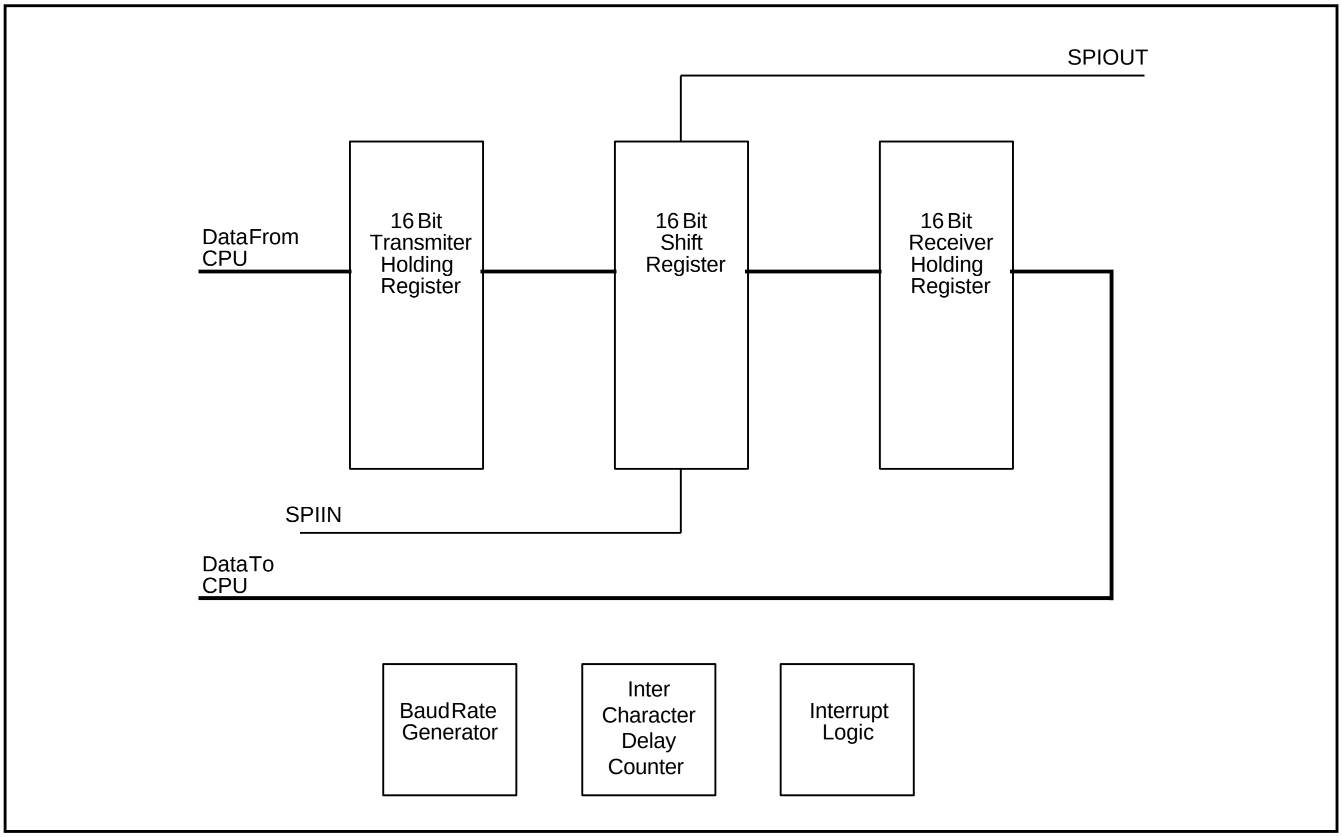


Figure 13-1. SPI Block Diagram

The SPI Module primarily consists of a 16-bit Shift Register, a 16-bit Transmitter Holding Register, a 16-bit Receiver Holding Register, a Baud Rate Generator, an Inter Character Delay Counter, and Interrupt Logic. A block diagram of the SPI Module is shown in Figure 13-1.

13.2.2 Baud Rate Generator

The rate of the SPICLK signal is determined by the value of the BAUDRATE(3:0) bits in the SPI Control Register. The BAUDRATE(3:0) bits are used by the Baud Rate Generator to divide the Master SPICLK generated by the Clock Module logic. The Master SPICLK is typically set to 7.3728 MHz when the main system clock is 36.864 MHz. The BAUDRATE(3:0) bits allows the SPICLK to vary from 3.6864 MHz down to 230 KHz.

13.2.3 Transmitter/Receiver

The SPI Module is kept in a reset state until the ENSPI bit is set in the SPI Control Register. Before setting the ENSPI bit, all other control bits in the SPI Control Register should be set to the desired values. Once the ENSPI bit is set, the SPIBUFAVAILINT interrupt will be asserted to indicate that the SPI Transmitter Holding Register is available. The SPI logic will then wait until the software writes to the SPI Transmitter Holding Register. Once the software writes to the Transmitter Holding Register then the contents will be transferred to the Shift Register and shifted out to the slave device. While data is shifting out to the slave device using the SPIOUT signal, data will shift in using the SPIIN signal.

Once the data has finished shifting, the contents of the Shift Register will be loaded into the Receiver Holding Register and the SPIRCVINT Interrupt will be asserted to indicate that there is valid receive data in the Receiver Holding Register. Once the contents of the Transmitter Holding Register are transferred to the Shift Register, the SPIBUFAVAILINT interrupt is again asserted to indicate that the Transmitter Holding register is once again available. Thus, as long as the software can keep the Transmitter Holding Register serviced before the data shifts out of the Shift Register, the SPI can maintain seamless data transfer. If the software fails to keep up with the transfer rate, then the SPI will simply wait until the next data is written to the Transmitter Holding Register.

The SPI supports either 8-bit per character or 16-bit per character operation, as defined by the WORD bit in the SPI Control Register. The software can also select whether the MSB or LSB should shift first using the LSB control bit in the SPI Control Register. Another set of control bits (CLKPOL and PHAPOL) select the polarity of the SPICLK and determine whether data should be sampled on the rising or falling edge of the SPICLK. SPI timing is shown in Figure 13-2.

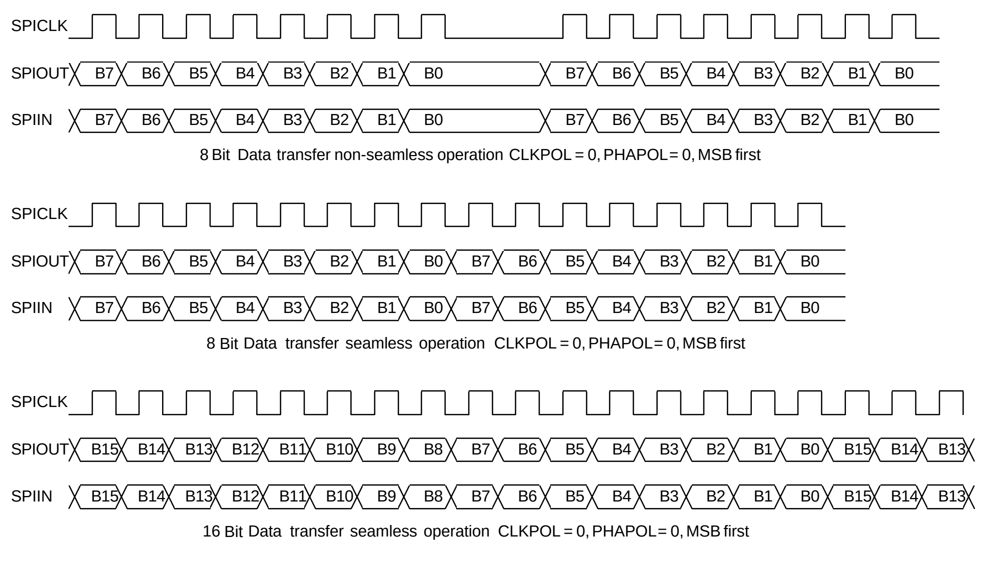

SPICLK
SPIOUT B7 B6 B5 B4 B3 B2 B1 B0 B7 B6 B5 B4 B3 B2 B1 B0
SPIIN B7 B6 B5 B4 B3 B2 B1 B0 B7 B6 B5 B4 B3 B2 B1 B0
8 Bit Data transfer non-seamless operation CLKPOL = 0, PHAPOL= 0, MSB first
SPICLK
SPIOUT B7 B6 B5 B4 B3 B2 B1 B0 B7 B6 B5 B4 B3 B2 B1 B0
SPIIN B7 B6 B5 B4 B3 B2 B1 B0 B7 B6 B5 B4 B3 B2 B1 B0
8 Bit Data transfer seamless operation CLKPOL = 0, PHAPOL= 0, MSB first
SPICLK
SPIOUT B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 B15 B14 B13
SPIIN B15 B14 B13 B12 B11 B10 B9 B8 B7 B6 B5 B4 B3 B2 B1 B0 B15 B14 B13
16 Bit Data transfer seamless operation CLKPOL = 0, PHAPOL= 0, MSB first

Figure 13-2. SPI Timing

13.2.4 CLKPOL/PHAPOL

The CLKPOL and PHAPOL bits in the SPI Control Register determine the idle phase of SPICLK and the valid clock edge for sampling data. Figure 13-3 shows the four possible combinations.

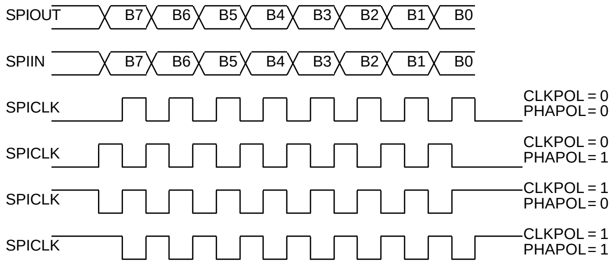


Figure 13-3. CLKPOL/PHAPOL Timing

Sometimes it is desirable to guarantee a minimum time between groups of data. The Inter Character Delay Counter is used to provide delay between groups of data. If WORD mode is selected in the SPI Control Register, delay will be inserted after 16 bits of data are shifted. If WORD mode is not selected, delay will be inserted after 8 bits of data are shifted, as shown in Figure 13-4. Inter character delay is added by setting the DELAYVAL(3:0) bits to a value other than $0. The number stored in these bits will directly correspond to the number of SPICLK periods of delay that will be inserted between characters. A zero value for these bits will imply seamless operation and the SPI will shift data and provide clocks continuously as long as the software keeps up with the transmitter rate.

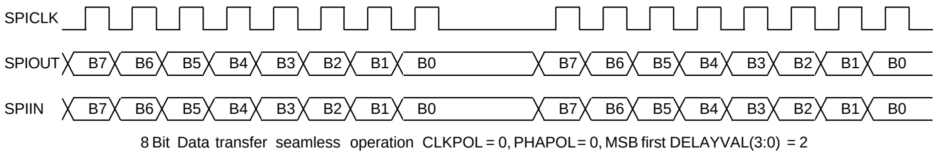

SPICLK
SPIOUT
SPIIN
B7 B6 B5 B4 B3 B2 B1 B0
B7 B6 B5 B4 B3 B2 B1 B0
8 Bit Data transfer seamless operation CLKPOL = 0, PHAPOL = 0, MSB first DELAYVAL(3:0) = 2

Figure 13-4. Inter Character Delay Counter

SPIBUFAVAILINT:

This interrupt is set when the ENSPI bit is first asserted and subsequently when the contents of the SPI Transmitter Holding Register are transferred to the SPI Shift Register. This interrupt is used to indicate that the SPI Transmitter Holding Register is available to be written by the software. See Section 8 for information on how to read and clear interrupts.

SPIERRINT:

This interrupt is set whenever the SPI Transmitter Holding Register is written, but the SPIBUFAVAILINT has not set to indicate that the register is available. This interrupt serves as a overrun indication for the software. See Section 8 for information on how to read and clear interrupts.

SPIRCVINT:

This interrupt is whenever the contents of the SPI Shift Register are transferred to the SPI Receiver Holding Register. This interrupt is used to indicate that there is valid data in the SPI Receiver Holding Register to be read by the software. See Section 8 for information on how to read and clear interrupts.

SPIEMPTYINT:

This interrupt is set whenever the both the SPI Shift Register and the SPI Transmitter Holding Register are empty. This interrupt can be used by the software to determine when the SPI is idle. See Section 8 for information on how to read and clear interrupts.

13.3 SPI Registers

13.3.1 SPI Control Register

OFFSET = $160:

Bit	Label	RESET	Read/Write
31-18	Reserved		
17	SPION	0	R
16	EMPTY	1	R
15-12	DELAYVAL(3:0)	X	R/W
11-8	BAUDRATE(3:0)	X	R/W
7-6	Reserved		
5	PHAPOL	0	R/W
4	CLKPOL	0	R/W
3	Reserved		
2	WORD	0	R/W
1	LSB	0	R/W
0	ENSPI	0	R/W

SPION: **read-only**

When the ENSPI bit is disabled, the SPI will not shut down until the Transmitter Holding Register and Shift Register are both empty, in order to make sure that any data still in the SPI is shifted out. This status bit allows the software to know when the module has shut down as a result of clearing the ENSPI control bit.

EMPTY: **read-only**

This bit is asserted if both the Transmitter Holding Register and Shift Register are empty and the Inter Character Delay Counter is finished inserting delay. This status bit allows the software to know when the SPI is idle.

DELAYVAL(3:0):

These bits define the number SPICLK periods of delay to insert between 16-bit or 8-bit characters, depending on whether WORD mode is selected or not. The value of these bits directly corresponds to the number of SPICLK periods of delay. Thus, a value of $0 will provide seamless operation with no inter character delay.

BAUDRATE(3:0):

These bits define the rate of the SPICLK. These bits divide the Master SPICLK of 7.3728 MHz to generate the desired SPICLK rate, as given by the following equation:

$$\text{SPICLK Rate} = \frac{7.\ 3728\ \text{MHz}}{\text{BAUDRATE(3:0)} * 2 + 2}$$

PHAPOL:

Setting this bit will cause the transmitter to clock data out on the rising edge of SPICLK, to be sampled by the peripherals on the falling edge of SPICLK.

CLKPOL:

Setting this bit will cause the SPICLK signal to idle high instead of low.

WORD:

Setting this bit will cause the SPI to shift 16 bits of data in and out. If this bit is cleared, 8 bits of data are shifted.

LSB:

Setting this bit will cause the data to be shifted out to the slave devices and in from the slave devices LSB-first instead of MSB-first.

ENSPI:

Setting this bit will enable the SPI Module. This bit should only be set after all other control bits have been set to the desired values. When this bit is cleared, the SPI will remain active until the Shift Register and Transmitter Holding Register are both empty. Once both registers are empty, the SPI will shut down and all the logic will be held in a reset state.

13.3.2 SPI Transmitter Holding Register

OFFSET = $164: **write-only**

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	TXDATA(15:0)	X	W

TXDATA(15:0): **write-only**

These bits are the data that is loaded into the Transmitter Holding Register. This register should only be loaded after the SPIBUFAVAILINT interrupt is asserted. If WORD mode is not set, only bits 7:0 are valid.

13.3.3 SPI Receiver Holding Register

OFFSET = $164: **read-only**

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	RXDATA(15:0)	X	R

RXDATA(15:0): **read-only**

These bits are the receive data in the Receiver Holding Register. The bits are only valid after the SPIBUFAVAILINT interrupt is asserted. If WORD mode is not set, only bits 7:0 are valid.

Chapter 13
SPI Module

Chapter 14 Timer Module

PR31700 V0.3

This section describes the Timer Module, which consists of a Real Time Clock (RTC) used to maintain the time of day and used for long wake-up events, and the Periodic Timer used by the software to provide periodic interrupts needed to monitor system events.

14.1 Overview

The Timer Module consists of two portions. The first is a 40-bit Real Time Clock (RTC) counter that uses a 32.768 KHz clock. The counter will provide a maximum count of 388 days. Also included is a 40-bit alarm register for the RTC that allows the software to set an alarm at any desired count of the RTC counter. The RTC will generate two interrupts for the CPU. The first is the ALARMINT that will generate an interrupt whenever the RTC reaches the value set by the alarm.

The second is the RTCINT that will generate an interrupt whenever the RTC counter "rolls over" after reaching a count of 388 days. The second portion of the Timer Module is the Periodic Timer used by the software to generate periodic interrupts for monitoring system events. The Periodic Timer uses the TIMERCLK generated by the Clock Module, which is normally set to 1.15 MHz. The Periodic Timer contains a programmable 16-bit counter. When enabled, the counter will count down and generate an interrupt (PERINT) whenever reaching a count of zero.

14.1.1 Related Pins

C32KIN: **INPUT**
This pin along with C32KOUT should be connected to a 32.768 KHz crystal.

C32KOUT: **OUTPUT**
This pin along with C32KIN should be connected to a 32.768 KHz crystal.

BC32K: **OUTPUT**
This pin is a buffered output of the 32.768 KHz clock.

14.2 RTC

14.2.1 RTC Block Diagram

The RTC consists of five 8-bit ripple counters, a 40-bit equality comparator, roll over detect logic and two interrupt flip-flops, as shown in Figure 14-1.

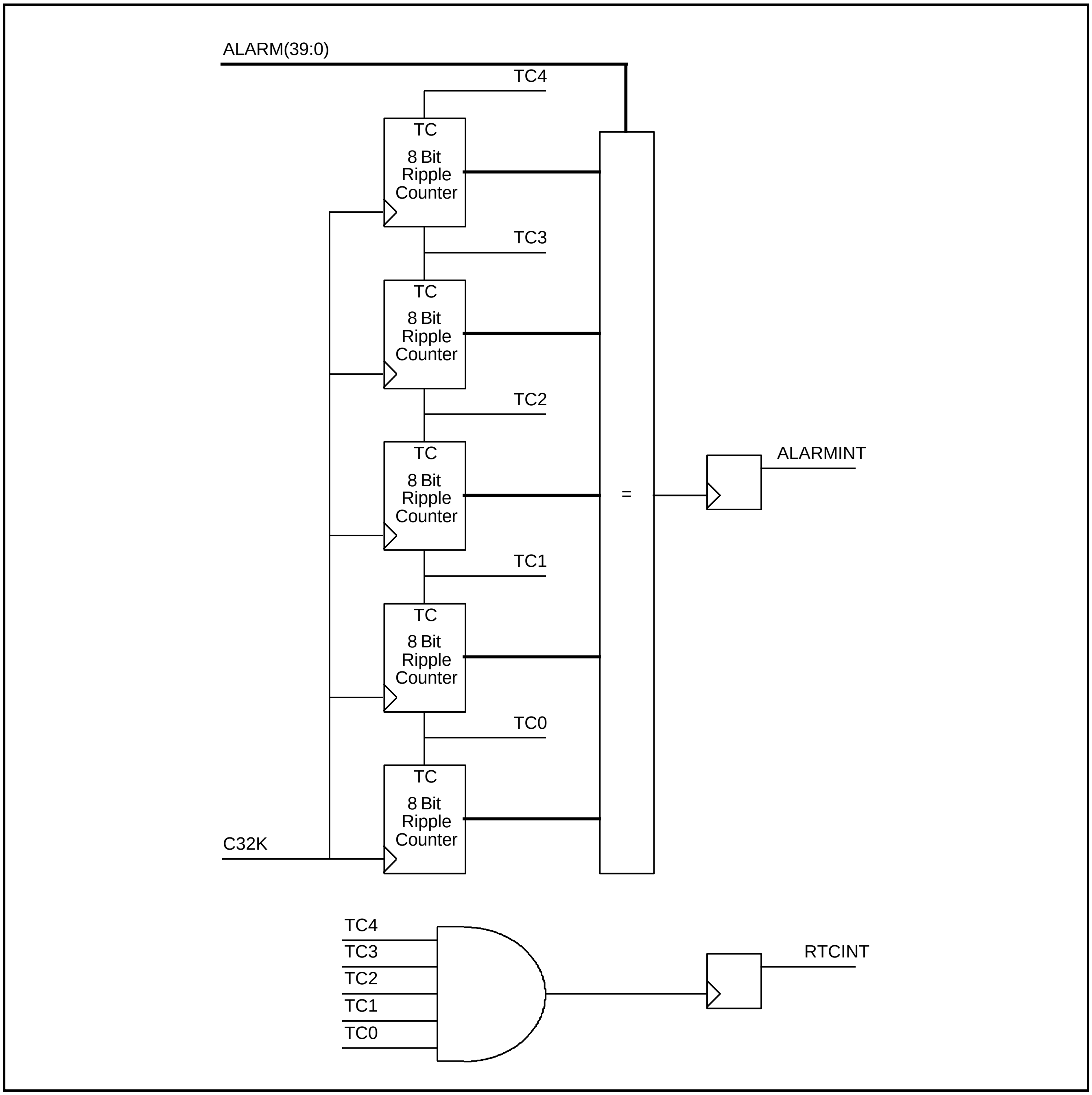


Figure 14-1. RTC block diagram

14.2.2 RTC Description

The RTC contains five 8-bit ripple counters connected in series. The first counter counts on each C32K clock, while each successive counter only counts when the previous count stage has reached a count of "$FF". Once all the counters reach a count of "$FFFFFFFFFF", the RTCINT interrupt will assert to indicate that the counter is "rolling over". Given a 40-bit counter for the RTC and an input clock C32K of 32.768 KHz, the time until the RTCINT interrupt will assert is 388 days.

The software can generate an alarm interrupt (ALARMINT) by setting the ALARM(39:0) bits in the Alarm Register. Whenever the RTC becomes equal to the value set in the Alarm Register, the ALARMINT will be triggered. The value of the RTC counter can be read via the RTC Register. The RTC counter is split into groups of five 8- bit counters to simplify IC testing.

14.2.3 RTC Interrupts

RTCINT:
This interrupt is set whenever all 40 bits of the RTC counter reach a value of "$FFFFFFFFFF" to alert the software that the counter is "rolling over". See Section 8 for information on how to read and clear interrupts.

ALARMINT:
This interrupt is set whenever the RTC counter reaches a count that is equal to the value of the ALARM(39:0) bits set in the Alarm Register. See Section 8 for information on how to read and clear interrupts.

14.3 Periodic Timer

14.3.1 Periodic Timer Block Diagram

The Periodic Timer consists of a 16-bit down counter, along with a zero detect and an interrupt flip-flop, as shown in Figure 14-2.

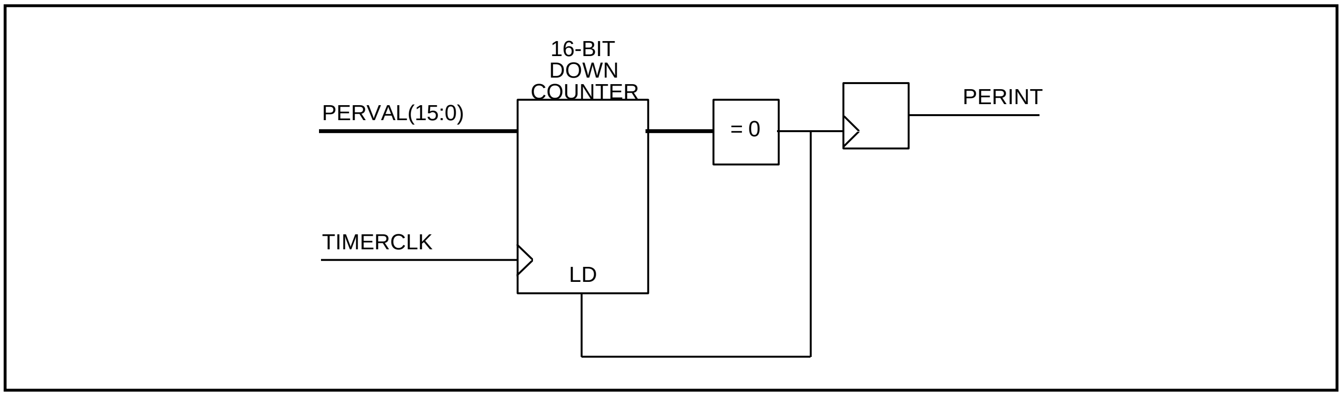


Figure 4-2. Periodic Timer block diagram

14.3.2 Periodic Timer Description

The Periodic Timer is used by the software to generate periodic interrupts needed to monitor system events. The Periodic Timer contains a 16-bit down counter whose initial value is programmed by the software via the PERVAL(15:0) control bits in the Periodic Timer Register. The counter uses the TIMERCLK signal generated by the Clock Module as the clock for the counter. With a system clock of 36.864 MHz, the TIMERCLK will normally be set to 1.15 MHz. This implies a maximum Periodic Timer count duration of 56.89 ms with a granularity of 0.87 s. The Periodic Timer is enabled by asserting the ENPERTIMER control bit in the Timer Control Register. Once enabled, the counter will load the PERVAL(15:0) control bits and down count to zero. Once the count of zero is reached, the PERINT interrupt is asserted and the counter re-loads the PERVAL(15:0) control bits. The value of the Periodic Timer counter can be read by the software via the PERCNT(15:0) status bits in the Periodic Timer Register.

14.3.3 Periodic Timer Interrupts

PERINT:

This interrupt is set whenever the Periodic Timer is enabled and the Periodic Timer counter reaches a count of zero. See Section 8 for information on how to read and clear interrupts.

14.4 Timer Registers

14.4.1 RTC Register

OFFSET = $140: **read-only**

Bit	Label	RESET	Read/Write
31-8		Reserved	XR
7-0	RTC(39:32)	X	R

OFFSET = $144: **read-only**

Bit	Label	RESET	Read/Write
31-0	RTC(31:0)	X	R

RTC(39:0): **read-only**

These bits provide the status of the 40-bit RTC counter. The software must read these bits twice and compare the values to ensure that the counter is not read while the counter is counting since the CPU clock is not synchronous with the RTC counter clock. If the two reads do not compare, the software must read the register again to read the correct count value.

14.4.2 Alarm Register

OFFSET = $148:

Bit	Label	RESET	Read/Write
31-8	Reserved		
7-0	ALARM(39:32)	X	R/W

OFFSET = $14C:

Bit	Label	RESET	Read/Write
31-0	ALARM(31:0)	X	R/W

ALARM(39:0):

Whenever the RTC counter reaches a count that is equal to these bits, the ALARMINT interrupt will be set.

14.4.3 Timer Control Register

OFFSET = $150:

Bit	Label	RESET	Read/Write
31-8	Reserved		
7	FREEZEPRE	0	R/W
6	FREEZERTC	0	R/W
5	FREEZETIMER	0	R/W
4	ENPERTIMER	0	R/W
3	RTCCLR	0	R/W
2	TESTC8MS	0	R/W
1	ENTESTCLK	0	R/W
0	ENRTCTST	0	R/W

FREEZEPRE:

Setting this bit will cause the lower 8 bits of the RTC counter to freeze. The lower 8 bits of the RTC counter are also used to generate an 8 ms reference signal for the IR Carrier Detect State Machine and the debouncers in the Power Module and IO Module. Thus, setting this bit will also cause these modules to lose their 8 ms reference signal. This bit can be used by the software to freeze the RTC counter during software debugging.

FREEZERTC:

Setting this bit will cause the upper 32 bits of the RTC counter to freeze. This bit can be used by the software to freeze the RTC counter during software debugging.

FREEZETIMER:

Setting this bit will cause the Periodic Timer counter to freeze. This bit can be used by the software to freeze the Periodic Timer counter during software debugging.

ENPERTIMER:

Setting this bit will cause the Periodic Timer to load the PERVAL(15:0) control bits and begin down counting. The

PERINT interrupt will only be asserted if this bit is enabled.

RTCCLR:
Setting this bit to a logic "1" will cause all 40 bits of the RTC counter to initialize to "$0000000000". The RTC counter will stay cleared and the counter will not start counting until this bit is cleared back to a logic "0".

TESTC8MS:
Setting this bit will cause the 8 ms reference pulse used by the IR logic and the debouncer circuits to run at 61 s instead of 8 ms. This bit is used for IC testing to speed up the counters and should normally never be set by the software.

ENTESTCLK:
Setting this bit will cause the 32 KHz input clock for the RTC counter to be driven by the TIMERCLK instead of the 32 KHz input pin. This bit is used for IC testing to speed up the amount of time needed to test the RTC counter and should normally never be set by the software.

ENRTCTST:
Setting this bit will cause all five of the 8-bit counters that comprise the RTC counter to count together. This provides a mechanism for fully exercising all the counters in a timely fashion for IC testing. The software may wish to set this bit in order to test the RTCINT interrupt since normally it would take 388 days to generate this interrupt.

14.4.4 Periodic Timer Register

OFFSET = $154:

Bit	**Label**	**RESET**	**Read/Write**
31-16	PERCNT(15:0)	X	R
15-0	PERVAL(15:0)	X	R/W

PERCNT(15:0): read-only
These bits provide the status of the Periodic Counter. The software should continually read this register until two consecutive reads provide the same value, in order to ensure that the register is not read while the counter is counting, since the TIMERCLK is not synchronous with the CPU clock.

PERVAL(15:0):
These bits are loaded into the Periodic Timer counter when the counter is enabled or when the counter reaches a count of zero. The TIMERCLK is normally set to 1.15 MHz, which implies the following equation for the time between the PERINT interrupts:

$$\text{Interrupt Rate} = \frac{\text{PERVAL(15:0)} + 1}{1.15\ \text{MHz}}$$

Chapter 15
UART Module

PR31700 V0.3

This section describes the dual Universal Asynchronous Receiver/Transmitter (UART) Module logic.

15.1 Overview

The UART Module contains two identical, fully independent, full duplex UARTs: UARTA and UARTB. Each UART contains the following features:

- adjustable baud rate counter from 230 KHz down to 225 Hz
- single buffered transmit register
- double buffered receive register
- DMA for either transmit or receive
- full duplex
- even, odd or no parity
- 7 or 8 bits per character
- one or two stop bits per character
- pulse option mode to support IRDA Infrared protocol

15.1.1 Related Pins

TXD: OUTPUT
This pin is the UART transmit signal from the UARTA module.

RXD: INPUT
This pin is the UART receive signal to the UARTA module.

IROUT: OUTPUT
This pin is the UART transmit signal from the UARTB module or the Consumer IR output signal if Consumer IR mode is enabled.

IRIN: INPUT
This pin is the UART receive signal to the UARTB module.

15.2 Overall Operation

The operation of each UART is controlled through six registers: Control 1 Register, Control 2 Register, DMA Control 1 Register, DMA Control 2 Register, DMA Count Register, and Transmit/Receive Holding Register.

15.2.1 Power On/Off

Each UART is powered on and off with the ENUART bit of the Control 1 Register. When the ENUART bit is cleared, the UART will power off only after all data in its transmit shift and holding registers has been clocked out. The on/off status of the UART can be monitored via the UARTON bit of the Control 1 Register. While powered off, the UART is completely disabled.

In addition to being powered on, each UART must also have its clock enabled to operate. The UART clock enables, as well as the UART clock frequency, is controlled by the Clock Module (see Section 6).

15.2.2 Baud Rate and Communication Parameters

The baud rate, bits per symbol (7 or 8), parity type (none, even, or odd), and line polarity (normal or inverted) apply to both the transmitter and receiver and may not be independently selected for each direction. All of these except the baud rate are selected with individual bits of the Control 1 Register (see Register Descriptions in Section 15.5).

The baud rate for each UART is determined by the BAUDRATE(9:0) value in the Control 2 Register and the UART clock frequency (nominally 3.6864 MHz). The following equation determines the baud rate:

Baud Rate = 3.6864 MHz   ((BAUDRATE(9:0) + 1) * 16)

The UART clock frequency is determined by the Clock Module configuration (see Section 6). Assuming the intended frequency of 3.6864 MHz, Table 15-1 shows the value that should be written to the BAUDRATE(9:0) control bits to generate the standard baud rates.

Table 15-1. Standard UART Baud Rates

baudrate	BAUDRATE(9:0)
38400	5
19200	11
9600	23
4800	47
2400	95
1200	191
600	383
300	767

15.2.3 Interrupt Operation

Each UART may be configured to generate CPU interrupts for a number of events. When any of these events occur, a corresponding status bit will be set in the Interrupt Status 2 Register (see Section 8 for details on the Interrupt Module). Table 15-2 lists the UART events that can generate interrupts. The entries are in terms of the mnemonic used in this document and the actual event. In the mnemonic the lower-case 'n' is replaced with 'A' for UART A or 'B' for UART B.

Table 15-2. UART Interrupt Events

mnemonic	interrupting event
UARTnRXINT	receiver holding register becomes full
UARTnRXOVERRUNINT	receiver shift register becomes full when both holding register and PRXHOLD are full
UARTnFRAMEERRINT	receiver does not detect stop bit-character not 0x0
UARTnBREAKINT	receiver detects BREAK condition-character is 0x0 and no stop bit
UARTnPARITYERRINT	receiver detects parity bit error
UARTnTXINT	transmit holding register becomes empty
UARTnTXOVERRUNINT	transmit holding register written to when full
UARTnEMPTYINT	transmit shift register becomes empty when holding register is also empty
UARTnDMAFULLINT	DMA controller reaches end of specified buffer
UARTnDMAHALFINT	DMA controller reaches half-way point in specified buffer

While these events cause a bit to be set in Interrupt Status 2 Register, they will not cause an interrupt unless they are configured to do so. Interrupts are enabled through the Enable Interrupt 2 Register. Interrupt status bits are cleared through the Clear Interrupt 2 Register. Note that interrupt status bits are set and need to be cleared regardless of whether or not the interrupt is enabled. See Section 8 for more general information about the PR31700 interrupts.

15.2.3.1 Responding to Interrupt Status Bits

When responding to an event (whether via interrupt or polling), it is important to clear the interrupt status bit before taking the action that will re-enable the event. This will ensure that an event will not be missed.

For example, when the Receive Holding Register becomes full, the UARTnRXINT bit will get set. If the Receive Holding Register is read before the UARTnRXINT bit is cleared, it is possible that sometime after the Receive Holding Register becomes empty and before the UARTnRXINT bit is cleared, a second character will be transferred to the Receive Holding Register. This transfer will also cause the UARTnRXINT bit to be set. Obviously, if the CPU (still reacting to the first event) then clears the UARTnRXINT bit, the second event will be missed. As a result, the CPU will not be "aware" that the Receive Holding Register is full and an overrun interrupt will eventually occur.

15.2.3.2 Related Interrupts

UARTARXINT:
Issues an interrupt whenever the UARTA Receive Holding Register is loaded with data.

UARTARXOVERRUNINT:
Issues an interrupt if the UARTA Receive Holding Register is loaded twice before the interrupt is service.

UARTAFRAMEERRINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register contains a frame error.

UARTABREAKINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register is a break.

UARTAPARITYERRINT:
Issues an interrupt if the current data in the UARTA Receive Holding Register contains a parity error.

UARTATXINT:
Issues an interrupt if the UARTA Transmit Holding Register is available.

UARTATXOVERRUNINT:
Issues an interrupt if the UARTA Transmit Holding Register is written to when the Transmit Holding Register is not available.

UARTAEMPTYINT:
Issues an interrupt if the UARTA Transmit Holding Register and Transmit Shift Register are both empty.

UARTADMAFULLINT:
Issues an interrupt if the UARTA DMA counter reaches the end of the buffer.

UARTADMAHALFINT:
Issues an interrupt if the UARTA DMA counter reaches the mid point of the buffer.

UARTBRXINT:
Issues an interrupt whenever the UARTB Receive Holding Register is loaded with data.

UARTBRXOVERRUNINT:
Issues an interrupt if the UARTB Receive Holding Register is loaded twice before the interrupt is service.

UARTBFRAMEERRINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register contains a frame error.

UARTBBREAKINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register is a break.

UARTBPARITYERRINT:
Issues an interrupt if the current data in the UARTB Receive Holding Register contains a parity error.

UARTBTXINT:
Issues an interrupt if the UARTB Transmit Holding Register is available.

UARTBTXOVERRUNINT:
Issues an interrupt if the UARTB Transmit Holding Register is written to when the Transmit Holding Register is not available.

UARTBEMPTYINT:
Issues an interrupt if the UARTB Transmit Holding Register and Transmit Shift Register are both empty.

UARTBDMAFULLINT:
Issues an interrupt if the UARTB DMA counter reaches the end of the buffer.

UARTBDMAHALFINT:
Issues an interrupt if the UARTB DMA counter reaches the mid point of the buffer.

15.2.4 DMA Operation

Each UART has one DMA channel which may be used with either the transmitter or the receiver. DMA operation involves specifying a memory buffer and enabling DMA. Once enabled, characters will flow either from the memory buffer to the transmitter, or from the receiver to the memory buffer.

The memory buffer is specified in terms of its start address in physical address space, and its length in bytes. Note that the CPU must know the mapping between virtual and physical addresses. The start address must be written to the DMA Control 1 Register.

The two LSB's are ignored, thereby forcing the buffer to begin on a long word boundary. The buffer length minus one must be written to the DMA Control 2 Register.

Once the buffer has been specified, DMA may be enabled. This is done with either ENDMARX or ENDMATX in the Control 1 Register. Only one of these two bits should be set at any one time. Operation is undefined if both are set. Once enabled, UART DMA will begin. For the transmitter, this means that when the Transmit Holding Register becomes empty, the DMA controller will transfer a byte from the specified buffer to the Transmit Holding Register, and update its internal buffer pointer. For the receiver, this means that when the Receive Holding Register becomes full, the DMA controller will transfer a byte from the Receive Holding Register to the specified buffer, and update its internal buffer pointer.

The ENDMALOOP control bit in the Control 1 Register determines when DMA will stop. If this bit is cleared, DMA will stop when the end of the buffer is reached. If this bit is set, the DMA controller will run continuously, looping back to the start of the buffer when the buffer end is reached. In either case, DMA can be explicitly halted by clearing the ENDMARX or ENDMATX bit.

The DMA Count Register is a read-only register which contains the UART DMA controller's buffer counter status. The counter counts down so that the value in this status register indicates how much of the buffer remains to be accessed (on this pass) by the DMA controller.

There are two interrupt status bits associated with the DMA controller: UARTnDMAFULLINT and UARTnDMAHALFINT. These are set as the DMA controller buffer pointer reaches the end and the half-way point of the buffer.

15.2.5 Internal Loopback

The LOOPBACK bit of the Control 1 Register, when set, places the UART in an internal loopback mode. When in this mode, the transmit and receive lines of the UART are internally tied together, the external transmit line is driven high, and the external receive line is ignored. This loopback mode is for testing purposes only.

15.3 Transmitter Operation

Figure 15-1 shows a block diagram of the transmitter portion of each UART. Characters to be transmitted are written by the CPU to the Transmit Holding Register. This should only be done when the Transmit Holding Register is empty which is indicated by either the EMPTY bit of the Control 1 Register, or the UARTnTXINT interrupt. Writing a character to the Transmit Holding Register when it is not empty results in a transmitter overrun.

The character in the Transmit Holding Register is transferred into the Transmit Shift Register when the Transmit Shift Register becomes empty. Transferred with the character is 1 start bit, 1 or 2 stop bits (depending on the setting of TWOSTOP in the Control 1 Register), and 0 or 1 parity bits (depending on the setting of ENPARITY and EVENPARITY in the Control 1 Register). Beginning at the next baud period following the transfer, the character is shifted out to the transmit line. The start bit is shifted first, followed by the LSB through MSB of the (7-bit or 8-bit) character, followed by the parity bit (if any), and ending with the stop bit(s). As the last stop bit is shifted out, the transmitter will load the next character from the Transmit Holding Register if one has been written. If the Transmit Holding Register is empty, the Transmit Shift Register will continue to shift out stop bits (logic level "1") until a character is written by the CPU into the Transmit Holding Register.

Depending on the configuration, as few as nine bits (7-bit characters, no parity, 1 stop bit), and as many as twelve bits (8-bit characters, parity enabled, 2 stop bits) are transmitted for every character.

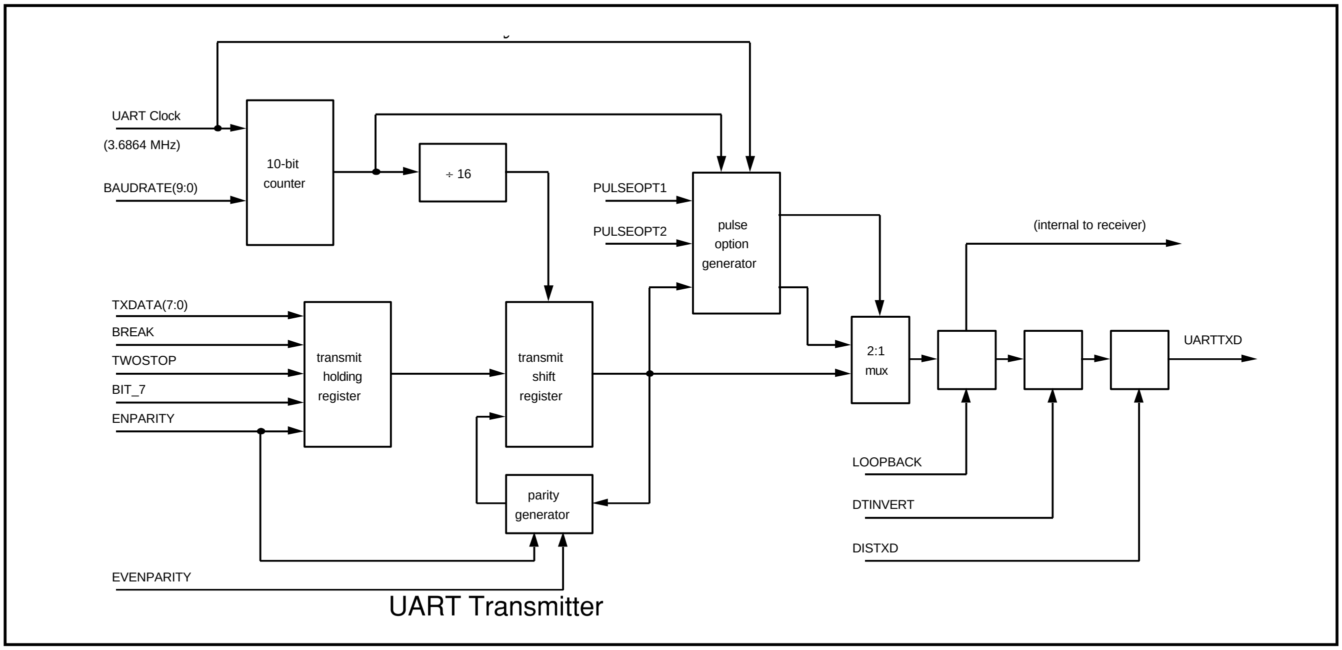


Figure 15-1. UART Transmitter

15.3.1 Transmitter Pulse Output Operation

In normal operation (neither pulse option selected), the transmit line is tied directly to the Transmit Shift Register output. Each transmitted bit causes the transmit line to go high or low for a full baud period.

To support the IRDA Infrared protocol, there are two pulse options which result in a modified signal on the output line. When either of these options are selected, transmitted zeros will cause the transmit line to go low for only a fraction of the baud period. Ones are transmitted normally. The difference between the two pulse option modes is the duration that the line will stay low for each transmitted zero.

Pulse Option 1 (selected with PULSEOPT1 of the Control 1 Register) results in a duration of six UART clock cycles. The UART clock frequency is controlled by the Clock Module (see Section 6), and is normally set to be 3.6864 MHz. For Pulse Option 1, this clock frequency would result in a low pulse of 1.63 sec(= 6 3.6864).Note that this duration is irrespective of the baud rate.

Pulse Option 2 (selected with PULSEOPT2 of the Control 1 Register) results in a duration of 3/16 of the selected baud period. The mode of operation with both pulse options selected (both PULSEOPT1 and PULSEOPT2 asserted) is reserved for a future implementation.

15.3.2 Transmitter Disable Operation

The transmit line may be disabled with the DISTXD bit of the Control 1 Register. In this mode the transmit line is held at logic level "0". Otherwise, the transmitter operates normally as described above.

15.3.3 Transmitter BREAK Operation

A BREAK may be transmitted by writing the value 0x100 to the Transmit Holding Register. The BREAK is transmitted after the character (if any) in the Transmit Shift Register is sent. The BREAK condition continues until explicitly terminated by the CPU. The CPU may continue to write the value 0x100 to the Transmit Holding Register in order to precisely control the duration of the BREAK. The BREAK condition is terminated when the CPU writes the value 0x00 to the Transmit Holding Register.

15.3.4 Transmitter Overrun

The transmit overrun condition occurs if the CPU writes a character to the Transmit Holding Register when it is not empty. This condition is indicated by the UARTnTXOVERRUNINT interrupt status bit. The character in the Transmit Holding Register resulting from an overrun is undefined.

15.4 Receiver Operation

Figure 15-2 shows a block diagram of the receiver portion of each UART. In normal operation, the receiver waits for a start bit on the receive line. One baud cycle after receiving the start bit, the next 7 or 8 bits (depending on BIT_7 in the Control 1 Register) are clocked into the Receive Shift Register. If parity is enabled (depending on ENPARITY and EVENPARITY in the Control 1 Register), the next bit is compared with the expected parity. As the stop bit is clocked in and checked, the assembled character is transferred from the Receive Shift Register to the PRXHOLD Register.

The PRXHOLD Register is a buffer between the Receive Shift Register and the Receive Holding Register. The PRXHOLDFULL bit of the Control 1 Register, when asserted, indicates that the PRXHOLD Register is full. On the first UART clock period (nominally 3.6864 MHz) that the Receive Holding Register is empty and the PRXHOLD Register is full, the character is transferred to the Receive Holding Register. This transfer results in the RXHOLDFULL bit of the Control 1 Register and the UARTnRXINT interrupt status bit both asserting.

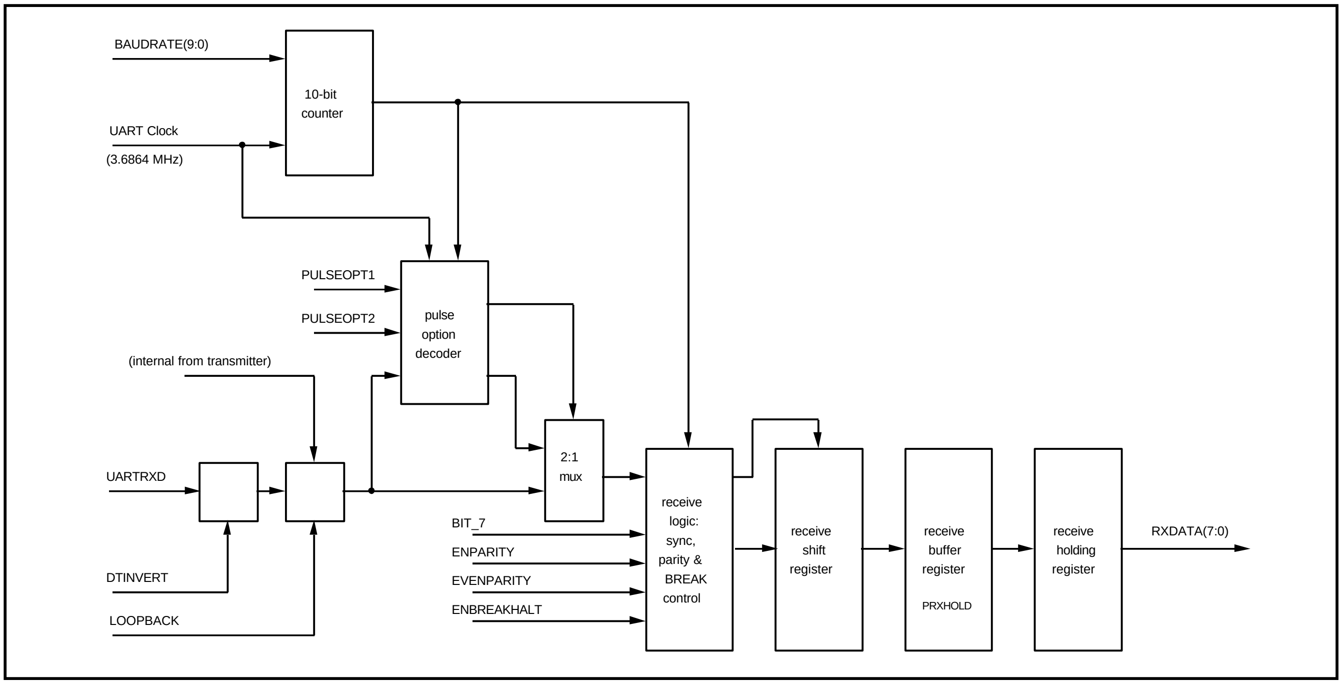


Figure 15-2. UART Receiver

15.4.1 Receiver BREAK Operation

The receiver detects a BREAK condition when the received character is 0x00 and there is no stop bit. When this occurs the receiver generates a UARTnBREAKINT interrupt. The interrupt does not assert until the BREAK character is transferred to the Receive Holding Register. In other words, the interrupt is held off until all characters preceding the BREAK are read by the CPU.

Additionally, the operation of the receiver in response to a BREAK is affected by the ENBREAKHALT bit of the Control 1 Register. Normally (with the ENBREAKHALT bit cleared), the receiver will start receiving characters again as soon as a stop bit is detected on the line. However, if the ENBREAKHALT bit is set, the receiver shuts down until the BREAK character is read by the CPU from the Receive Holding Register. While shut down, the receiver ignores all data on the receive line.

15.4.2 Receiver Frame Error Condition

A receiver frame error occurs when the receiver does not detect a stop bit following the character. This condition results in a UARTnFRAMEERRINT interrupt being generated. Like the UARTnBREAKINT interrupt, the interrupt is not asserted until the character associated with the frame error is transferred to the Receive Holding Register.

15.4.3 Receiver Overrun Condition

A receiver overrun occurs when a character is received and both the PRXHOLD Register and Receive Holding Register are full. This condition occurs at the point that the stop bit of the new character is clocked in and results in a UARTnRXOVERRUNINT interrupt. The new character will overwrite the character in the PRXHOLD Register. Like the UARTnBREAKINT interrupt, the interrupt is not asserted until the character associated with the overrun error is transferred to the Receive Holding Register.

15.4.4 Receiver Parity Error Condition

A receiver parity error occurs when parity is enabled and the receiver detects the wrong polarity of the bit following the character and preceding the stop bit. This condition results in a UARTnPARITYERRINT interrupt being generated. Like the UARTnBREAKINT interrupt, the interrupt is not asserted until the character associated with the parity error is transferred to the Receive Holding Register.

15.4.5 Receiver Pulse Operation

In normal operation the received signal corresponds to the polarity of the data for the entire baud period. When the received signal is from a pulsed transmitter, zeros are low for only a fraction of the baud period. If either pulse option is selected (with PULSEOPT1 or PULSEOPT2 of the Control 1 Register) the receiver reconstructs the data stream by extending low pulses to the full baud period.

15.5 UART Registers

15.5.1 UART Control 1 Register

OFFSET = $0B0: **UARTA**
OFFSET = $0C8: **UARTB**

Bit	Label	RESET	Read/Write
31	UARTON	0	R
30	EMPTY	1	R
29	PRXHOLDFULL	0	R
28	RXHOLDFULL	0	R
27-16	Reserved		
15	ENDMARX	0	R/W
14	ENDMATX	0	R/W
13	TESTMODE	0	R/W
12	ENBREAKHALT	0	R/W
11	ENDMATEST	0	R/W
10	ENDMALOOP	0	R/W
9	PULSEOPT2	0	R/W
8	PULSEOPT1	0	R/W
7	DTINVERT	0	R/W
6	DISTXD	1	R/W
5	TWOSTOP	0	R/W
4	LOOPBACK	0	R/W
3	BIT_7	0	R/W
2	EVENPARITY	0	R/W
1	ENPARITY	0	R/W
0	ENUART	0	R/W

UARTON: **read-only**
When the ENUART bit is disabled, the module will not shut down until the Transmit Holding Register and Transmit Shift Register are empty plus a couple of clocks. This bit provides the status as to whether the module is still enabled or not.

EMPTY: **read-only**
This bit is high if the Transmit Holding Register and Transmit Shift Register are both empty.

PRXHOLDFULL: **read-only**
The receive data path consists of an 8-bit shift register plus two 8-bit holding registers. Whenever the receiver finishes receiving a character, it will transfer the contents into the PRXHOLD Register. The contents of this register will then be loaded into the RXHOLD Register whenever this RXHOLD Register is empty. The RXHOLD Register is emptied by reading a byte from the Receive Holding Register. This PRXHOLDFULL bit provides status as to whether there is a valid byte of data in the PRXHOLD Register.

RXHOLDFULL: **read-only**
This bit provides the status of the RXHOLD Register.

ENDMARX:
This bit enables the DMA receive function. This bit should not be set until the UARTDMACNTL1 and UARTDMACNTL2 Registers are setup and the module is enabled. Only one of ENDMARX or ENDMATX can be set at a time since there is only one DMA channel.

ENDMATX:
This bit enables the DMA transmit function. This bit should not be set until the UARTDMACNTL1 and UARTDMACNTL2 registers are setup, the module is enabled and the empty flag is set. Only one of ENDMARX or ENDMATX can be set at a time since there is only one DMA channel.

TESTMODE:
This bit is used for IC testing and should never be set.

ENBREAKHALT:
Setting this bit will cause the receiver to halt after receiving a break, until the Receive Holding Register is emptied and the UARTRXD signal goes to the marking state. Otherwise, the receiver will halt until the UARTRXD signal goes to the marking state without regard to the status of the Receive Holding Register.

ENDMATEST:
This bit is used for IC testing and should not be set.

ENDMALOOP:
The DMA controller supports two modes depending on the state of this bit. When ENDMALOOP is low, the DMA controller will stop executing when it reaches the end of the DMA buffer. When ENDMALOOP is high, the DMA controller will loop back to the start of the DMA buffer when the end of the DMA buffer is reached and will continue operating.

PULSEOPT2:
Setting this bit will cause the transmitted data to pulse low for three baud clocks instead of the normal 16 baud clocks. Setting either this bit or PULSEOPT1 will cause the receiver to expect the data to be a pulsed input.

PULSEOPT1:
Setting this bit will cause the transmitted data to pulse low for a fixed six 3.6864 MHz clocks instead of the normal 16 baud clocks.

DTINVERT:
Setting this bit will cause the UARTTXD and UARTRXD signals to be inverted.

DISTXD:
Setting this bit will cause the UARTTXD signal to go low.

TWOSTOP:
Setting this bit will cause the transmitter to transmit two stop bits instead of the normal one stop bit.

LOOPBACK:
Setting this bit will cause the transmitted data to internally loop back to the receive data. The UARTTXD pin is held high when this bit is set.

BIT_7:
Setting this bit selects 7-bit per character mode instead of 8-bit per character mode.

EVENPARITY:
Setting this bit selects even parity instead of odd parity, if the ENPARITY bit is set.

ENPARITY:
Setting this bit will cause parity to be generated and received.

ENUART:
Setting this bit will enable the UART Module. When this bit is cleared the module is kept in a reset state. This bit should not be set until all other control bits are setup.

15.5.2 UART Control 2 Register

OFFSET = $0B4: **UARTA write-only**
OFFSET = $0CC: **UARTB write-only**

Bit	Label	RESET	Read/Write
31-10	Reserved		
9-0	BAUDRATE(9:0)	X	W

BAUDRATE(9:0): **write-only**
These bits define the baud rate of the transmitter and receiver. The following equation determines the baud rate:

Baud Rate = 3.6864 MHz   ((BAUDRATE(9:0) + 1) * 16)

15.5.3 UART DMA Control 1 Register

OFFSET = $0B8: **UARTA write-only**
OFFSET = $0D0: **UARTB write-only**

Bit	Label	RESET	Read/Write
31-2	DMASTARTVAL(31:2)	X	W
1-0	Reserved		

DMASTARTVAL(31:2): **write-only**
These bits define the start address for the DMA buffer.

15.5.4 UART DMA Control 2 Register

OFFSET = $0BC: **UARTA write-only**
OFFSET = $0D4: **UARTB write-only**

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	DMALENGTH(15:0)	X	W

DMALENGTH(15:0): **write-only**
These bits define the length of the DMA buffer. The last address in the DMA buffer is given by DMASTARTVAL(31:2) + DMALENGTH(15:0).

15.5.5 UART DMA Count

OFFSET = $0C0: **UARTA read-only**
OFFSET = $0D8: **UARTB read-only**

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	DMACNT(15:0)	X	R

DMACNT(15:0): **read-only**
These bits provide the status of the DMA counter.

15.5.6 UART Transmit Holding Register

OFFSET = $0C4: **UARTA write-only**
OFFSET = $0DC: **UARTB write-only**

Bit	Label	RESET	Read/Write
31-9	Reserved		
8	BREAK	X	W
7-0	TXDATA(7:0)	X	W

BREAK: **write-only**
Setting this bit, along with writing $00 to TXDATA(7:0), will cause a break to be generated. The break will continue until this bit is cleared, along with TXDATA(7:0) maintained at $00. This register should only be loaded after the UARTTXINT interrupt is set. The break will flow through the Transmit Holding Register, thus it will not start the break until the current character that is being transmitted from the Transmit Shift Register is finished.

TXDATA(7:0): **write-only**
These bits are the data to be transmitted. This register should only be loaded after the UARTTXINT interrupt is set. For 7-bit mode, only bits 6:0 are valid

15.5.7 UART Receiver Holding Register

OFFSET = $0C4: **UARTA read-only**
OFFSET = $0DC: **UARTB read-only**

Bit	Label	RESET	Read/Write
31-8	Reserved		
7-0	RXDATA(7:0)	X	R

RXDATA(7:0): **read-only**
These bits are the receive data. The bits are valid after the UARTRXINT interrupt is set or when the RXHOLD flag is asserted. For 7-bit mode, only bits 6:0 are valid.

Chapter 16
Video Module

PR31700 V0.3

This section describes the Video Module, which provides the interface logic between a video buffer and an external Liquid Crystal Display (LCD).

16.1 Overview

The Video Module within the PR31700 contains logic for transferring bit-mapped graphics data between a video buffer located in system memory to an external LCD. The LCD must be refreshed at a rate of 50 to 100 frames per second to maintain a steady picture on the display.

Data is clocked into the LCD's shift register using parallel data lines (4 or 8 bits wide). Once a horizontal line of data has been shifted into the shift register, this line of pixels is transferred to successive lines of the LCD panel. This sequence is repeated until all the lines of the display have been shifted and transferred, producing a full frame of display. At the end of the frame the pointers in the LCD return to the top of the display.

Different LCD's have different requirements on the number of horizontal and vertical pixels, the number of parallel lines required for the data interface, interface timing, and refresh rate. The Video Module inside the PR31700 is programmable in order to support many different types of LCD's. The Video Module also contains logic that produces 4-level or 16-level grey scale on a monochrome LCD using a time-based dithering algorithm. Also supported is an 8-bit per pixel color mode for interfacing to color LCD's.

16.1.1 Related Pins

FRAME: **OUTPUT**

This pin is the frame synchronization pulse signal between the Video Module and the LCD, and is used by the LCD to return it's pointers to the top of the display. The Video Module asserts FRAME after all the lines of the LCD have been shifted and transferred, producing a full frame of display.

DF: **OUTPUT**

This pin is the AC signal for the LCD. Since LCD plasma tends to deteriorate whenever subjected to a DC voltage, the DF signal is used by the LCD to alternate the polarity of the row and column voltages used to turn the pixels on and off. The DF signal can be configured to toggle on every frame or can be configured to toggle every programmable number of LOAD signals.

LOAD: **OUTPUT**

This pin is the line synchronization pulse signal between the Video Module and the LCD, and is used by the LCD to transfer the contents of it's horizontal line shift register to the LCD panel for display. The Video Module asserts LOAD after an entire horizontal line of data has been shifted into the LCD.

CP: **OUTPUT**

This pin is the clock signal for the LCD. Data is pushed by the Video Module on the rising edge of CP and sampled by the LCD on the falling edge of CP.

VDAT(3:0): **OUTPUT**

These pins are the data for the LCD. These signals are directly connected to the LCD for 4-bit non-split displays. For 4-bit split and 8-bit non-split displays, an external register is required to demultiplex the 4-bit data into the desired 8 parallel data lines needed for the LCD.

DISPON: **OUTPUT**

This pin is the display-on enable signal for the LCD.

VIDDONE: **OUTPUT**

This pin is used to externally synchronize events to periods when the video is not shifting.

16.2 Interface Requirements

16.2.1 Display Types

The Video Module interface supports 3 types of LCD's: 4-bit split, 4-bit non-split, and 8-bit non split. Figure 16-1a shows these 3 different display types for monochrome displays and Figure 16-1b shows these 3 different display types for color displays.

A 4-bit split display uses 8 parallel data lines to shift data to both the upper and lower halves of the display simultaneously, with four bits of data shifted to the upper half and four bits of data shifted to the lower half. The end of frame is reached when each half of the display has been shifted and transferred. The Video Module within the PR31700 contains control logic which generates the appropriate video buffer addressing required for split displays (see Section 16.3.5). Since the PR31700 only provides 4 pins for video output data (VDAT[3:0]), an external 4-bit register is required to demultiplex the 4-bit data into the desired 8 parallel data lines (4 upper, 4 lower) needed for the LCD, as shown in Figure 16-3.

A 4-bit non-split display uses 4 parallel data lines to shift data to successive single horizontal lines of the display at a time, until the entire frame has been shifted and transferred. No external register is required for this configuration; the 4 pins for video output data from the PR31700 can be directly connected to the LCD.

An 8-bit non-split display uses 8 parallel data lines to shift data to successive single horizontal lines of the display at a time, until the entire frame has been shifted and transferred. This operates the same as the 4-bit non-split display, except that 8 parallel lines are shifted at a time instead of 4 parallel lines. As also required for the 4-bit split displays, an external 4-bit register is required to demultiplex the 4-bit video output data from the PR31700 into the desired 8 parallel data lines needed for the 8-bit non-split LCD, as shown in Figure 16-3.

Monochrome displays require 1 bit of video data per pixel. Figure 16-1a shows the pixel ordering of the parallel data lines for the 3 types of monochrome displays. Color displays require 3 bits (red, green, and blue) of video data per pixel, resulting in a horizontal shift register of length 3 times the number of pixels per horizontal line. This RGB-tri is shifted to the LCD as consecutive bits via the parallel data lines. Figure 16-1b shows the RGB-tri and pixel ordering of the parallel data lines for the 3 types of color displays.

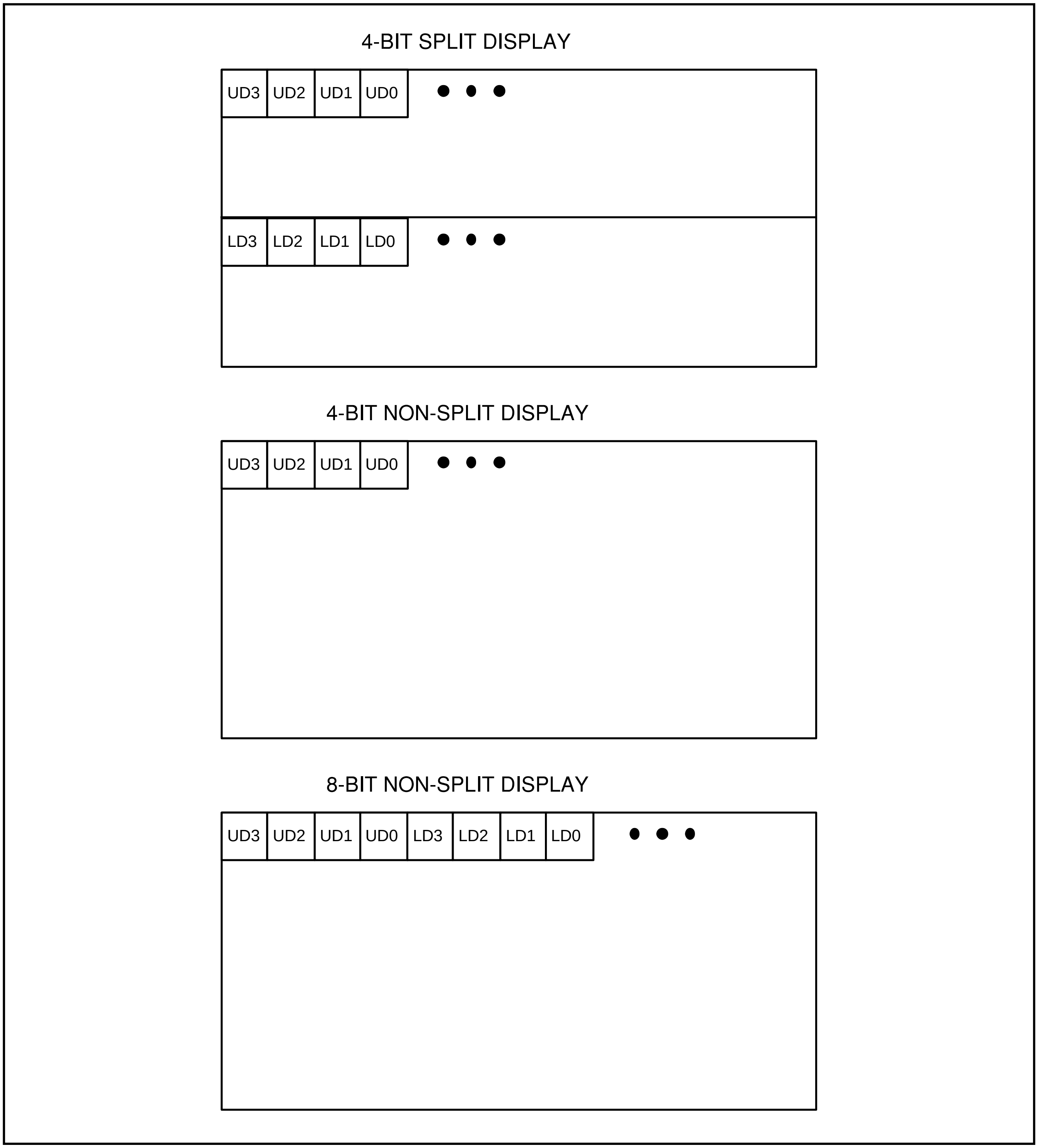
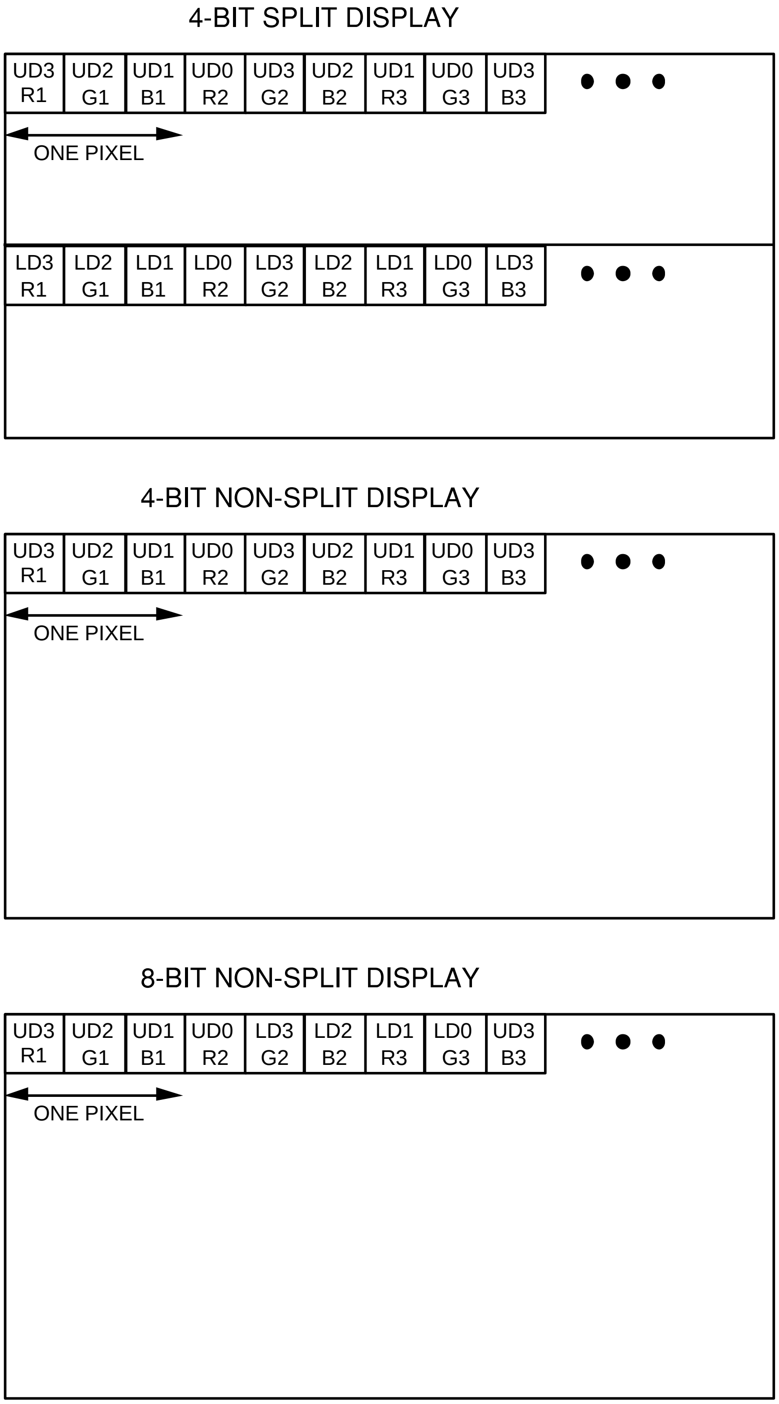


Figure 16-1a. Monochrome Display Types



4-BIT SPLIT DISPLAY

UD3 R1	UD2 G1	UD1 B1	UD0 R2	UD3 G2	UD2 B2	UD1 R3	UD0 G3	UD3 B3

LD3 R1	LD2 G1	LD1 B1	LD0 R2	LD3 G2	LD2 B2	LD1 R3	LD0 G3	LD3 B3

4-BIT NON-SPLIT DISPLAY

UD3 R1	UD2 G1	UD1 B1	UD0 R2	UD3 G2	UD2 B2	UD1 R3	UD0 G3	UD3 B3

8-BIT NON-SPLIT DISPLAY

UD3 R1	UD2 G1	UD1 B1	UD0 R2	LD3 G2	LD2 B2	LD1 R3	LD0 G3	UD3 B3

Figure 16-1b. Color Display Types

16.2.2 Timing Requirements

Data is shifted from the PR31700 to the LCD using the VDAT[3:0] signals. The CP signal is used to clock the data into the LCD's shift register. After each horizontal line of data has been shifted into the LCD's shift register, the LOAD signal is asserted to cause the line to be displayed on the LCD panel. Then after all of the lines have been displayed, the FRAME signal is asserted to cause the LCD's line pointer to start over at the top of the display. Figure 16-2 shows the various timing requirements for the LCD interface (the example shown is for 4-bit non-split displays).

The DF signal provides an AC reference for the display. Since LCD plasma tends to deteriorate whenever subjected to a DC voltage, the DF signal is used by the LCD to alternate the polarity of the row and column voltages used to turn the pixels on and off.

The DF signal can be configured to toggle on every frame (DFMODE = 0) or can beconfigured to toggle every programmable number of LOAD signals (DFMODE = 1),with this toggle period determined by the DFVAL(7:0) control register setting. Figure16-3 shows an example for DFMODE = 0 and for DFMODE = 1 with DFVAL(7:0) = $1.

The rate of the CP clock signal is controlled by the BAUDVAL(4:0) control registersetting. The number of CP clock pulses per line bursted between successive LOADpulses is determined by the number of pixels per line programmed into theHORZVAL(9:0) control register. The VIDRATE(9:0) control register setting determinesthe frequency of the LOAD pulses, which corresponds to the video line rate. This linerate, along with the number of LCD lines programmed into the LINEVAL(9:0) controlregister, then determines the video frame rate.

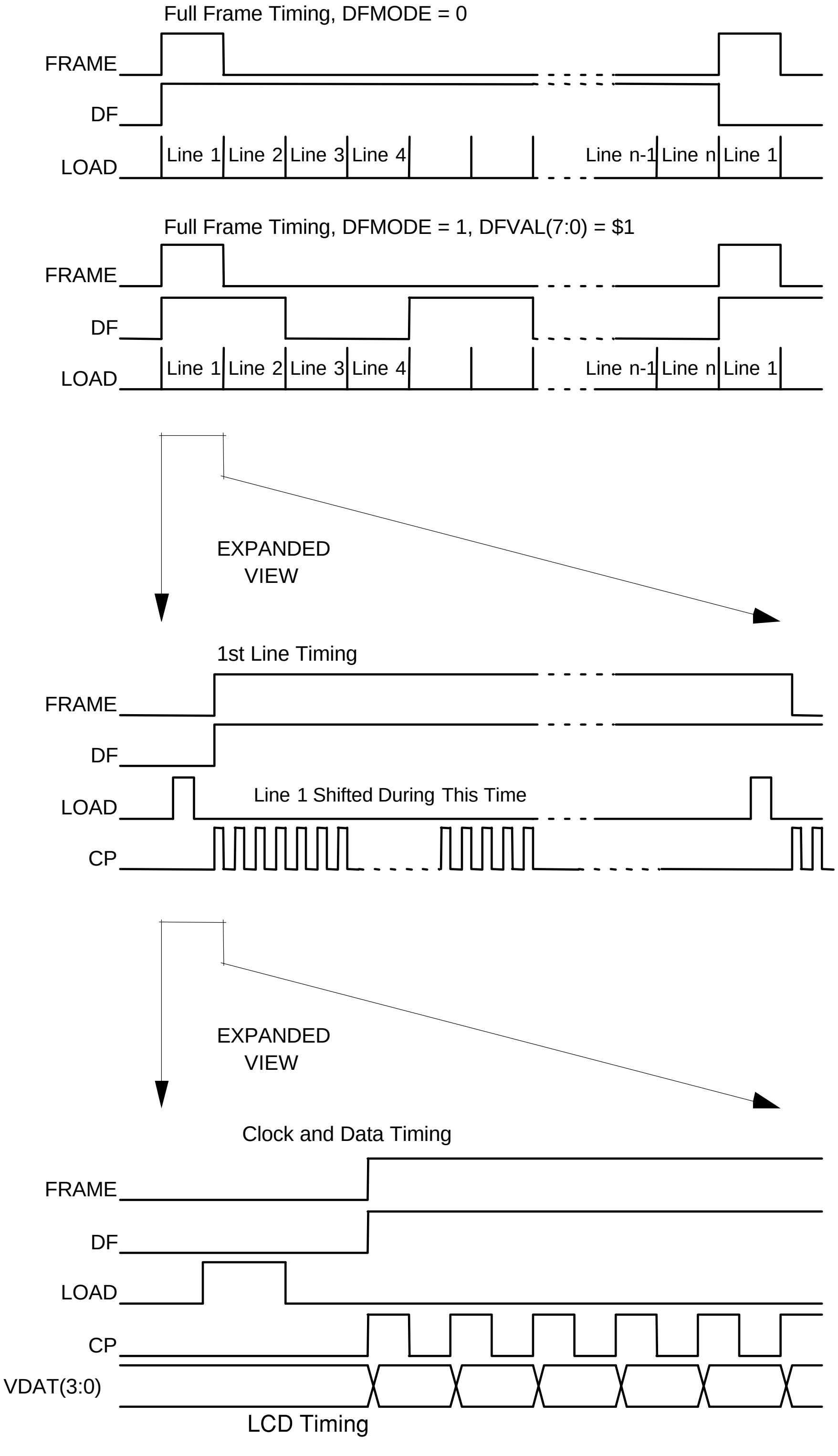


Figure 16-2. LCD Timing

As mentioned previously, since the PR31700 only provides 4 pins for video output data (VDAT[3:0]), an external 4-bit register is required to demultiplex the 4-bit data into the desired 8 parallel data lines needed for both 4-bit split and 8-bit non-split LCD's, as shown in Figure 16-3. To support these configurations, the Video Module drives the VDAT output data in a time-multiplexed format, alternating 4 bits of upper data (UD) with 4 bits of lower data (LD) on successive data cycles. The UD data is latched using the external register with CP used as the register's clock. Then, the LCD can clock in all 8 bits in parallel (UD plus LD, which are the VDAT signals directly connected to the LCD) using the falling edge of CP, as shown in Figure 16-4.

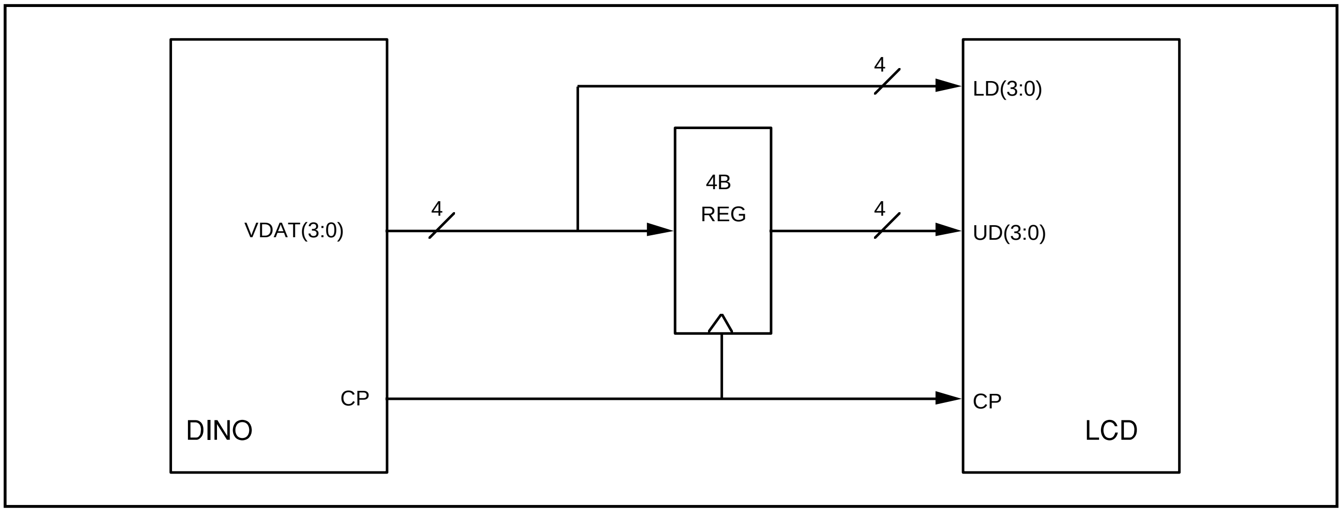


Figure 16-3. External Register Configuration for 4-Bit Split and 8-Displays

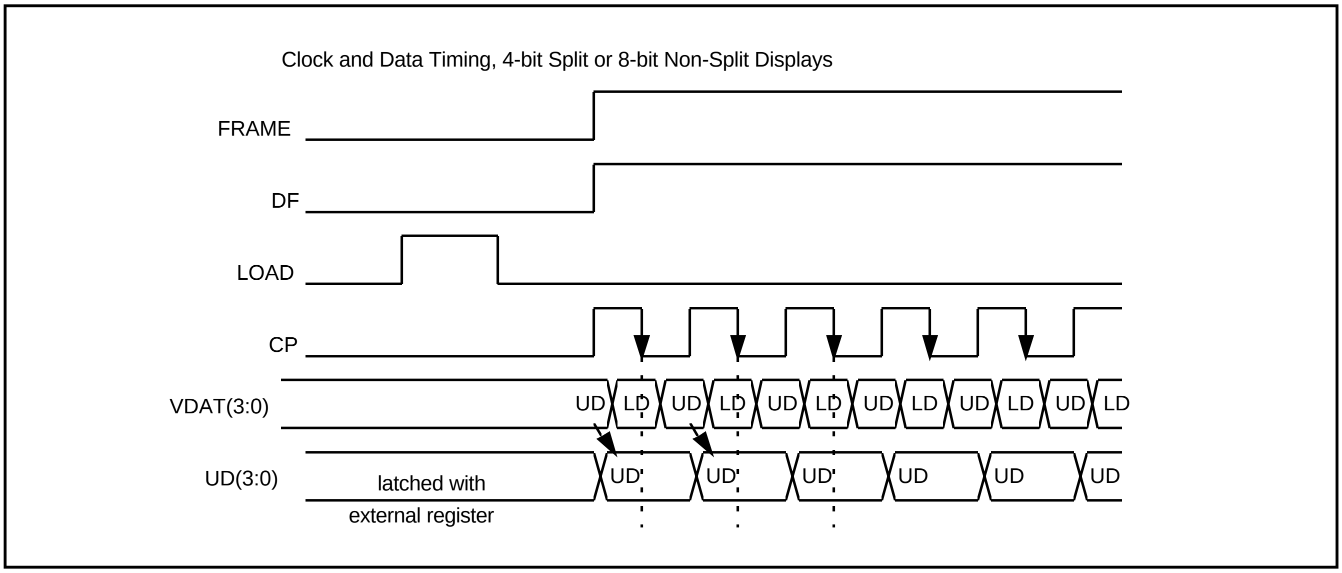


Figure 16-4. LCD Timing for 4-Bit Split and 8-Bit Non-Split

16.2.3 Synchronization

A new feature has been added to the Video Module that will provide a synchronization pulse external to the PR31700 that can be used to externally synchronize events to periods when the video is not shifting. Video is shifted immediately following a LOAD pulse and will shift until the proper number of bits are shifted for a given LCD line. Once the data finishes shifting, there will be a gap until next LOAD pulse, at which time the line is loaded into LCD and the next line begins shifting. There may be applications where it is desirable to synchronize events to this “dead” periods when video data is not shifting.

To facilitates this, a new signal has been added called VIDDONE. This signal replaces a previously unused PR31700 signal and pin called TESTOUT (pin 73 of PR31700).

Normally, the VIDDONE signal will be logic 0. After the video finishes shifting to the LCD, the VIDDONE signal will assert for a fixed ten “CLK” periods (where CLK is nominally 36.864 Mhz or period of 270ns). The time between the end of video shifting and assertion of the VIDDONE signal is programmable using the VIDDONE(6:0) control bits that have been added to bits (15:9) of the Video Control 1 Register.

Additionally, control bits FRAMEMASKVAL(3:0) have been added to bits (23:20) of the Video Control 4 Register. These bits define the number of lines for which the VIDDONE signal will not assert following the assertion of the FRAME signal. Use of these bits allow the VIDDONE synchronization pulses to be generated away from the periods after the video FRAME and DF transitions occur.

16.3 Implementation

16.3.1 Block Diagram

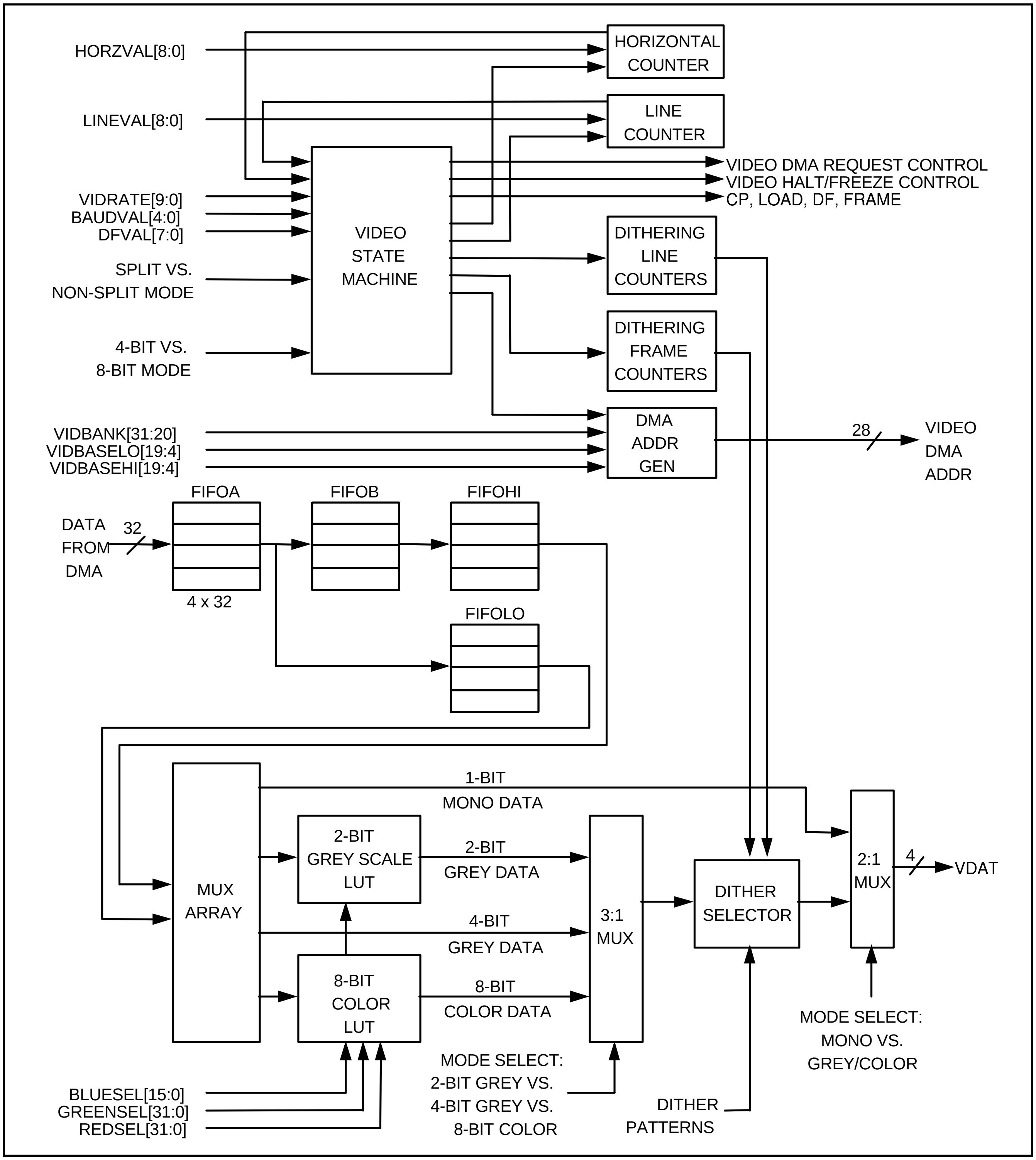


Figure 16-5. Video Module Block Diagram

16.3.2 Video State Machine

The Video State Machine contains programmable logic which allows the PR31700 to support the different interface timing and rates commonly found in different LCD's. This is implemented using several sets of programmable counters.

The CP clock rate is controlled by the Baud Rate Counter, which is preloaded with the BAUDVAL(4:0) control register setting. This counter generates the baud clock by dividing down from the main video clock (which is the same rate as CLK). The Baud Rate Counter is free-running, except whenever the HALT or FREEZEFRAME conditions are asserted. The HALT condition occurs if the video DMA requests cannot be serviced fast enough by the memory subsystem. In this case, HALT is asserted and the video baud clock is stopped, such that video data will momentarily be stopped from being sent to the LCD. The FREEZEFRAME condition is asserted in response to the assertion of the ENFREEZEFRAME control bit. This causes the video baud clock to stop and the video to stop shifting cleanly at the end of the current frame.

The actual CP clock signal is generated by gating the video baud clock with the valid shift period, as determined by the count output of the Horizontal Counter. The clock used to shift the video output data also depends on the display type (4-bit split, 4-bit non-split, or 8-bit non split), since the 4-bit split and 8-bit non split types require data to be shifted as twice the CP rate such that an external register can be used to demultiplex the 4-bit data into the desired 8 parallel data lines.

The video line rate and LOAD pulse frequency is controlled by the Video Rate Counter, which is preloaded with the VIDRATE(9:0) control register setting. This counter generates the LOAD pulse by counting the programmed number of video baud clocks. In addition, setting the LOADDLY control bit will result in an additional baud clock delay to be inserted for the FRAME and CP signals with respect to the LOAD signal, as shown in Figure 16-6. Some LCD's may require this timing restriction.

The toggle rate of the DF signal is controlled by the DF Counter, which is preloaded with the DFVAL(7:0) control register setting. The DF Counter will count each time a LOAD pulse is generated. If the DFMODE control bit is set to logic "0", the DF signal is configured to toggle on every frame. If the DFMODE control bit is set to logic "1", the DF signal is configured to toggle every time the DF Counter rolls over.

The FRAME pulse generation is controlled by both the Horizontal Counter and Line Counter (see Sections 16.3.3 and 16.3.4). Together, these counters determine the video frame rate. The FRAME pulse is asserted for a duration of the entire first line at a frequency of once per frame.

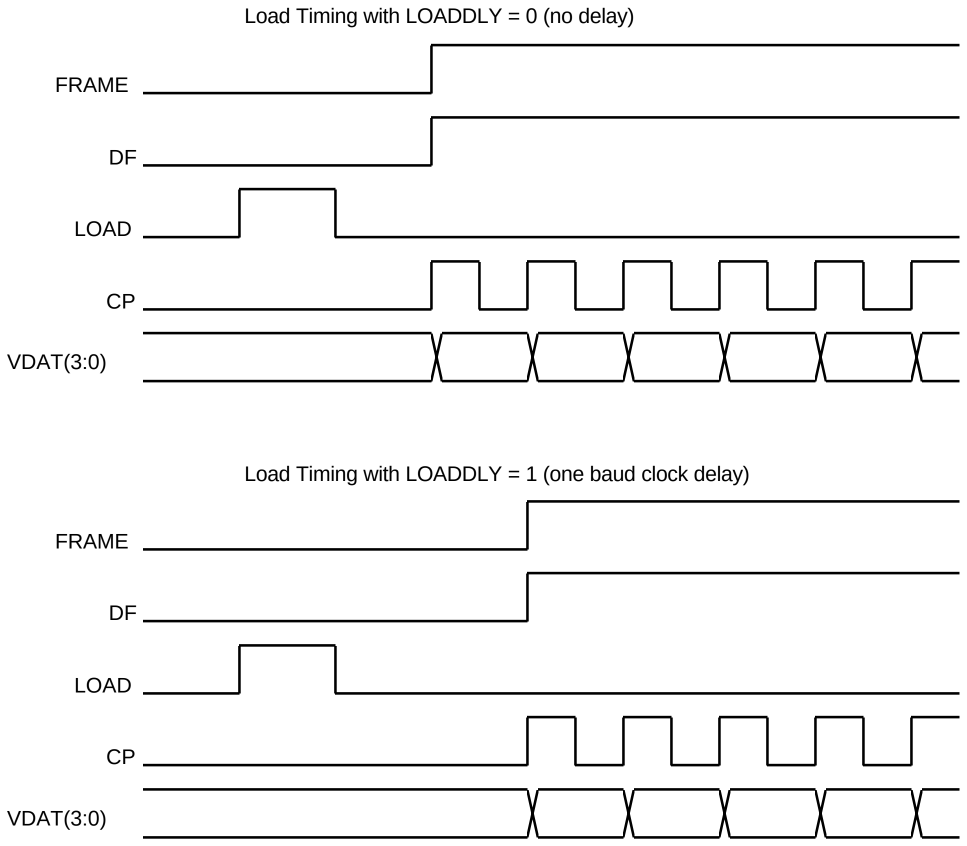

Load Timing with LOADDLY = 0 (no delay)
FRAME
DF
LOAD
CP
VDAT(3:0)
Load Timing with LOADDLY = 1 (one baud clock delay)
FRAME
DF
LOAD
CP
VDAT(3:0)

Figure 16-6. LOADDLY Timing

The Video State Machine also generates the video DMA request signals used to fetch the video image data from memory. Each data fetch sequence consists of one DMA request followed by 4 successive video DMA transactions, since video DMA data transfers are bursted as 4 longwords (32 bits each) at a time from system memory to the FIFO buffers within the Video Module. The video DMA requests are initiated whenever the FIFO buffers are available to accept more data (see Section 16.3.6).

16.3.3 Horizontal Counter

The Horizontal Counter is used to count the number of bits that are shifted (in a burst fashion) into the LCD's line shift register. For each new line, the Horizontal Counter is preloaded with the HORZVAL(8:0) control register setting and counts each time the 4 bits (or 8 bits for 8-bit non-split LCD's) of video output data are shifted out. The HORZVAL value should equal the horizontal size of the LCD divided by 4 (or 8 for 8-bit non-split LCD's) - 1. Thus, the 9-bit Horizontal Counter supports a maximum horizontal size of 2048 pixels.

Each time the Video Rate Counter reaches it's terminal count, the Video State Machine asserts the LOAD pulse in order to transfer the entire line shift register contents to the LCD panel. Thus, the Video Rate Counter preload value VIDRATE(9:0) should be configured with a value greater than the Horizontal Counter preload value HORZVAL(8:0) in order to load the entire line shift register contents to the LCD after all the bits have been shifted into the line shift register. Figure 16-2 shows an example of the relative timing for the LOAD pulse versus the burst of CP clocks used to shift a horizontal line of data.

16.3.4 Line Counter

The Line Counter is used to count the number of lines that are displayed by the LCD. Each time the Line Counter reaches it's terminal count, the Video State Machine asserts the FRAME pulse in order to begin a new frame and cause the pointers in the LCD to return to the top of the display. For each new frame, the Line Counter is preloaded with the LINEVAL(9:0) control register setting and counts each time a LOAD pulse is generated. The LINEVAL value should equal the number of lines for the LCD - 1. The 10-bit Line Counter supports a maximum vertical size of 1024 lines.

16.3.5 DMA Address Generation

The DMA Address Generation circuit consists of logic which generates the address for fetching video data from the video buffer located in system memory. The video buffer can be relocated anywhere in memory (over the full 32-bit address space) by configuring the VIDBANK(31:20), VIDBASEHI(19:4), and VIDBASELO(19:4) control register settings. See Figure 16-7 for a block diagram of the video DMA address generation circuit.

For a split LCD, the VIDBANK and VIDBASEHI bits are concatenated to provide the start address for the upper (HI) portion of the video buffer, while the VIDBANK and VIDBASELO bits are concatenated to provide the start address for the lower (LO) portion of the video buffer. For a non-split LCD, the VIDBANK and VIDBASEHI bits are concatenated to provide the start address for the video buffer.

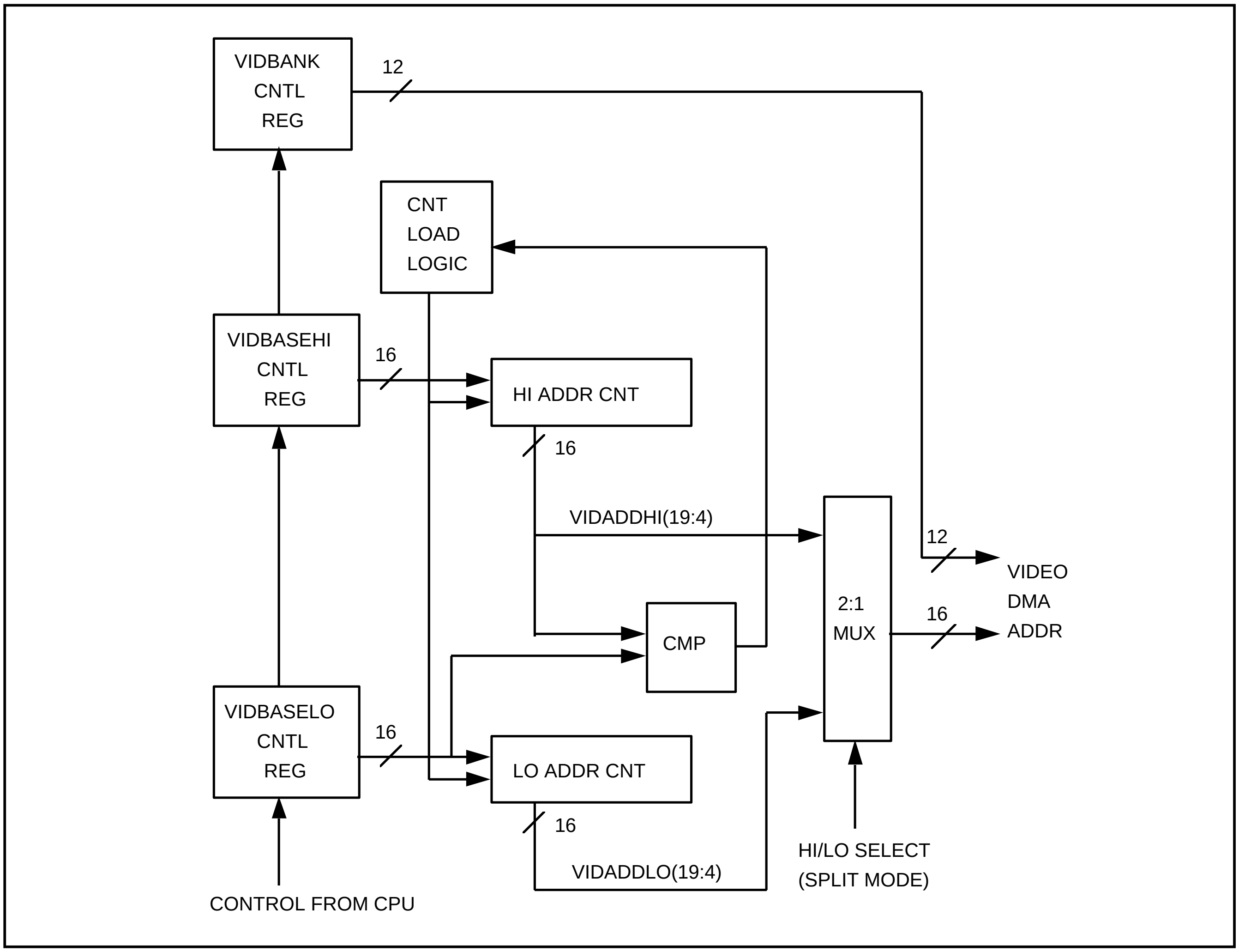


Figure 16-7. Video DMA Address Generation

The DMA Address Generation circuit consists of 2 sets of counters, the HI Address Counter and the LO Address Counter, which generate the address for the HI and LO portions of the video buffer, respectively. Both of these counters are incremented whenever the video FIFO buffer is filled from a burst of DMA-initiated data transfers. For a split LCD, the VIDBANK and VIDADDHI bits are concatenated to provide the address for the upper (HI) portion of the video buffer, while the VIDBANK and VIDADDLO bits are concatenated to provide the address for the lower (LO) portion of the video buffer. A 2:1 multiplexer is used to select between these two combined addresses, depending on whether the current video data fetch is for a HI or LO portion of the buffer. For a non-split LCD, the VIDBANK and VIDADDHI bits are concatenated to provide the address for the video buffer. Figure 16-8 shows the DMA address fields for split and non-split display formats. Video data is fetch as a burst of 4 longwords at a time and takes advantage of the fast-page mode available for DRAM or SDRAM.

SPLIT DISPLAY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIDBANK(31:20)												VIDADDHI(19:4)																0	0	0	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIDBANK(31:20)												VIDADDLO(19:4)																0	0	0	0

NON-SPLIT DISPLAY

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VIDBANK(31:20)												VIDADDHI(19:4)																0	0	0	0

Figure 16-8. Video DMA Address Fields

The VIDBANK control bits provide a bank location for the video buffer. The video bank can be located anywhere in memory over the full 32-bit address space, in increments of 1 MByte. These bits are latched at the beginning of each frame, making it possible to change these bits while the Video Module is active and have the video address switch to the new bank at the start of the next frame. In addition, the video buffer start address can be located anywhere within the 1 MByte bank, in increments of 16 bytes.

The HI and LO Address Counters are both up-counters. Thus, VIDBASEHI must be configured to a lower value than VIDBASELO. The HI and LO Address Counters are reloaded with the VIDBASEHI and VIDBASELO values, respectively, whenever the HI Address Counter output VIDADDHI equals the VIDBASELO value. Therefore, VIDBASELO value must always be set to one address greater than the end address of the upper (HI) portion of the display. For non-split displays, the VIDBASELO value must always be set to one address greater than the end address of the display.

16.3.6 Video FIFO

The Video Module contains several sets of First-In First-Out (FIFO) buffers to support video DMA burst data transfers. As shown in Figure 16-5, these FIFO's are organized as 4 sets of FIFO's. Each set of FIFO's is further subdivided into 4 latches (4 x 32-bits or a total of 128 bits wide).

The first set of FIFO's is referred to as FIFOA. Video data is loaded one longword (32 bits) at a time from the memory subsystem consecutively into each of the 4 latches of FIFOA. FIFOA then feeds both FIFOB and FIFOLO, while FIFOB feeds FIFOHI, with all of these data transfers performed 128 bits in parallel at a time. For example, the 1st latch of FIFOA directly loads into the 1st latch of FIFOB, at the same time the 2nd latch of FIFOA directly loads into the 2nd latch of FIFOB, etc. FIFOHI and FIFOLO are used to separately buffer the video data for the upper (HI) and lower (LO) portions of a split display, if this mode is enabled.

For a non-split display, FIFOLO is not used. In this mode, FIFOA accepts and buffers the 4-longword burst of data from the memory subsystem. Whenever FIFOA is full, and FIFOB is empty, the 4 longwords from FIFOA will then be transferred to FIFOB. Similarly, whenever FIFOA is full again and FIFOB is full, and FIFOHI is empty, the 4 longwords from FIFOB will then be transferred to FIFOHI. Then, the data is emptied from FIFOHI one longword at a time as the data is fetched for eventual shifting out to the LCD.

For a split display, both FIFOHI and FIFOLO are used. In this mode, FIFOA accepts and buffers the 4-longword burst of data from the memory subsystem. Whenever FIFOA is full, and FIFOB is empty, the 4 longwords from FIFOA will then be transferred to FIFOB. Then whenever FIFOA and FIFOB are both full, and FIFOHI and FIFOLO are both empty, the 4 longwords from FIFOB will then be transferred to FIFOHI at the same time the 4 longwords from FIFOA are transferred to FIFOLO. This is the mechanism used for separately buffering the video data for the upper (HI) and lower (LO) portions of the split display. As explained in the previous section, the video DMA Address Generation logic ensures that the correct HI or LO buffer address is generated at the correct time. Finally, the data is emptied from FIFOHI or FIFOLO one longword at a time via a multiplexer array, based on whether the current output cycle corresponds to an upper or lower portion of the split display.

16.3.7 Grey Scale LUT

The Video Module supports monochrome, 4-level or 16-level grey scale, and 8-bit per pixel color modes. The grey scale and color modes are implemented using a programmable lookup table (LUT) for selecting the shades of grey or color, followed by a programmable time-based dithering algorithm in the Dither Selector circuit. The monochrome mode bypasses these circuits and basically serializes the FIFOHI (and FIFOLO if a split display is used) output data into 4-bit streams for shifting to the LCD.

The grey scale modes of the Video Module are implemented by varying the duty cycle (i.e., number of frames) for which a given pixel is turned on, giving the monochrome display an appearance of grey scale. In order to reduce the noticeable flicker caused by turning on and off adjacent pixels at the same time, a time-based dithering algorithm is used.

Two grey scale modes are supported by the Video Module: 2-bit grey (4-level) or 4-bit grey (16-level). Since the LUT used in the Video Module consists of 16 total entries or grey scales, the 4-bit grey scale processing path does not require a LUT (all 16 grey scales are available and used in this mode). The 4-bit grey scale mode requires that the number of bits stored in the video buffer is equal to 4 times the number of pixels, with each pixel corresponding to 4 consecutive data bits stored in the video buffer. Each set of these 4 grey scale bits maps to one pixel of dithered data generated by the Dither Selector.

The 2-bit grey mode does use a LUT, which allows any 4 grey levels to be selected out of the 16 total possible grey levels. The 2-bit grey scale mode requires that the number of bits stored in the video buffer is equal to 2 times the number of pixels, with each pixel corresponding to 2 consecutive data bits stored in the video buffer. The 2-bit grey scale LUT uses the BLUESEL(15:0) control bits as the programmable LUT entries. The 16-bit BLUESEL field is divided into 4 nibbles: BLUESEL(15:12), BLUESEL(11:8), BLUESEL(7:4), and BLUESEL(3:0). Each nibble then corresponds to one of the 4 grey scales selected out of 16 possible, with each 2 pre-grey-scale bits from memory used to lookup one of these 4 grey scales at a time (2 to 4 mapping). Figure 16-9 shows a block diagram of the 2-bit grey scale LUT logic. Each set of the 4 grey scale bits from the LUT output maps to one pixel of dithered data generated by the Dither Selector.

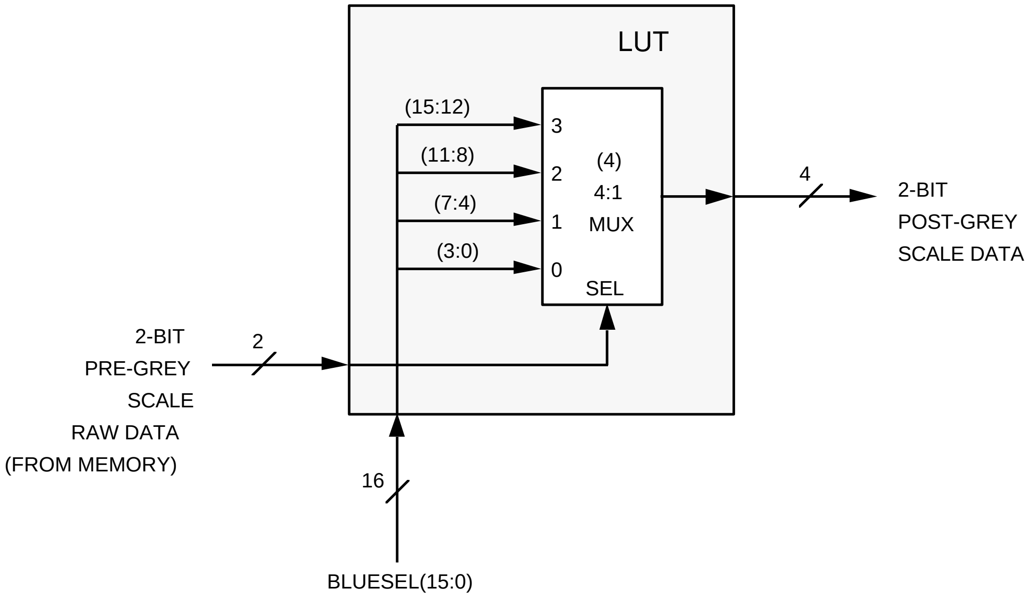


Figure 16-9. 2-Bit Grey Scale LUT

16.3.8 Color LUT

The Video Module supports an 8-bit per pixel color mode for interfacing to color LCD's. The color mode generates 16 different shades using the time-based dithering algorithm. The 8-bits used per pixel are encoded into 3 bits of red (R), 3 bits of green (G), and 2 bits of blue (B). The color mode requires that 8 bits (3:3:2 for R:G:B) are stored in the video buffer for every RGB-tri (one pixel) of dithered data shifted to the color LCD.

The color mode uses separate LUT's for red, green, and blue. The red LUT uses the REDSEL(31:0) control bits as the programmable LUT entries. The 32-bit REDSEL field is divided into 8 nibbles: REDSEL (31:28), REDSEL (27:24), ... , REDSEL (7:4), and REDSEL (3:0). Each nibble then corresponds to one of the 8 shades selected out of 16 possible, with each 3 pre-red bits from memory used to lookup one of these 4 shades at a time (3 to 4 mapping). The green LUT is identical to the red LUT, except the LUT uses the GREENSEL(31:0) control bits as the programmable LUT entries. The blue LUT uses the BLUESEL(15:0) control bits as the programmable LUT entries, and each LUT output nibble corresponds to one of the 4 shades selected out of 16 possible, with each 2 pre-blue bits from memory used to lookup one of these 4 shades at a time (2 to 4 mapping). Figure 16-10 shows a block diagram of the 8-bit per pixel color LUT logic.

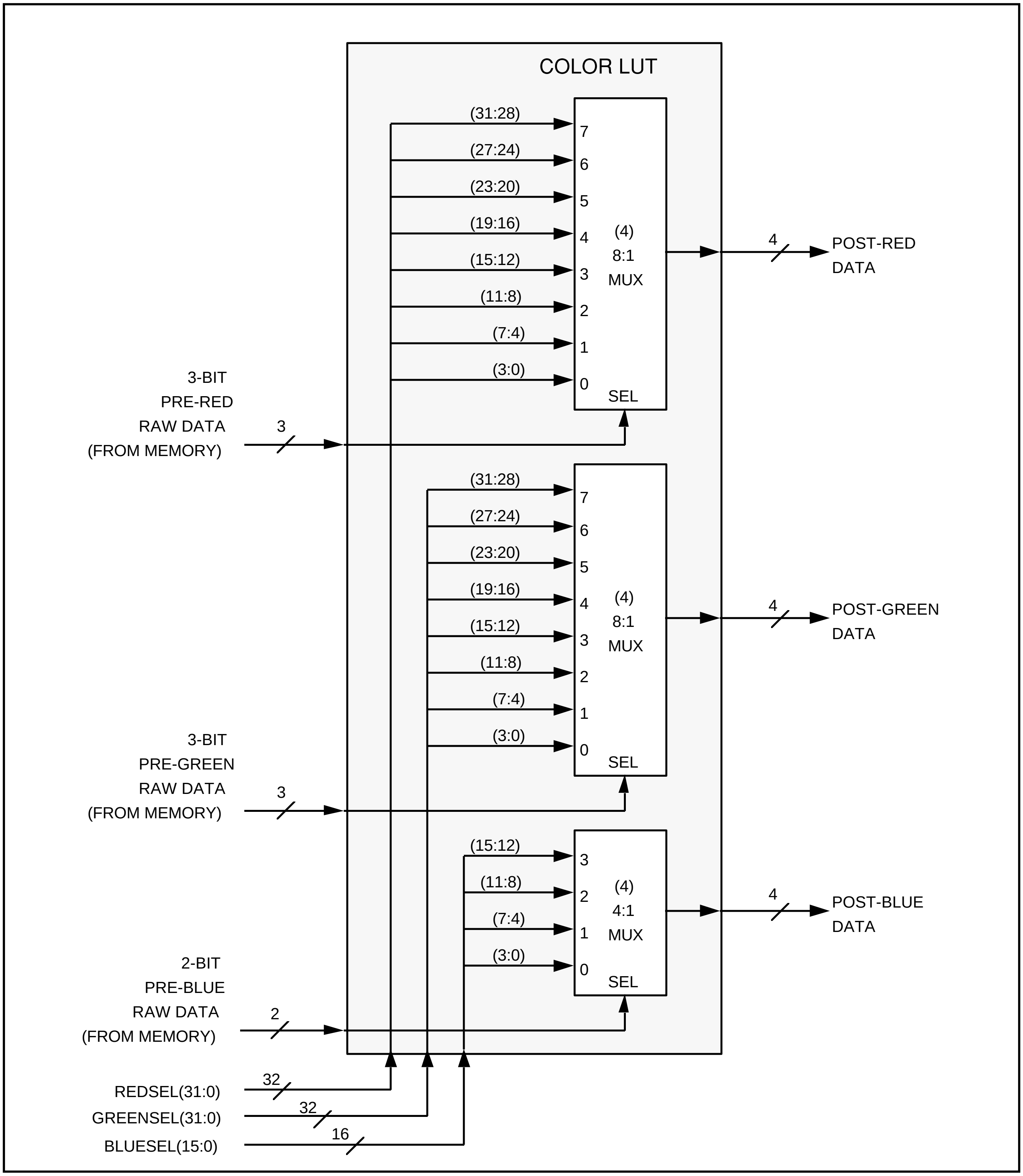


Figure 16-10. 8-Bit Per Pixel Color LUT

16.3.9 Dithering

The Dither Selector is based on a scheme which varies the duty cycle (i.e., number of frames) for which a given pixel is turned on, giving the display an appearance of multiple shades. In order to reduce the noticeable flicker caused by turning on and off adjacent pixels at the same time, a time-based dithering algorithm is used to vary the pattern of adjacent pixels every frame. A block diagram of the Dither Selector is shown in Figure 16-11. The Dither Selector contains a set of Dithering Line Counters and a set of Dithering Frame Counters. The set of Dithering Line Counters contains 4 counters which count modulo-3, modulo-4, modulo-5, and modulo-7 every horizontal line. The set of Dithering Frame Counters contains 4 counters which count modulo-3, modulo-4, modulo-5, and modulo-7 each video frame. The output of the Dithering Line Counters are used to lookup the programmable dither patterns configured in Video Control Registers 8 through 14, generating a 16-bit Enable Pattern. Each new line count looks up a different field of the appropriate Video Control Register, resulting in a time-varying and dithered pattern of on and off pixel control with the desired duty cycle. Section 16.4 lists recommended values for each of these dither patterns. The pre-dithered video data is processed by sending 16 adjacent bits at a time through 4 identical Enable Select circuits, each of which processes 4 bits of pre-dithered data. Each 4 bits of pre-dithered data is used to lookup one of the 16 possible enable patterns, which thus determine whether or not a given pixel is turned on or off for the given frame and line. Thus every 4 bits of pre-dithered data produces 1 bit of dithered data. Table 16-1 shows the duty cycle ratios used by the Enable Select circuits, which correspond to the 16 possible shades (of grey or color) available.

Table 16-1. Dither Duty Cycles

pre-dithered duty cycle	duty cycle
15	1
14	6/7
13	4/5
12	3/4
11	5/7
10	2/3
9	3/5
8	4/7
7	2/4
6	3/7
5	2/5
4	1/3
3	2/7
2	1/5
1	1/7
0	0

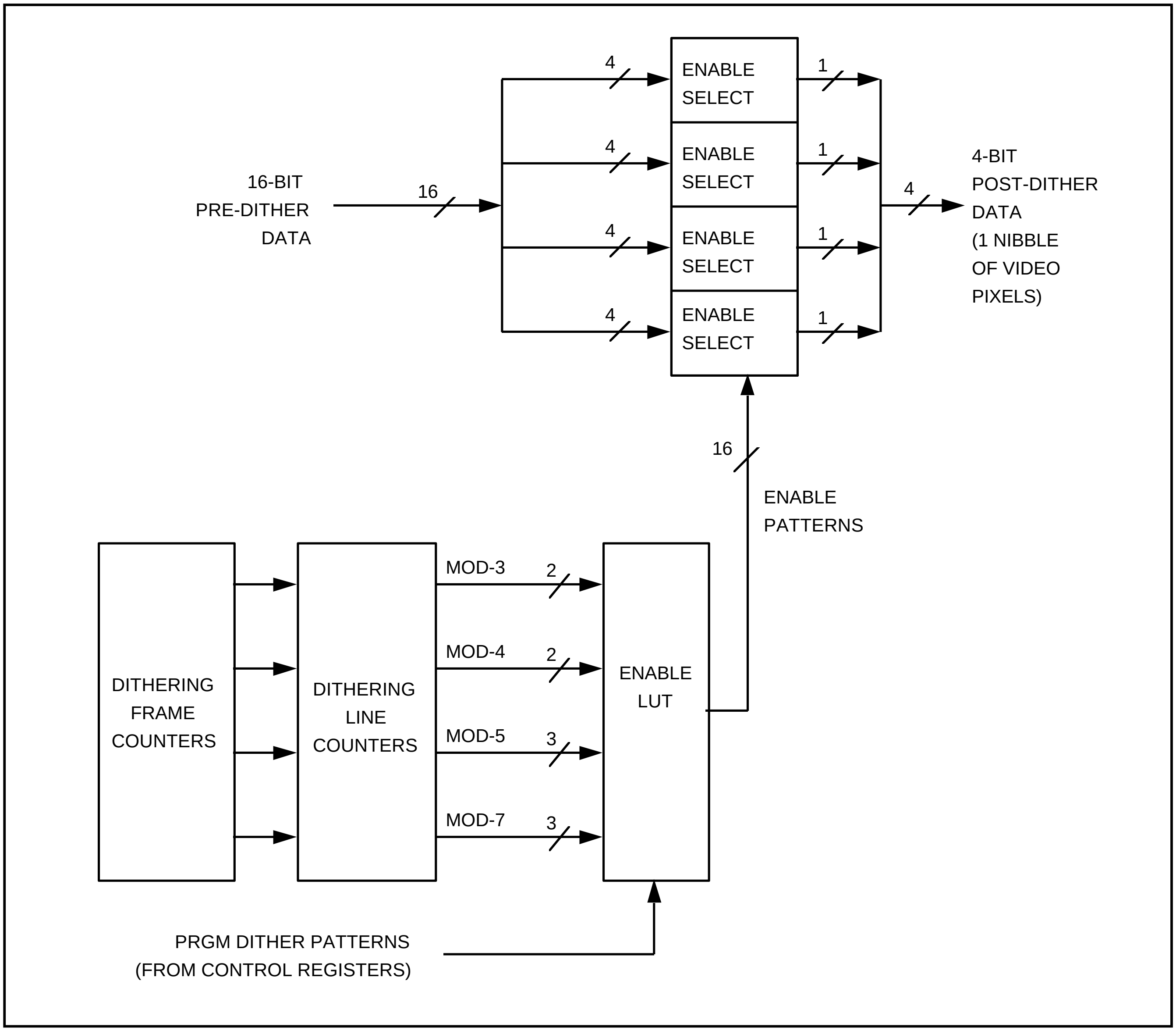


Figure 16-11. Dither Selector Block Diagram

16.3.10 Related Interrupts

LCDINT:
Issues an interrupt at the end of each video frame.

DFINT:
Issues an interrupt each time the video DF signal toggles.

16.4 Video Registers

16.4.1 Video Control 1 Register

OFFSET = $028:

Bit	Label	RESET	Read/Write
31-22	LINECNT(9:0)	0	R
21	LOADDLY	0	R/W
20-16	BAUDVAL(4:0)	X	R/W
15-9	VIDDONEVAL(6:0)	X	R/W
8	ENFREEZEFRAME	0	R/W
7-6	BITSEL(1:0)	0	R/W
5	DISPSPLIT	0	R/W
4	DISP8	0	R/W
3	DFMODE	0	R/W
2	INVVID	0	R/W
1	DISPON	0	R/W
0	ENVID	0	R/W

LINECNT(9:0): **read-only**

These bits provide the status of the line counter.

LOADDLY:

Setting this bit will cause the FRAME and CP signals to be delayed by an additional baud clock relative to the LOAD signal. This is only required by a few LCDs that have a long specification time between LOAD and CP.

BAUDVAL(4:0):

These bits determine the rate of the CP clock. These bits must be set to a minimum of $1 and should be set to as high a value as possible to interface to the selected LCD. The following equation provides clock rate:

$$\text{CP Rate} = \frac{36.864\text{ MHz}}{\text{BAUDVAL(4:0)} * 2 + 2}$$

VIDDONEVAL(6:0):

These bits control the time between the end of video shifting and the assertion of the VIDDONE signal. These bits must be set to at least a value of $1. Setting the VIDDONEVAL(6:0) bits to a value of $1 will cause a delay from the video finishing shifting to the assertion of VIDDONE of 5 CLK periods (normally 135ns). Each increment to the VIDDONEVAL(6:0) bits will provide an additional 16 CLK periods (normally 434 ns). Care must taken to avoid setting the values too large such that the next LOAD pulse will occur before the VIDDONE signal completes. These control bits not affected by a Power On Reset.

ENFREEZEFRAME:

Setting this bit will cause the Video logic to freeze at the end of the current frame. This feature is used to blip a single frame. The video logic will not start transferring data again until this bit is cleared and the ENVID bit is toggled.

BITSEL(1:0):

These bits define the bit depth for the Video Module to generate according to the following:

$3	8-bit per pixel Color Mode
$2	4-bit per pixel Grey Scale Mode
$1	2-bit per pixel Grey Scale Mode
$0	Monochrome Mode

DISPSPLIT:

This bit should be set if interfacing to a split LCD.

DISP8:

This bit should be set if interfacing to an 8-bit non-split LCD.

DFMODE:

If this bit is a logic "0" then the DF signal will be toggled on each frame. If this bit is a logic "1" then the DF signal will toggle at the rate defined by the DFVAL(7:0) bits.

INVVID:
Setting this bit will cause the VDAT(3:0) signals to be inverted.

DISPON:
This bit is directly connected to the DISPON signal pin.

ENVID:
Setting this bit will enable the video logic. This bit should not be set until all other control bits are setup.

16.4.2 Video Control 2 Register

OFFSET = $02C: **write-only**

Bit	Label	RESET	Read/Write
31-22	VIDRATE(9:0)	X	W
21	Reserved		
20-12	HORZVAL(8:0)	X	W
15-9	Reserved		
9-0	LINEVAL(9:0)	X	W

VIDRATE(9:0): **write-only**
These bits determine the frequency at which the LOAD pulse is generated, which in turn sets the Frame Rate. The Line Rate and Frame Rate are given by the following equations:

HORZVAL(8:0): **write-only**
These bits define the horizontal size of the LCD according to the following equation:

HORZVAL = (HorzSize 4 - 1) for 4-bit split or non-split LCD
HORZVAL = (HorzSize 8 - 1) for 8-bit non-split LCD

LINEVAL(9:0): **write-only**
These bits define the number of lines for the LCD.

LINEVAL = (#of Lines - 1) for a non-split LCD
LINEVAL = (#of Lines 2 - 1) for a split LCD

16.4.3 Video Control 3 Register

OFFSET = $030: **write-only**

Bit	Label	RESET	Read/Write
31-20	VIDBANK(31:20)	X	W
19-4	VIDBASEHI(19:4)	X	W
3-0	Reserved		

VIDBANK(31:20): **write-only**
These bits provide the upper bits of the address from which video data is fetched from memory. These bits are concatenated with the upper and lower address counters to provide the actual address. These bits are latched at the beginning of each frame, thus it is possible to change these bits "on the fly" and the address will not change until the beginning of the next frame.

VIDBASEHI(19:4): **write-only**
These bits provide the start address for the upper address counter.

16.4.4 Video Control 4 Register

OFFSET = $034: **write-only**

Bit	Label	RESET	Read/Write
31-24	DFVAL(7:0)	X	W
23-20	FRAMEMASKVAL(3:0)	X	W
19-4	VIDBASELO(19:4)	X	W
3-0	Reserved		

DFVAL(7:0): **write-only**

These bits define the rate at which the DF signal will toggle if the DFMODE bit is set. The DF counter counts on each LOAD pulse, thus the DF Rate is given by the following equation:

$$\text{DF Rate} = \frac{\text{Line Rate}}{\text{DFVAL(7:0)} + 1}$$

FRAMEMASKVAL(3:0): **write-only**

These bits define the number of lines for which the VIDDONE signal will not assert following the assertion of the FRAME signal. If these bits are set to $0 then the VIDDONE signal will assert after every line completes shifting. Setting these bits to $1 will cause the VIDDONE signal to not assert after the line that immediately follow the assertion of the FRAME signal. Increases in the FRAMEMASKVAL(3:0) will cause subsequent lines to not assert the VIDDONE signal. These controls bits are not affected by a Power On Reset.

VIDBASELO(19:4): **write-only**

These bits provide the start address for the lower address counter. If a non-split LCD is used, these bits must be set to "1" plus the last address of the video buffer.

16.4.5 Video Control 5 Register

OFFSET = $038: **write-only**

Bit	Label	RESET	Read/Write
31-0	REDSEL(31:0)	X	W

REDSEL(31:0): **write-only**

The Video Module logic generates 16 different shades using a time-based dithering algorithm. The 8-bit per pixel color mode encodes 3 bits of Red, 3 bits of Green, and 2 bits of Blue. These bits define which of the sixteen shades each of the 8 possible red combinations will choose. In other words, 3 bits are expanded into 4 bits using these values as the lookup table.

111	REDSEL(31:28)
110	REDSEL(27:24)
101	REDSEL(23:20)
100	REDSEL(19:16)
011	REDSEL(15:12)
010	REDSEL(11:8)
001	REDSEL(7:4)
000	REDSEL(3:0)

16.4.6 Video Control 6 Register

OFFSET = $03C: **write-only**

Bit	Label	RESET	Read/Write
31-0	GREENSEL(31:0)	X	W

GREENSEL(31:0): **write-only**

The Video Module logic generates 16 different shades using a time-based dithering algorithm. The 8-bit per pixel color mode encodes 3 bits of Red, 3 bits of Green, and 2 bits of Blue. These bits define which of the sixteen shades each of the 8 possible green combinations will choose. In other words, 3 bits are expanded into 4 bits using these values as the lookup table.

111	GREENSEL(31:28)
110	GREENSEL(27:24)
101	GREENSEL(23:20)
100	GREENSEL(19:16)
011	GREENSEL(15:12)
010	GREENSEL(11:8)
001	GREENSEL(7:4)
000	GREENSEL(3:0)

16.4.7 Video Control 7 Register

OFFSET = $040: **write-only**

Bit	Label	RESET	Read/Write
31-16	Reserved		
15-0	BLUESEL(15:0)	X	W

BLUESEL(15:0): **write-only**

The Video Module logic generates 16 different shades using a time-based dithering algorithm. The 8-bit per pixel color mode encodes 3 bits of Red, 3 bits of Green, and 2 bits of Blue. These bits define which of the sixteen shades each of the 4 possible blue combinations will choose. In other words, 2 bits are expanded into 4 bits using these values as the lookup table. The BLUESEL values are also used as the LUT for the 2-bit grey scale mode.

11	BLUESEL(15:12)
10	BLUESEL(11:8)
01	BLUESEL(7:4)
00	BLUESEL(3:0)

16.4.8 Video Control 8 Register

OFFSET = $044: **write-only**

Bit	Label	RESET	Read/Write
31-12	Reserved		
11-0	PAT2_3(11:0)	X	W

PAT2_3(11:0): **write-only**

These bits define the (2 out of 3) patterning for the dithering algorithm. The (1 out of 3) patterning for the dithering algorithm is the inverse of the (2 out of 3) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT2_3(11:8)	0111
1	PAT2_3(7:4)	1101
2	PAT2_3(3:0)	1010

16.4.9 Video Control 9 Register

OFFSET = $048: **write-only**

Bit	Label	RESET	Read/Write
31-16	PAT3_4(15:0)	X	W
15-0	PAT2_4(15:0)	X	W

PAT3_4(15:0): **write-only**

These bits define the (3 out of 4) patterning for the dithering algorithm.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT3_4(15:12)	0111
1	PAT3_4(11:8)	1101
2	PAT3_4(7:4)	1011
3	PAT3_4(3:0)	1110

PAT2_4(15:0): **write-only**

These bits define the (2 out of 4) patterning for the dithering algorithm.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT2_4(15:12)	1010
1	PAT2_4(11:8)	0101
2	PAT2_4(7:4)	1010
3	PAT2_4(3:0)	0101

16.4.10 Video Control 10 Register

OFFSET = $04C: **write-only**

Bit	Label	RESET	Read/Write
31-20	Reserved		
19-0	PAT4_5(19:0)	X	W

PAT4_5(19:0): **write-only**

These bits define the (4 out of 5) patterning for the dithering algorithm. The (1 out of 5) patterning for the dithering algorithm is the inverse of the (4 out of 5) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT4_5(19:16)	0111
1	PAT4_5(15:12)	1101
2	PAT4_5(11:8)	1111
3	PAT4_5(7:4)	1011
4	PAT4_5(3:0)	1110

16.4.11 Video Control 11 Register

OFFSET = $050: **write-only**

Bit	Label	RESET	Read/Write
31-20	Reserved		
19-0	PAT3_5(19:0)	X	W

PAT3_5(19:0): **write-only**

These bits define the (3 out of 5) patterning for the dithering algorithm. The (2 out of 5) patterning for the dithering algorithm is the inverse of the (3 out of 5) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT3_5(19:16)	0111
1	PAT3_5(15:12)	1010
2	PAT3_5(11:8)	0101
3	PAT3_5(7:4)	1010
4	PAT3_5(3:0)	1101

16.4.12 Video Control 12 Register

OFFSET = $054: write-only

Bit	Label	RESET	Read/Write
31-28	Reserved		
27-0	PAT6_7(27:0)	X	W

PAT6_7(27:0): write-only

These bits define the (6 out of 7) patterning for the dithering algorithm. The (1 out of 7) patterning for the dithering algorithm is the inverse of the (6 out of 7) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT6_7(27:24)	1111
1	PAT6_7(23:20)	1011
2	PAT6_7(19:16)	1111
3	PAT6_7(15:12)	1101
4	PAT6_7(11:8)	1111
5	PAT6_7(7:4)	1110
6	PAT6_7(3:0)	0111

16.4.13 Video Control 13 Register

OFFSET = $058: write-only

Bit	Label	RESET	Read/Write
31-28	Reserved		
27-0	PAT5_7(27:0)	X	W

PAT5_7(27:0): write-only

These bits define the (5 out of 7) patterning for the dithering algorithm. The (2 out of 7) patterning for the dithering algorithm is the inverse of the (5 out of 7) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT5_7(27:24)	0111
1	PAT5_7(23:20)	1011
2	PAT5_7(19:16)	0101
3	PAT5_7(15:12)	1010
4	PAT5_7(11:8)	1101
5	PAT5_7(7:4)	1110
6	PAT5_7(3:0)	1111

16.4.14 Video Control 14 Register

OFFSET = $05C: write-only

Bit	Label	RESET	Read/Write
31-28	Reserved		
27-0	PAT4_7(27:0)	X	W

PAT4_7(27:0): write-only

These bits define the (4 out of 7) patterning for the dithering algorithm. The (3 out of 7) patterning for the dithering algorithm is the inverse of the (4 out of 7) patterning.

LINE	PATTERN FIELD	RECOMMENDED PATTERN
0	PAT4_7(27:24)	1011
1	PAT4_7(23:20)	1001
2	PAT4_7(19:16)	1101
3	PAT4_7(15:12)	1100
4	PAT4_7(11:8)	0110
5	PAT4_7(7:4)	0110
6	PAT4_7(3:0)	0011

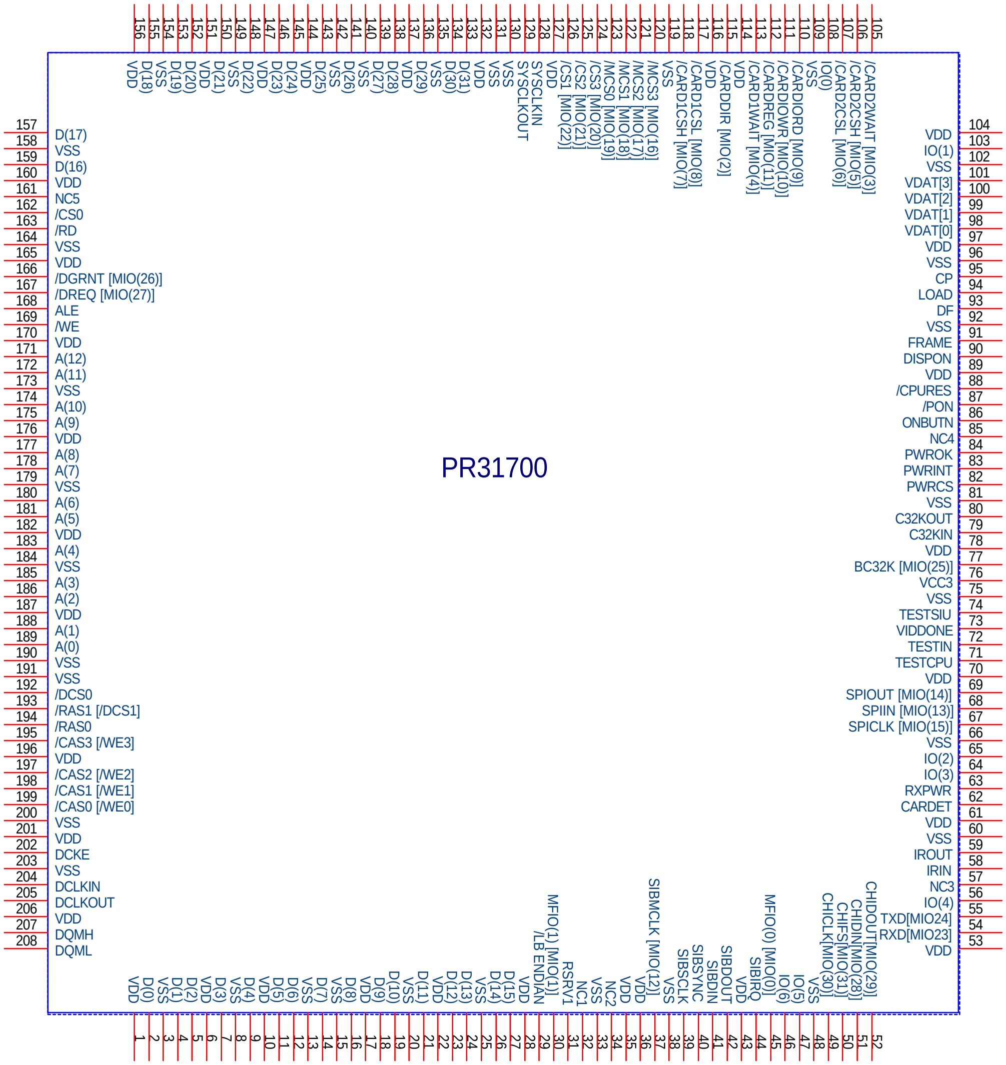
Chapter 17
Package Information

17.1 Package Information

This section contains pin assignments and package information for the PR31700 Processor.

17.1.1 Pin Assignment

17.1.1.1 208-Pin Plastic Quad Flat Pack



Chapter 17
Package Information

Chapter 18
Appendix

This section contains the following:

I. Changes and differences between PR31500 and PR31700.

II. Updates from ver. 0.1 to ver. 0.2

I. The changes and differences between PR31500 and PR31700.
##### PR3910 Core #####

1. Core Clock

   R3900 core inside PR31700 can run at 75MHz while current PR31500 can run at 37MHz.

2. Kseg2 TLB mapping

   The address from 0xFFFF_FFFF to 0xFFFF_0000 in Kseg2 area is TLB mapped.

Segment	VirtualAddress	PhysicalAddress	Cacheable
Kseg2	0xFFFF_FFFF 0xFFFF_0000	TLB Mapped	Yes
Reserved	0xFFFE_FFFF 0xFF00_0000	0xFFFE_FFFF 0xFF00_0000	No
Kseg2	0xFEFF_FFFF 0x0000_0000	TLB Mapped	Yes
Kseg1	0xBFFF_FFFF 0xA000_0000	0x1FFF_FFFF 0x0000_0000	No
Kseg0	0x9FFF_FFFF 0x8000_0000	0x1FFF_FFFF 0x0000_0000	Yes
Kuseg	0x7FFF_FFFF 0xC000_0000	TLB Mapped	Yes

##### Peripheracy #####

1. PCMCIA

   a. Content of modification

      * Support for 8-bit port access

      * Timing change for write access in memory and attribute space (if possible)

   b. Modified specification

	/CARDxCSH	/CARDxCSL	HA (0)	CD (15:8)	CD (7:0)
Standby	1	1	X	High-Z	High-Z
Byte	1	0	0	High-Z	Even-Byte
Byte (8bit)	1	0	1	High-Z	Odd-Byte
Byte (16bit)	0	1	X	Odd-Byte	High-Z
Word	0	0	X	Odd-Byte	Even-Byte

c. Additional bit of “Dino” Register

Memory Configuration 3 Register (offset = $00C)

Bit	Label	Reset	Read/Write
3	8PORTSEL	0	R/W

SPORTSEL:

This bit defines the PCMCIA port size.

0 : 16 bit port access

1 : 8 bit port access

2. Enhanced Doze mode

a. Content of modification

* VIDCLK can be divided independent of other internal clocks

* PR31700 internal clocks can be divided up to by 16 according to RF bits and slowbus

* “Better” Div mode

b. Specification

RF [1:0]	SLOWBUS	CORECLK	DCLKOUT	CLK2X	FREECLK	CLK	XHFREE
00	0	F	F	F	F	F/2	F/2
01	0	F/2	F/2	F/2	F	F/4	F/2
10	0	F/4	F/4	F/4	F	F/8	F/2
11	0	F/8	F/8	F/8	F	F/16	F/2
00	1	F/2	F/2	F/2	F	F/4	F/2
01	1	F/4	F/4	F/4	F	F/8	F/2
10	1	F/8	F/8	F/8	F	F/16	F/2
11	1	F/16	F/16	F/16	F	F/32	F/2

VIDRF[1:0]	VIDCLK
00	F/2
01	F/4
10	F/8
11	F/16

<Comment>

VIDCLK is come from XHFREE

* “Better” Div mode

In this mode all clocks in “ASTRO” module is independent of RF bits.

3. Additional bit of “Dino” register

Power Control Register (OFFSET = $1C4)

Bit	Label	Reset	Read/Write
28-27	VIDRF [1:0]	00	R/W
26	Slowbus	0	R/W
25	DIVMOD	0	R/W

VIDRF [1:0] :
Please refer to above table regarding VIDRF [1:0].

SLOWBUS :
Please refer to above table regarding SLOWBUS

DIVMOD :
Setting this bit will be cause all clocks in “ASTRO” module to be independent of RF bits.

4. Support for Lager DRAMS

a. Content of modification

* Support for up to 32MB DRAM

b. Modified Specification

ROWA(12:0)	ROWSEL	COLA(12:0)	COLSEL
18, 17:9	00	22,20,18,8:1	0000
22, 18, 20, 19, 17:9	01	19,18,8:1	0001
20, 22, 21, 19, 17:9	10<-	21,20,18,8:1	0010
22, 23, 21, 19, 17:9	11<-	23,22,20,18,8:1	0011
		24, 22, 20, 18, 8:1	0100
		24, 23, 21, 8:1	1001 <-

5. Fixed CPU reset bug

a. Content of bug

* There is a bug in the PR31700 that will cause the DMA arbitration logic to hang if the /CPURES button is pressed simultaneous with an internal DMA request and CPU memory access.

II. Updates PR31700 user manual version 0.1 to 0.2

1. On page 76, Figure 6-1 : Clock Module Block Diagram shows drawing errors on the internal clock distribution scheme. (See the block diagram below for correction)

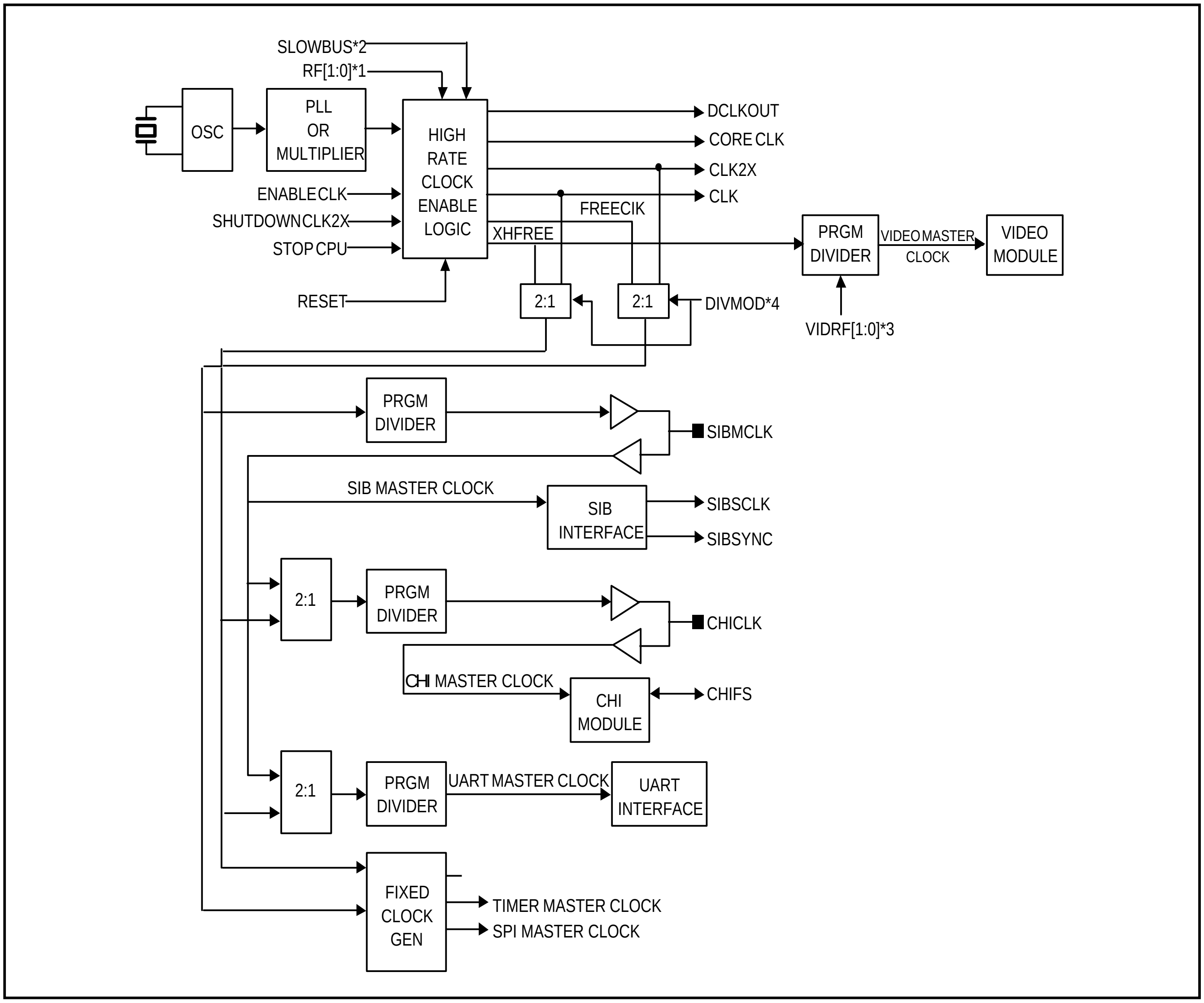


Figure 6-1. Clock Module Block Diagram

*1 : RF[1:0] is defined in the Config. Register in PR3910 Core
*2 : SLOWBUS is defined in the Power Control Register
*3 : VIDREF[1:0] is defined in the Power Control Register
*4 : DIVMOD is defined in the Power Control Register

As shown in this block diagram, when DIVMOD bit is on default state (low); CLK2X and CLK are selected as input clocks for each function module such as UART and Timer etc. which would affect by changing of RF [1:0] Config. Register. If this bit is set; and then, FREECLK and XHFREE which are independent of RF[1:0] configuration, will be selected as clock sources for these module.

2. On page 77, the clock matrix table has a missing column of FREECLK.

RF[1:0]*1	SLOWBUS*2	CORECLK	DCLKOUT	CLK2X	FREECLK	CLK	XHFREE
00	0	F	F	F	F	F/2	F/2
01	0	F/2	F/2	F/2	F	F/4	F/2
10	0	F/4	F/4	F/4	F	F/8	F/2
11	0	F/8	F/8	F/8	F	F/16	F/2
00	1	F/2	F/2	F/2	F	F/4	F/2
01	1	F/4	F/4	F/4	F	F/8	F/2
10	1	F/8	F/8	F/8	F	F/16	F/2
11	1	F/16	F/16	F/16	F	F/32	F/2

3. On page 77 and 78, start from the last paragraph on page 77.

(Correction)

Video master clock is generated from XHFREE clock, independent of DIVMOD bit in Power Control Register (OFFSET = $1C4). The XHFREE clock is not affected by switching RF[1:0] bits, constant one-half rate of highest clock rate will be fed into the video master clock. This video master clock rate is only controlled by VIDRF[1:0]. (Refer Power Control Register (offset = $ 1C4) for VIDRF[1:0] bit description).

The SIBMCLK pin can be configured as an output. In this mode, SIB master clock is generated internally by dividing down CLK2X/FREE clock with the internal programmable divider. The SIBMCLK pin can also be configured as input. In this mode, all SIB clocks are derived from an external SIBMCLK oscillator source, which is asynchronous with respect to CLK. This mode allows optional decoupling of the SIB (as well as the CHI and UART) rates from the CPU core clock rate. The selected SIBMCLK source is then used as the SIB Master Clock for the SIB Module circuits, and is also used to generate the SIBSCLK and SIBSYNC externally driven PR31700 output signals.

The CHICLK pin can be configured as an output, for which the programmable rate is generated by dividing down from either CLK/XHFREE or the SIB Master Clock. The CHICLK pin can also be configured as input. In this mode, all CHI clocks are derived from an external peripheral source and the CHI Module will slave to this external clock. The selected CHICLK source is used as the CHI Master Clock for the CHI Module circuits, and is also used to generate the CHIFS externally driven PR31700 output signal.

The programmable UART Master Clock is generated by dividing down from either CLK/XHFREE or the SIB Master Clock, and is used as the master clock for the baud generator circuit within each UART Module.

The Clock Module also contains several fixed dividers for generating the master clocks for the RTC timer, and SPI circuits. These clocks are divided down from either CLK/XHFREE or CLK2X/FREE.

4. On page 145, The correct description of DIVMOD is;

DIVMOD:
Setting this bit will cause the internal clocks for each functional module to be independent of RF[1:0] . If this bit is set high, the source of internal clocks would be switched to FREECLK/XHFREE (independent system clocks) from CLK2X/CLK.

Data sheet status

Data sheet status	Product status	Definition [1]
Objective specification	Development	This data sheet contains the design target or goal specifications for product development. Specification may change in any manner without notice.
Preliminary specification	Qualification	This data sheet contains preliminary data, and supplementary data will be published at a later date. Philips Semiconductors reserves the right to make chages at any time without notice in order to improve design and supply the best possible product.
Product specification	Production	This data sheet contains final specifications. Philips Semiconductors reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.

[1] Please consult the most recently issued datasheet before initiating or completing a design.

Definitions

Short-form specification — The data in a short-form specification is extracted from a full data sheet with the same type number and title. For detailed information see the relevant data sheet or data handbook.

Limiting values definition — Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.

Application information — Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Disclaimers

Life support — These products are not designed for use in life support appliances, devices or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes — Philips Semiconductors reserves the right to make changes, without notice, in the products, including circuits, standard cells, and/or software, described or contained herein in order to improve design and/or performance. Philips Semiconductors assumes no responsibility or liability for the use of any of these products, conveys no license or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Philips Semiconductors
811 East Arques Avenue
P.O. Box 3409
Sunnyvale, California 94088–3409
Telephone 800-447-1500

485003/CR2/244pp Date of release: 10-98

Document order number: 9397 750 04585

Let's make things better.

Philips
Semiconductors

PHILIPS