Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem (PRU-ICSS)

This chapter describes the PRU-ICSS for the device.

Topic **Page**

4.1 Introduction

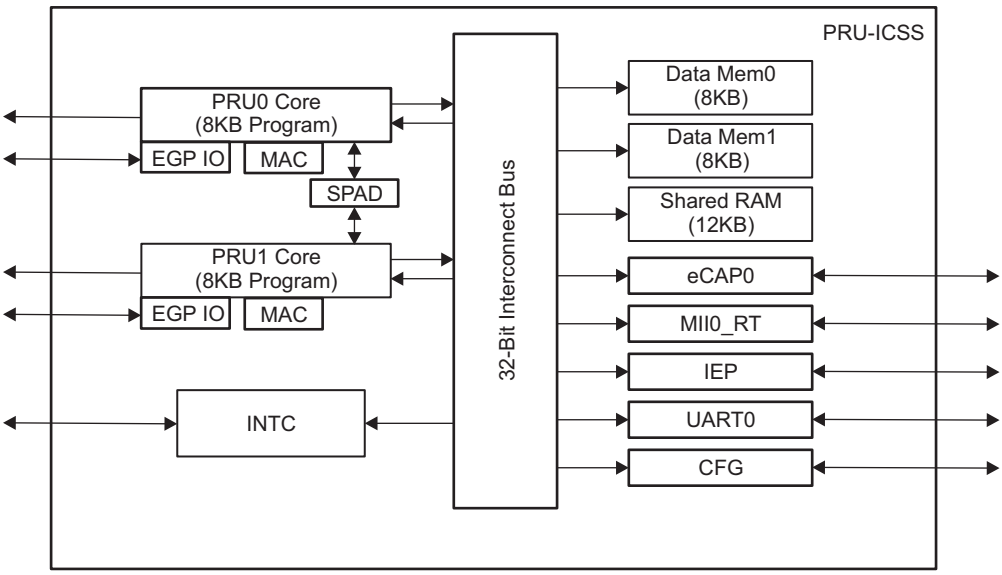
The Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem (PRU-ICSS) consists of dual 32-bit RISC cores (Programmable Real-Time Units, or PRUs), shared, data, and instruction memories, internal peripheral modules, and an interrupt controller (INTC). The programmable nature of the PRU, along with its access to pins, events and all SoC resources, provides flexibility in implementing fast real-time responses, specialized data handling operations, custom peripheral interfaces, and in offloading tasks from the other processor cores of the system-on-chip (SoC).

Figure 4-1 shows the PRU-ICSS details.

The PRUs have access to all resources on the SoC through the Interface/OCP Master port, and the external host processors can access the PRU-ICSS resources through the Interface/OCP Slave port. The 32-bit interconnect bus connects the various internal and external masters to the resources inside the PRU-ICSS. The INTC handles system input events and posts events back to the device-level host CPU.

The PRU cores are programmed with a small, deterministic instruction set. Each PRU can operate independently or in coordination with each other and can also work in coordination with the device-level host CPU. This interaction between processors is determined by the nature of the firmware loaded into the PRU's instruction memories.

Figure 4-1. PRU-ICSS Block Diagram



4.1.1 Features

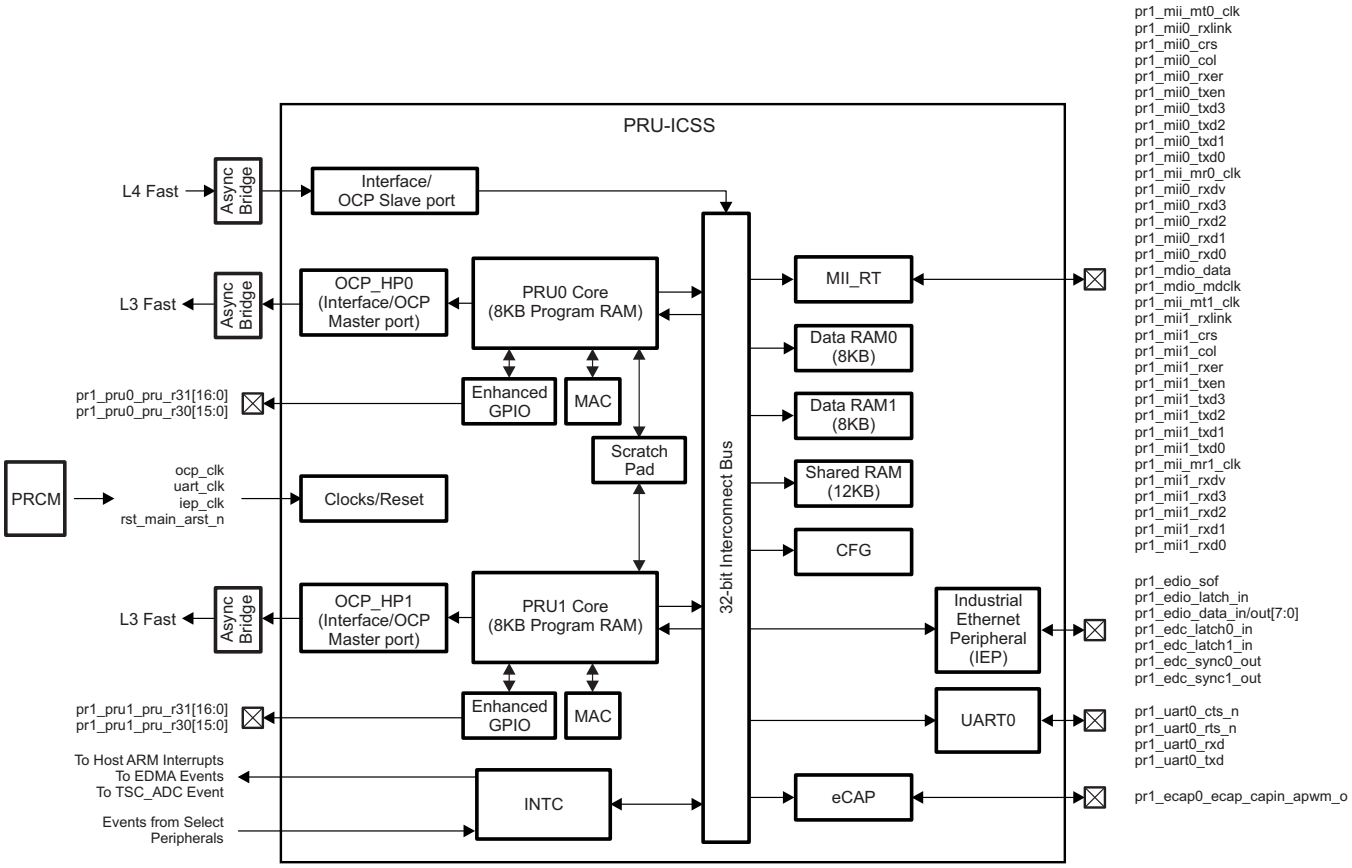
The PRU-ICSS includes the following main features:

- Two PRUs each with:
  - 8KB program memory
  - 8KB data memory
  - High Performance Interface/OCP Master port for accessing external memories
  - Enhanced GPIO (EGPIO) with async capture and serial support
  - Multiplier with optional accumulation (MPY/MAC)
- One scratch pad (SPAD) memory
  - 3 Banks of 30 32-bit registers
- Broadside direct connect between PRU cores within subsystem
- 12 KB general purpose shared memory
- One Interrupt Controller (INTC)
  - Up to 64 input events supported
  - Interrupt mapping to 10 interrupt channels
  - 10 Host interrupts (2 to PRU0 and PRU1, 8 output to chip level)
  - Each system event can be enabled and disabled
  - Each host event can be enabled and disabled
  - Hardware prioritization of events
- 16 software events generated by 2 PRUs
- One Ethernet MII_RT module with two MII ports and configurable connections to PRUs*
- One MDIO Port*
- One Industrial Ethernet Peripheral (IEP) to manage/generate Industrial Ethernet functions
  - One Industrial Ethernet timer with 10 capture* and eight compare events
  - Two Industrial Ethernet sync signals*
  - Two Industrial Ethernet 16-bit watchdog timers*
  - Industrial Ethernet digital IOs
- One 16550-compatible UART with a dedicated 192-MHz clock, supporting up to 12Mbaud for PROFIBUS DP
- One Enhanced Capture Module (ECAP)
- Flexible power management support
- Integrated 32-bit interconnect bus for connecting the various internal and external masters to the resources inside the PRU-ICSS
- Interface/OCP Slave port for external masters to access PRU-ICSS memories
- Optional address translation for PRU transaction to External Host
- All memories within the PRU-ICSS support parity

4.2 Integration

The device includes a Programmable Real-Time Unit Subsystem and Industrial Communication Subsystem (PRU-ICSS) consisting of two independent Programmable Real-time Units (PRUs). Each PRU is a 32-bit Load/Store RISC processor with dedicated memories. The PRU-ICSS integration is shown in Figure 4-2.

Figure 4-2. PRU-ICSS Integration



For the availability of all features, see the device features in Chapter 1, *Introduction*.

4.2.1 PRU-ICSS Connectivity Attributes

The general connectivity attributes for the PRU subsystem are shown in Table 4-1.

Table 4-1. PRU-ICSS Connectivity Attributes

Attributes	Type
Power Domain	Peripheral Domain
Clock Domain	PD_PER_PRU_ICSS_OCP_GCLK (OCP clock) PD_PER_PRU_ICSS_IEP_GCLK (IEP clock) PD_PER_PRU_ICSS_UART_GCLK (UART clock)
Reset Signals	PRU_ICSS_LRST_N
Idle/Wakeup Signals	Standby Idle
Interrupt Requests	8 Interrupts pr1_host_intr[7:1](1) to MPU Subsystem pr1_host_intr[0](1) to MPU Subsystem and TSC_ADC
DMA Requests	No dedicated DMA events but pr1_host_intr[7:6](1) interrupt outputs also connected as DMA events
Physical Address	L4 Fast Slave Port

(1) pr1_host_intr[0:7] corresponds to Host-2 to Host-9 of the PRU-ICSS interrupt controller.

4.2.2 PRU-ICSS Clock and Reset Management

The PRU-ICSS module uses the following functional and OCP interface clocks.

Table 4-2. PRU-ICSS Clock Signals

Clock Signal	Max Freq	Reference / Source	Comments
l3_clk Interface Clock	200 MHz	CORE_CLKOUTM4 or Display PLL CLKOUT	pd_per_pru_icss_ocp_gclk from PRCM Clocks both L3 master and L4F slave
uart_clk Functional Clock	192 MHz	PER_CLKOUTM2	pd_per_pru_icss_uart_gclk from PRCM UART Clock
iep_clk Functional Clock	200 MHz	CORE_CLKOUTM4	pd_per_pru_icss_iep_gclk from PRCM Industrial Ethernet Peripheral Clock

4.2.3 PRU-ICSS Pin List

The PRU-ICSS external interface signals are shown in Table 4-3. PRU GPI/GPO pin function depends on the operation mode. SeeTable 4-3 for a complete list of pin functions for each mode.

Table 4-3. PRU-ICSS Pin List

Pin	Type	Description
pr1_mii_mr0_clk	I	MII0 Receive Clock
pr1_mii0_rxdv	I	MII0 Receive Data Valid
pr1_mii0_rxd[3:0]	I	MII0 Receive Data
pr1_mii0_rxlink	I	MII0 Receive Link
pr1_mii0_rxer	I	MII0 Receive Data Error
pr1_mii0_crs	I	MII0 Carrier Sense
pr1_mii0_col	I	MII0 Carrier Sense
pr1_mii_mt0_clk	I	MII0 Transmit Clock
pr1_mii0_txen	O	MII0 Transmit Enable
pr1_mii0_txd[3:0]	O	MII0 Transmit Data
pr1_mii_mr1_clk	I	MII1 Receive Clock
pr1_mii1_rxdv	I	MII1 Receive Data Valid
pr1_mii1_rxd[3:0]	I	MII1 Receive Data
pr1_mii1_rxlink	I	MII1 Receive Link
pr1_mii1_rxer	I	MII1 Receive Data Error
pr1_mii1_crs	I	MII1 Carrier Sense
pr1_mii1_col	I	MII1 Carrier Sense
pr1_mii_mt1_clk	I	MII1 Transmit Clock
pr1_mii1_txen	O	MII1 Transmit Enable
pr1_mii1_txd[3:0]	O	MII1 Transmit Data
pr1_mdio_mdclk	O	MDIO Clk
pr1_mdio_data	I/O	MDIO Data
pr1_edio_sof	O	ECAT Digital I/O Start of Frame
pr1_edio_latch_in	I	ECAT Digital I/O Latch In
pr1_edio_data_in[7:0]	I	ECAT Digital I/O Data In
pr1_edio_data_out[7:0]	O	ECAT Digital I/O Data Out
pr1_edc_sync0_out	O	ECAT Distributed Clock Sync Out
pr1_edc_sync1_out	O	ECAT Distributed Clock Sync Out
pr1_edc_latch0_in	I	ECAT Distributed Clock Latch In
pr1_edc_latch1_in	I	ECAT Distributed Clock Latch In
pr1_uart0_cts_n	I	UART Clear to Send
pr1_uart0_rts_n	O	UART Request to Send
pr1_uart0_rxd	I	UART Receive Data
pr1_uart0_txd	O	UART Transmit Data
pr1_ecap0_ecap_capin_apwm_o	I/O	Enhanced capture (ECAP) input or Auxiliary PWM out
pr1_pru0_pru_r30[15:0]	O	PRU0 Register R30 (GPO) Outputs
pr1_pru0_pru_r31[16:0]	I	PRU0 Register R31 (GPI)Inputs
pr1_pru1_pru_r30[15:0]	O	PRU1 Register R30 (GPO) Outputs
pr1_pru1_pru_r31[16:0]	I	PRU1 Register R31 (GPI) Inputs

4.2.4 PRU-ICSS Internal Pinmux

The PRU-ICSS supports an internal pinmux selection option that expands the device-level pinmuxing. The internal pinmuxing is programmable through the PIN_MX register of the PRU-ICSS CFG register space.

The pin_mux_sel[0] determines the external signals routed to the internal input signals, mii0_rxd[3:0]. The pin_mux_sel[1] determines the internal output signals routed to the external signals, pr1_pru0_pru_r30[13:8], and the external signals routed to the internal input signals, pru0_r30[5:0].

Note: pin_mux_sel[x] = 0 is always the standard pin mapping (default).

Table 4-4. PRU-ICSS Internal Signal Muxing: pin_mux_sel[0]

	pin_mux_sel[0] = 1	pin_mux_sel[0] = 0
Internal PRU-ICSS Signal Name	**External Chip Level Signal Name**	
mii0_rxd[3:0]	pr1_pru1_pru_r31[8:11]	pr1_mii0_rxd[3:0]

Figure 4-3. PRU-ICSS Internal Signal Muxing: pin_mux_sel[0]

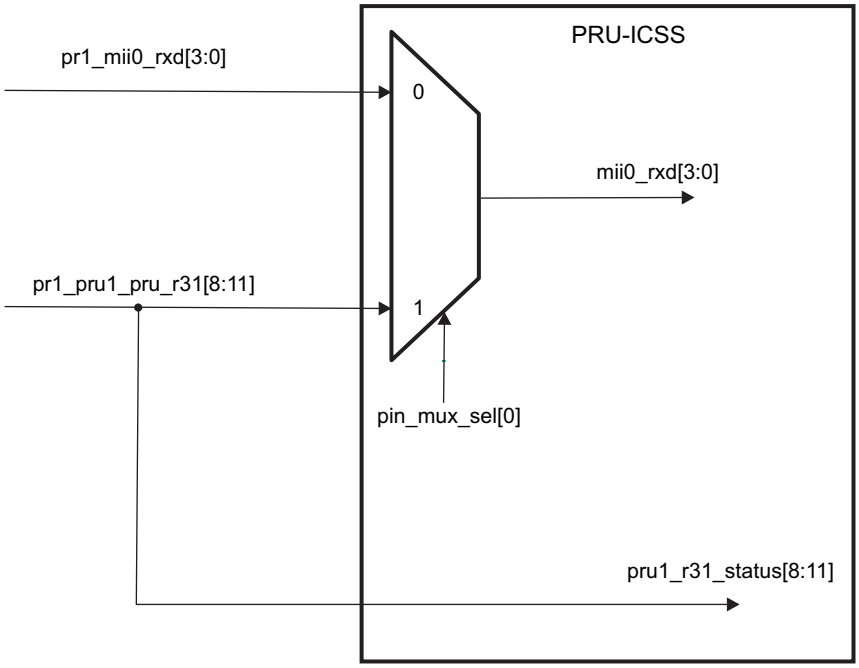
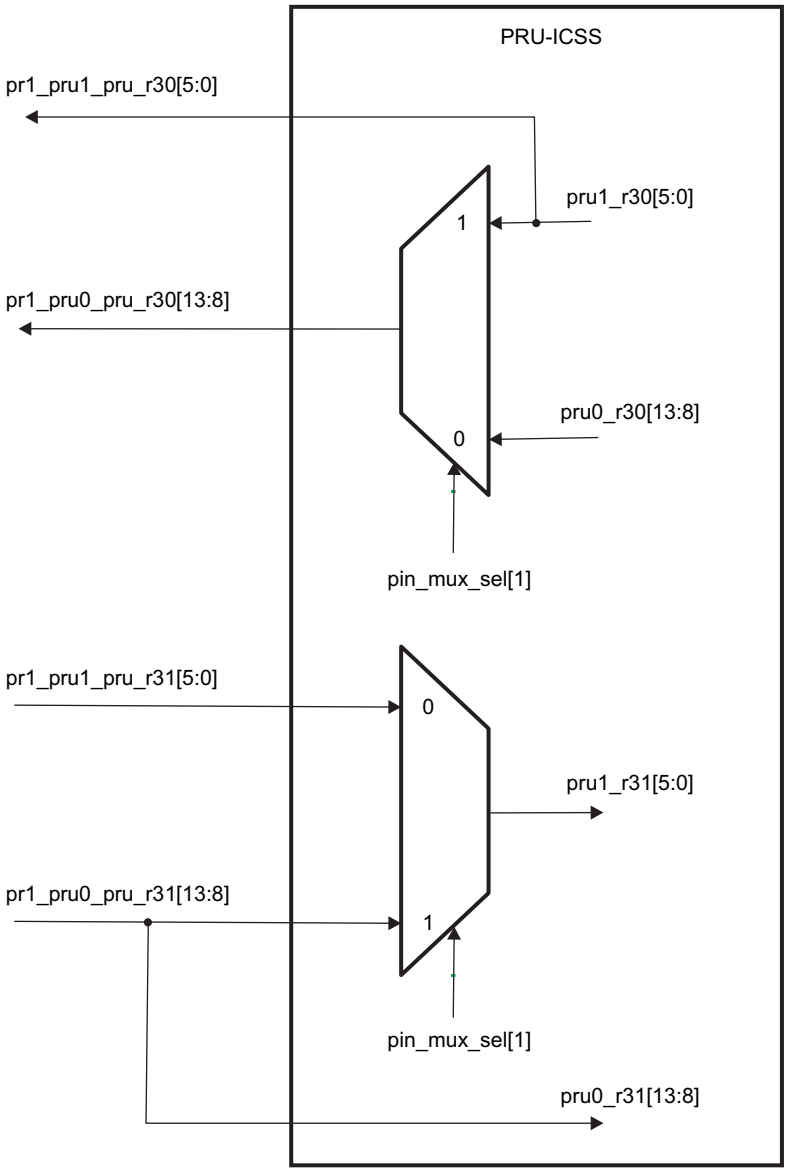


Table 4-5. PRU-ICSS Internal Signal Muxing: pin_mux_sel[1]

	pin_mux_sel[1] = 1	pin_mux_sel[1] = 0
External Chip Level Signal Name	**Internal PRU-ICSS Signal Name**	
pr1_pru0_pru_r30[13:8]	pru1_r30[5:0]	pru0_r30[13:8]
Internal PRU-ICSS Signal Name	**External Chip Level Signal Name**	
pru1_r31_status[5:0]	pr1_pru0_pru_r31[13:8]	pr1_pru1_pru_r31[5:0]

Figure 4-4. PRU-ICSS Internal Signal Muxing: pin_mux_sel[1]



4.3 PRU-ICSS Memory Map Overview

The PRU-ICSS comprises various distinct addressable regions that are mapped to both a local and global memory map. The local memory maps are maps with respect to the PRU core point of view. The global memory maps are maps with respect to the Host point of view, but can also be accessed by the PRU-ICSS.

4.3.1 Local Memory Map

The PRU-ICSS memory map is documented in Table 4-6 (Instruction Space) and in Table 4-7 (Data Space). Note that these two memory maps are implemented inside the PRU-ICSS and are local to the components of the PRU-ICSS.

4.3.1.1 Local Instruction Memory Map

Each PRU core has a dedicated 8KB of Instruction Memory which needs to be initialized by a Host processor before the PRU executes instructions. This region is only accessible to masters via the interface/ OCP slave port when the PRU is not running.

Table 4-6. Local Instruction Memory Map

Start Address	PRU0	PRU1
0x0000_0000	8KB IRAM	8KB IRAM

4.3.1.2 Local Data Memory Map

The local data memory map in Table 4-7 allows each PRU core to access the PRU-ICSS addressable regions and the external host's memory map.

The PRU accesses the external Host memory map through the Interface/OCP Master port (System OCP_HP0/1) starting at address 0x0008_0000. By default, memory addresses between 0x0000_0000 – 0x0007_FFFF will correspond to the PRU-ICSS local address in Table 4-7. To access an address between 0x0000_0000–0x0007_FFFF of the external Host map, the address offset of –0x0008_0000 feature is enabled through the PMAO register of the PRU-ICSS CFG register space.

Table 4-7. Local Data Memory Map

Start Address	PRU0	PRU1
0x0000_0000	Data 8KB RAM 0 (1)	Data 8KB RAM 1 (1)
0x0000_2000	Data 8KB RAM 1 (1)	Data 8KB RAM 0 (1)
0x0001_0000	Shared Data 12KB RAM 2	Shared Data 12KB RAM 2
0x0002_0000	INTC	INTC
0x0002_2000	PRU0 Control	PRU0 Control
0x0002_2400	Reserved	Reserved
0x0002_4000	PRU1 Control	PRU1 Control
0x0002_4400	Reserved	Reserved
0x0002_6000	CFG	CFG
0x0002_8000	UART 0	UART 0
0x0002_A000	Reserved	Reserved
0x0002_C000	Reserved	Reserved
0x0002_E000	IEP	IEP
0x0003_0000	eCAP 0	eCAP 0
0x0003_2000	MII_RT_CFG	MII_RT_CFG
0x0003_2400	MII_MDIO	MII_MDIO
0x0003_4000	Reserved	Reserved

(1) Data RAM0 is intended to be the primary data memory for PRU0, as is Data RAM1 for PRU1. However, both PRU cores can access Data RAM0 and Data RAM1 to pass information between PRUs. Each PRU core accesses their intended Data RAM at address 0x0000_0000 and the other Data RAM at address 0x0000_2000.

Table 4-7. Local Data Memory Map (continued)

Start Address	PRU0	PRU1
0x0003_8000	Reserved	Reserved
0x0004_0000	Reserved	Reserved
0x0008_0000	System OCP_HP0	System OCP_HP1

4.3.2 Global Memory Map

The global view of the PRU-ICSS internal memories and control ports is shown in Table 4-8. The offset addresses of each region are implemented inside the PRU-ICSS but the global device memory mapping places the PRU-ICSS slave port in the address range shown in the external Host top-level memory map.

The global memory map is with respect to the Host point of view, but it can also be accessed by the PRU-ICSS. Note that PRU0 and PRU1 can use either the local or global addresses to access their internal memories, but using the local addresses will provide access time several cycles faster than using the global addresses. This is because when accessing via the global address the access needs to be routed through the switch fabric outside PRU-ICSS and back in through the PRU-ICSS slave port.

Each of the PRUs can access the rest of the device memory (including memory mapped peripheral and configuration registers) using the global memory space addresses. See Table 4-8, *Global Memory Map*, for base addresses of each module in the device.

Table 4-8. Global Memory Map

Offset Address	PRU-ICSS
0x0000_0000	Data 8KB RAM 0
0x0000_2000	Data 8KB RAM 1
0x0001_0000	Shared Data 12KB RAM 2
0x0002_0000	INTC
0x0002_2000	PRU0 Control
0x0002_2400	PRU0 Debug
0x0002_4000	PRU1 Control
0x0002_4400	PRU1 Debug
0x0002_6000	CFG
0x0002_8000	UART 0
0x0002_A000	Reserved
0x0002_C000	Reserved
0x0002_E000	IEP
0x0003_0000	eCAP 0
0x0003_2000	MII_RT_CFG
0x0003_2400	MII_MDIO
0x0003_4000	PRU0 8KB IRAM
0x0003_8000	PRU1 8KB IRAM
0x0004_0000	Reserved

4.4 Functional Description

4.4.1 PRU Cores

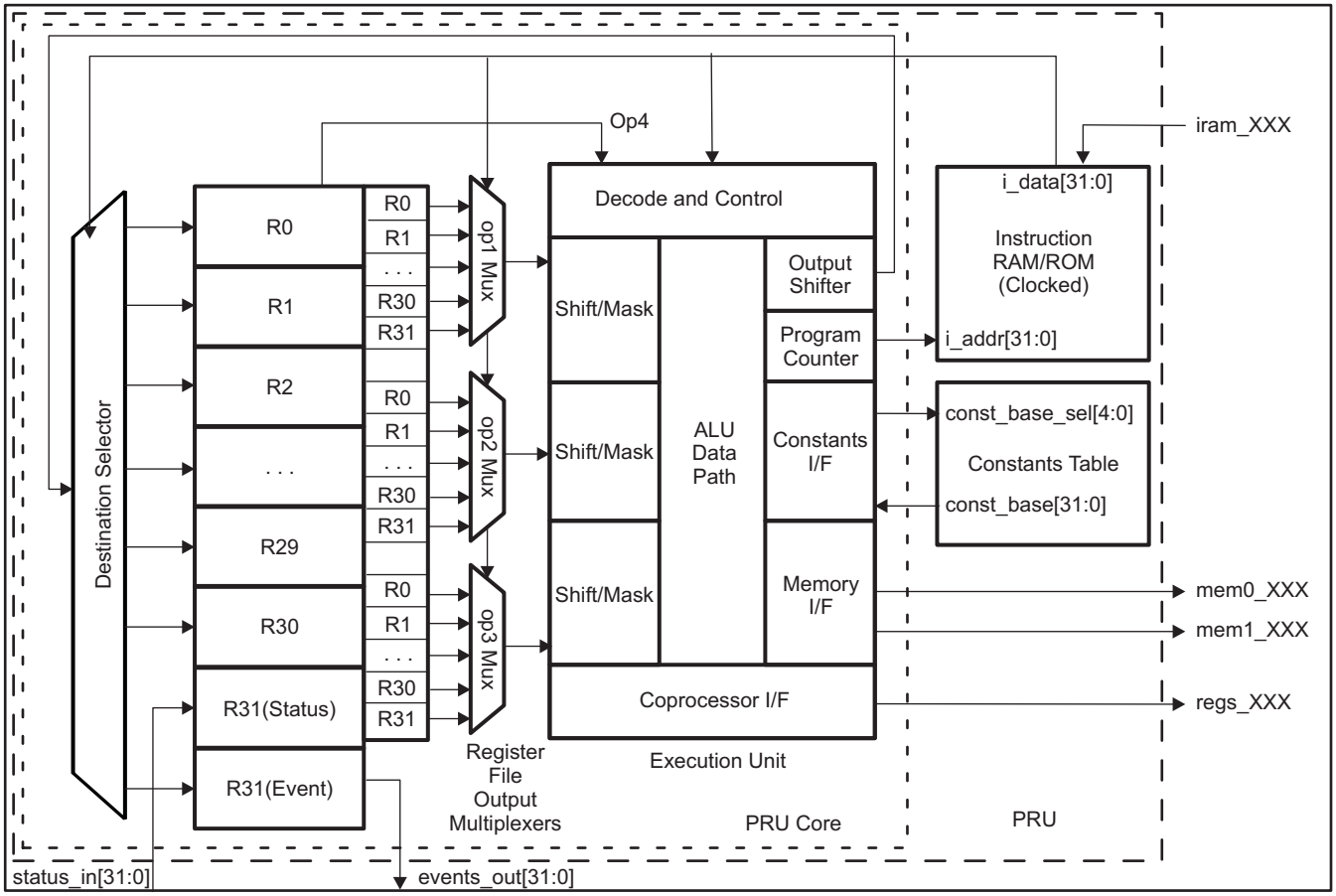
The PRU is a processor optimized for performing embedded tasks that require manipulation of packed memory mapped data structures, handling of system events that have tight real-time constraints and interfacing with systems external to the SoC. The PRU is both very small and very efficient at handling such tasks.

The major attributes of the PRU are as follows.

Attribute	Value
IO Architecture	Load / Store
Data Flow Architecture	Register to Register
Core Level Bus Architecture Type Instruction I/F Memory I/F 0 Memory I/F 1	 4-Bus Harvard (1 Instruction, 3 Data) 32-Bit 32-Bit 32-Bit
Execution Model Issue Type Pipelining Ordering ALU Type	 Scalar None (Purposefully) In Order Unsigned Integer
Registers General Purpose (GP) External Status GP / Indexing Addressability in Instruction	 30 (R1 – R30) 1 (R31) 1 (R0) Bit, Byte (8-bit), Halfword (16-bit), Word (32-bit), Pointer
Addressing Modes Load Immediate Load / Store – Memory	 16-bit Immediate Register Base + Register Offset Register Base + 8-bit Immediate Offset Register Base with auto increment / decrement Constant Table Base + Register Offset Constant Table Base + 8-bit Immediate Offset Constant Table Base with auto increment / decrement
Data Path Width	32-Bits
Instruction Width	32-Bits
Accessibility to Internal PRU Structures	Provides 32-bit slave with three regions: • Instruction RAM • Control / Status registers • Debug access to internal registers (R0-R31) and constant table

The processor is based on a four-bus architecture which allows instructions to be fetched and executed concurrently with data transfers. In addition, an input is provided in order to allow external status information to be reflected in the internal processor status register. Figure 4-5 shows a block diagram of the processing element and the associated instruction RAM/ROM that contains the code that is to be executed.

Figure 4-5. PRU Block Diagram



4.4.1.1 Constants Table

The PRU Constants Table is a structure of hard-coded memory addresses for commonly used peripherals and memories. The Constants table exists to more efficiently load/store data to these commonly accessed addresses by:

- Reduce a PRU instruction by not needing to pre-load an address into the internal register file before loading or storing data to a memory address
- Maximize the usage of the PRU register file for embedded processing applications by moving many of the commonly used constants or deterministically calculated base addresses from the internal register file to an external table.

Table 4-9. PRU0/1 Constants Table

Entry No.	Region Pointed To	Value [31:0]
0	PRU-ICSS INTC (local)	0x0002_0000
1	DMTIMER2	0x4804_0000
2	I2C1	0x4802_A000
3	PRU-ICSS eCAP (local)	0x0003_0000
4	PRU-ICSS CFG (local)	0x0002_6000

Table 4-9. PRU0/1 Constants Table (continued)

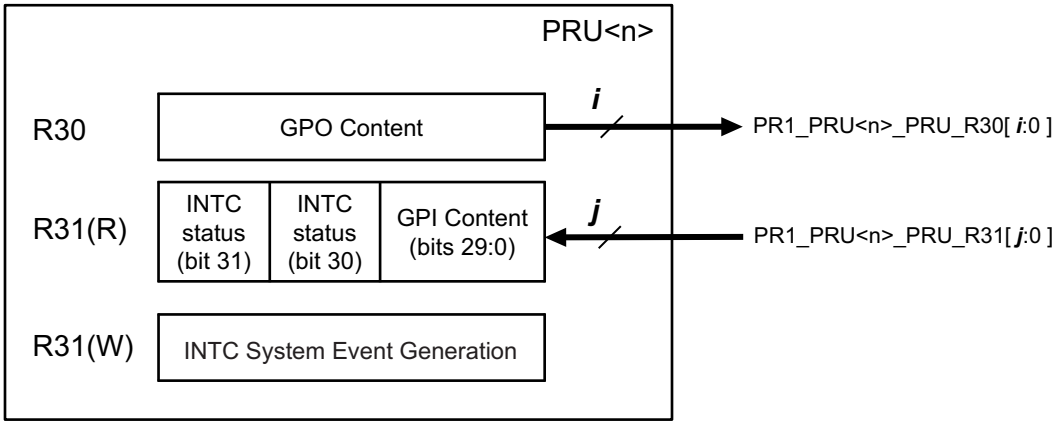
Entry No.	Region Pointed To	Value [31:0]
5	MMCHS 0	0x4806_0000
6	MCSPI 0	0x4803_0000
7	PRU-ICSS UART0 (local)	0x0002_8000
8	McASP0 DMA	0x4600_0000
9	GEMAC	0x4A10_0000
10	Reserved	0x4831_8000
11	UART1	0x4802_2000
12	UART2	0x4802_4000
13	Reserved	0x4831_0000
14	DCAN0	0x481C_C000
15	DCAN1	0x481D_0000
16	MCSPI 1	0x481A_0000
17	I2C2	0x4819_C000
18	eHRPWM1/eCAP1/eQEP1	0x4830_0000
19	eHRPWM2/eCAP2/eQEP2	0x4830_2000
20	eHRPWM3/eCAP3/eQEP3	0x4830_4000
21	PRU-ICSS MDIO (local)	0x0003_2400
22	Mailbox 0	0x480C_8000
23	Spinlock	0x480C_A000
24	PRU-ICSS PRU0/1 Data RAM (local)	0x0000_0n00, n = c24_blk_index[3:0]
25	PRU-ICSS PRU1/0 Data RAM (local)	0x0000_2n00, n = c25_blk_index[3:0]
26	PRU-ICSS IEP (local)	0x0002_En00, n = c26_blk_index[3:0]
27	PRU-ICSS MII_RT (local)	0x0003_2n00, n = c27_blk_index[3:0]
28	PRU-ICSS Shared RAM (local)	0x00nn_nn00, nnnn = c28_pointer[15:0]
29	TPCC	0x49nn_nn00, nnnn = c29_pointer[15:0]
30	L3 OCMC0	0x40nn_nn00, nnnn = c30_pointer[15:0]
31	EMIF0 DDR Base	0x80nn_nn00, nnnn = c31_pointer[15:0]

NOTE: Addresses in constants entries 24–31 are partially programmable. Their programmable bit field (for example, c24_blk_index[3:0]) is programmable through the PRU CTRL register space. As a general rule, the PRU should configure this field before using the partially programmable constant entries.

4.4.1.2 PRU Module Interface to PRU I/Os and INTC

The PRU module interface consists of the PRU internal registers 30 and 31 (R30 and R31). Figure 4-6 shows the PRU module interface and the functionality of R30 and R31. The register R31 serves as an interface with the dedicated PRU general purpose input (GPI) pins and PRU interrupt controller (INTC). Reading R31 returns status information from the GPI pins and INTC via the PRU Real-Time Status Interface. Writing to R31 generates PRU system events via the PRU Event Interface. The register R30 serves as an interface with the dedicated PRU general purpose output (GPO) pins. The R30/R31 GPO or GPI register content changes depending on the selected configuration (such as MII_RT, Parallel Capture, Shift Out, and so forth). See the subsequent sections for more details.

Figure 4-6. PRU Module Interface



4.4.1.2.1 Real-Time Status Interface Mapping (R31): Interrupt Events Input

The PRU Real-Time Status Interface directly feeds information into register 31 (R31) of the PRU's internal register file. The firmware on the PRU uses the status information to make decisions during execution. The status interface is comprised of signals from different modules inside of the PRU-ICSS which require some level of interaction with the PRU. More details on the Host interrupts imported into bit 30 and 31 of register R31 of both the PRUs is provided in Section 4.4.2, *Interrupt Controller (INTC)*.

Table 4-10. Real-Time Status Interface Mapping (R31) Field Descriptions

Bit	Field	Value	Description
31	pru_intr_in[1]		PRU Host Interrupt 1 from local INTC
30	pru_intr_in[0]		PRU Host Interrupt 0 from local INTC
29-0	pru<n>_r31_status[29:0]		Status inputs from primary input via Enhanced GPI port

4.4.1.2.2 Event Interface Mapping (R31): PRU System Events

This PRU Event Interface directly feeds pulsed event information out of the PRU's internal ALU. These events are exported out of the PRU-ICSS and need to be connected to the system event controller at the SoC level. The event interface can be used by the firmware to create interrupts from the PRU to the Host processor.

Figure 4-7. Event Interface Mapping (R31)

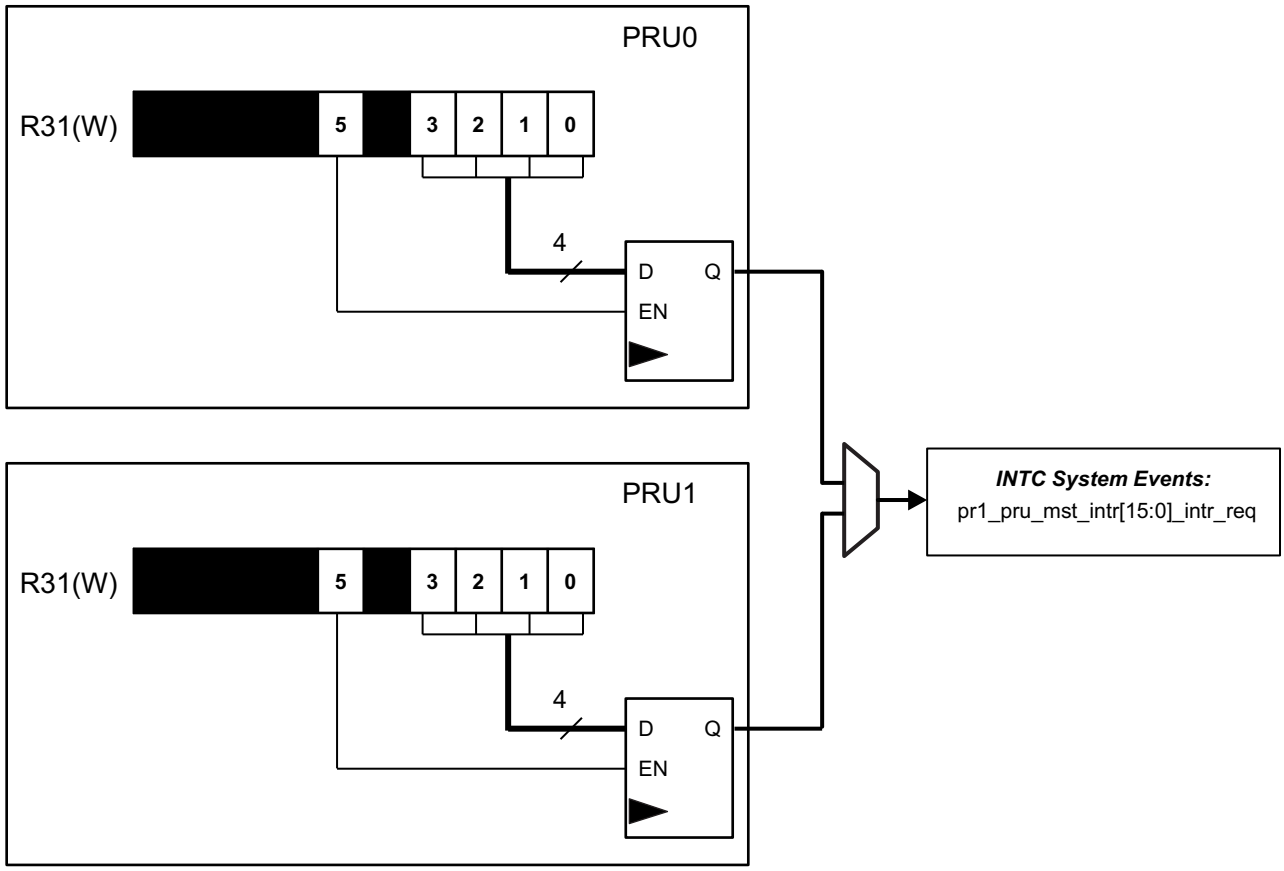


Table 4-11. Event Interface Mapping (R31) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved		
5	pru<n>_r31_vec_valid		Valid strobe for vector output
4	Reserved		
3-0	pru<n>_r31_vec[3:0]		Vector output

Simultaneously writing a '1' to pru<n>_r31_vec_valid (R31 bit 5) and a channel number from 0-15 to pru<n>_r31_vec[3:0] (R31 bits 3:0) creates a pulse on the output of the corresponding pr1_pru_mst_intr[x]_intr_req INTC system event (Table 4-22). For example, writing '100000' will generate a pulse on pr1_pru_mst_intr[0]_intr_req, writing '100001' will generate a pulse on pr1_pru_mst_intr[1]_intr_req, and so on to where writing '101111' will generate a pulse on pr1_pru_mst_intr[15]_intr_req and writing '0xxxxx' will not generate any system event pulses. The output values from both PRU cores in a subsystem are ORed together.

The output channels 0-15 are connected to the PRU-ICSS INTC system events 16-31, respectively. This allows the PRU to assert one of the system events 16-31 by writing to its own R31 register. The system event is used to either post a completion event to one of the host CPUs (ARM) or to signal the other PRU. The host to be signaled is determined by the system event to interrupt channel mapping (programmable). The 16 events are named as pr1_pru_mst_intr<15:0>_intr_req. For more details, see Section 4.4.2, *Interrupt Controller (INTC)*.

4.4.1.2.3 General-Purpose Inputs (R31): Enhanced PRU GP Module

The PRU-ICSS implements an enhanced General-Purpose Input/Output (GPIO) module that supports the following general-purpose input modes: direct input, 16-bit parallel capture, 28-bit serial shift in, and MII_RT. Register R31 serves as an interface with the general purpose inputs.

Table 4-12 describes the input modes in detail. Note each PRU core can only be configured for one GPI mode at a time. Each mode uses the same R31 signals and internal register bits for different purposes. A summary is found in Table 4-13.

Table 4-12. PRU R31 (GPI) Modes

Mode	Function	Configuration
Direct input	GPI[0:29] feeds directly into the PRU R31	Default mode
16-bit parallel capture	DATAIN[0:15] is captured by the posedge or negedge of CLOCKIN	• Enabled by CFG_GPCFGn register • CLOCKIN edge selected by CFG_GPCFGn register
28-bit shift in	DATAIN is sampled and shifted into a 28-bit shift. Shift Counter (Cnt_16) feature uses: • Cnt_16 (Shift Counter) feature is mapped to pru<n>_r31_status[28] • SB (Start Bit detection) feature is mapped to pru<n>_r31_status[29]	• Enabled by CFG_GPCFGn register • Cnt_16 is self clearing and is connected to the PRU INTC • Start Bit (SB) is cleared by CFG_GPCFGn register
MII_RT	mii_rt_r31_status [29:0] internally driven by the MII_RT module	Enabled by CFG

Table 4-13. PRU GPI Signals and Configurations

Pad Names at Device Level	GPI Modes		
	Direct Input	Parallel Capture	28-Bit Shift In
pr1_pru<n>_pru_r31_0	GPI0	DATAIN0	DATAIN
pr1_pru<n>_pru_r31_1	GPI1	DATAIN1	
pr1_pru<n>_pru_r31_2	GPI2	DATAIN2	
pr1_pru<n>_pru_r31_3	GPI3	DATAIN3	
pr1_pru<n>_pru_r31_4	GPI4	DATAIN4	
pr1_pru<n>_pru_r31_5	GPI5	DATAIN5	
pr1_pru<n>_pru_r31_6	GPI6	DATAIN6	
pr1_pru<n>_pru_r31_7	GPI7	DATAIN7	
pr1_pru<n>_pru_r31_8	GPI8	DATAIN8	
pr1_pru<n>_pru_r31_9	GPI9	DATAIN9	
pr1_pru<n>_pru_r31_10	GPI10	DATAIN10	
pr1_pru<n>_pru_r31_11	GPI11	DATAIN11	
pr1_pru<n>_pru_r31_12	GPI12	DATAIN12	
pr1_pru<n>_pru_r31_13	GPI13	DATAIN13	
pr1_pru<n>_pru_r31_14	GPI14	DATAIN14	
pr1_pru<n>_pru_r31_15	GPI15	DATAIN15	
pr1_pru<n>_pru_r31_16	GPI16	CLOCKIN	
pr1_pru<n>_pru_r31_17	GPI17		
pr1_pru<n>_pru_r31_18	GPI18		
pr1_pru<n>_pru_r31_19	GPI19		
pr1_pru<n>_pru_r31_20	GPI20		
pr1_pru<n>_pru_r31_21	GPI21		
pr1_pru<n>_pru_r31_22	GPI22		
pr1_pru<n>_pru_r31_23	GPI23		

Table 4-13. PRU GPI Signals and Configurations (continued)

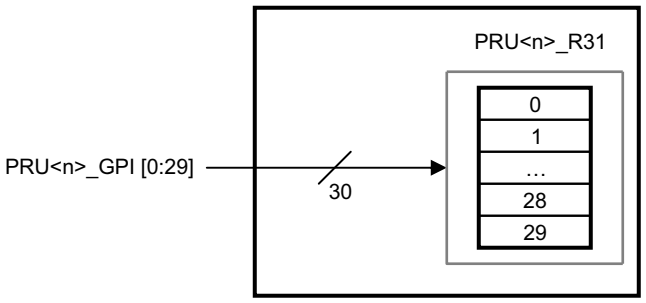
Pad Names at Device Level	GPI Modes		
	Direct Input	Parallel Capture	28-Bit Shift In
pr1_pru<n>_pru_r31_24	GPI24		
pr1_pru<n>_pru_r31_25	GPI25		
pr1_pru<n>_pru_r31_26	GPI26		
pr1_pru<n>_pru_r31_27	GPI27		
pr1_pru<n>_pru_r31_28	GPI28		
pr1_pru<n>_pru_r31_29	GPI29		

NOTE: Some devices may not pin out all 30 bits of R31. For which pins are available on this device, see Section 4.2.3, *PRU-ICSS Pin List*. See the device's datasheet for device-specific pin mapping.

4.4.1.2.3.1 Direct Input

The pru<n>_r31_status [0:29] bits of the internal PRU register file are mapped to device-level general purpose input pins (PRU<n>_GPI[0:29]). In GPI Direct Input mode, PRU<n>_GPI[0:29] feeds directly to pru<n>_r31_status [0:29]. There are 30 possible general purpose inputs per PRU core; however, some devices may not pin out all of these signals. See the device-specific datasheet for device specific pin mapping.

Figure 4-8. PRU R31 (GPI) Direct Input Mode Block Diagram

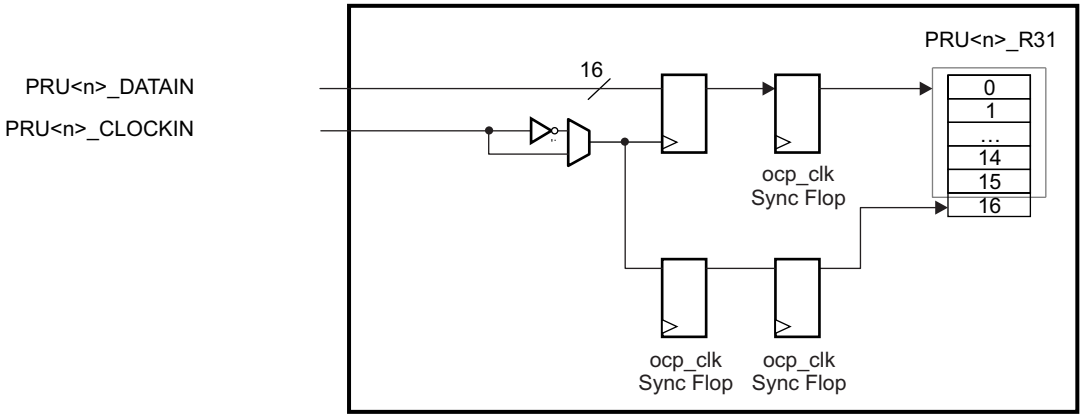


4.4.1.2.3.2 16-Bit Parallel Capture

The pru<n>_r31_status [0:15] and pru<n>_r31_status [16] bits of the internal PRU register file are mapped to device-level general purpose input pins (PRU<n>_DATAIN[0:15] and PRU<n>_CLOCKIN, respectively). PRU<n>_CLOCKIN is designated as an external strobe clock and is used to capture PRU<n>_DATAIN[0:15].

The PRU<n>_DATAIN can be captured either on the positive or negative edge of PRU<n>_CLOCK, programmable through the PRU-ICSS CFG register space. If the clocking is configured through the PRU-ICSS CFG register space to be positive, then it will equal PRU<n>_CLOCK. If the clocking is configured to be negative, then it will equal PRU<n>_CLOCK inverted.

Figure 4-9. PRU R31 (GPI) 16-Bit Parallel Capture Mode Block Diagram



4.4.1.2.3.3 28-Bit Shift In

In 28-bit Shift In mode, the device-level general purpose input pin PRU<n>_DATAIN is sampled and shifted into a 28-bit shift register on an internal clock pulse. The register fills in lsb order (from bit 0 to 27) and then overflows into a bit bucket. The 28-bit register is mapped to pru<n>_r31_status [0:27] and can be cleared in software through the PRU-ICSS CFG register space.

Note that the PRU will continually capture and shift the DATAIN input when the GPI mode has been set to 28-bit shift in.

The shift rate is controlled by the effective divisor of two cascaded dividers applied to the ocp_clk. These cascaded dividers can each be configured through the PRU-ICSS CFG register space to a value of {1, 1.5, …, 16}. Table 4-14 lists sample effective clock values and the divisor values that can be used to generate these clocks.

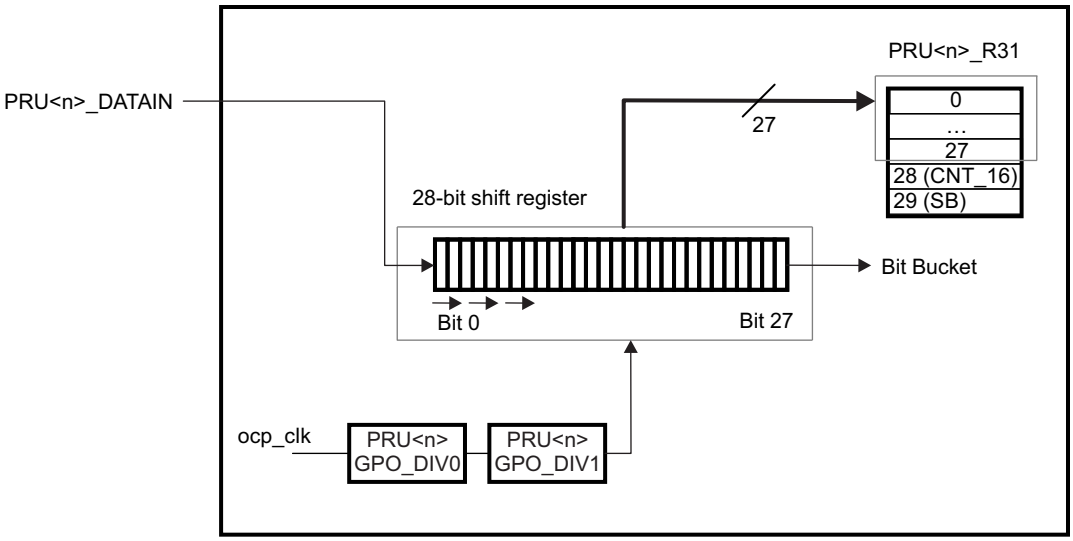
Table 4-14. Effective Clock Values

Generated clock	PRU<n>_GPI_DIV0	PRU<n>_GPI_DIV1
8-MHz	12.5 (0x17)	2 (0x02)
10-MHz	10 (0x12)	2 (0x02)
16-MHz	16 (0x1e)	1 (0x00)
20-MHz	10 (0x12)	1 (0x00)

The 28-bit Shift In mode also supports these features:

- Start Bit detection (SB) is mapped to pru<n>_r31_status [29] and is set when the first 1 is captured on PRU<n>_DATAIN. The SB flag in pru<n>_r31_status [29] is cleared in software through the CFG_GPCFGn register of the PRU-ICSS CFG register space.
- CNT_16 (Shift Counter) is mapped to pru<n>_r31_status [28] and is set on every 16 shift clock sample after the Start Bit has been received. CNT_16 is self clearing and is connected to the PRU-ICSS INTC. For more details, see Section 4.4.2, *Interrupt Controller (INTC)*.

Figure 4-10. PRU R31 (GPI) 28-Bit Shift In Mode



4.4.1.2.4 General-Purpose Outputs (R30): Enhanced PRU GP Module

The PRU-ICSS implements an enhanced General Purpose Input Output (GPIO) module that supports two general-purpose output modes: direct output and shift out..

Table 4-15 describes these modes in detail. Note that each PRU core can only be configured for one GPO mode at a time. Each mode uses the same R30 signals and internal register bits for different purposes. A summary is found in Table 4-16.

Table 4-15. PRU R30 (GPO) Modes

Mode	Function	Configuration
Direct Output	pru<n>_r30[0:31] feeds directly to GPO[0:31]	Default mode
Shift Out	• pru<n>_r30[0] is shifted out on DATAOUT on every rising edge of pru<n>_r30[1] (CLOCKOUT). • LOAD_GPO_SH0 (Load Shadow Register 0) is mapped to pru<n>_r30[29] • LOAD_GPO_SH1 (Load Shadow Register 1) is mapped to pru<n>_r30[30] • ENABLE_SHIFT is mapped to pru<n>_r30[31]	Enabled by CFG_GPCFGn register

Table 4-16. PRU GPO Signals and Configurations

Pad Names at Device Level	GPO Modes	
	Direct Output	Shift Out
pr1_pru<n>_pru_r30_0	GPO0	DATAOUT
pr1_pru<n>_pru_r30_1	GPO1	CLOCKOUT
pr1_pru<n>_pru_r30_2	GPO2	
pr1_pru<n>_pru_r30_3	GPO3	
pr1_pru<n>_pru_r30_4	GPO4	
pr1_pru<n>_pru_r30_5	GPO5	
pr1_pru<n>_pru_r30_6	GPO6	
pr1_pru<n>_pru_r30_7	GPO7	
pr1_pru<n>_pru_r30_8	GPO8	
pr1_pru<n>_pru_r30_9	GPO9	
pr1_pru<n>_pru_r30_10	GPO10	
pr1_pru<n>_pru_r30_11	GPO11	
pr1_pru<n>_pru_r30_12	GPO12	
pr1_pru<n>_pru_r30_13	GPO13	
pr1_pru<n>_pru_r30_14	GPO14	
pr1_pru<n>_pru_r30_15	GPO15	
pr1_pru<n>_pru_r30_16	GPO16	
pr1_pru<n>_pru_r30_17	GPO17	
pr1_pru<n>_pru_r30_18	GPO18	
pr1_pru<n>_pru_r30_19	GPO19	
pr1_pru<n>_pru_r30_20	GPO20	
pr1_pru<n>_pru_r30_21	GPO21	
pr1_pru<n>_pru_r30_22	GPO22	
pr1_pru<n>_pru_r30_23	GPO23	
pr1_pru<n>_pru_r30_24	GPO24	
pr1_pru<n>_pru_r30_25	GPO25	
pr1_pru<n>_pru_r30_26	GPO26	
pr1_pru<n>_pru_r30_27	GPO27	
pr1_pru<n>_pru_r30_28	GPO28	
pr1_pru<n>_pru_r30_29	GPO29	
pr1_pru<n>_pru_r30_30	GPO30	
pr1_pru<n>_pru_r30_31	GPO31	

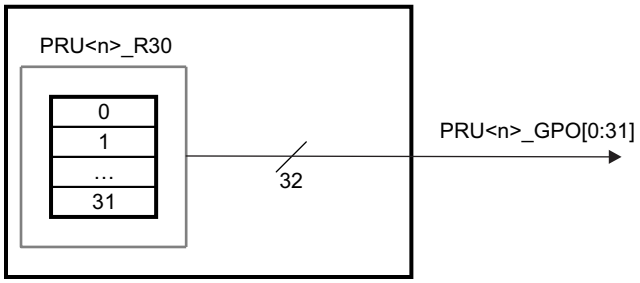
NOTE: Some devices may not pin out all 32 bits of R30. For which pins are available on this device, see Section 4.2.3, *PRU-ICSS Pin List*. See the device's datasheet for device-specific pin mapping.

4.4.1.2.4.1 Direct Output

The pru<n>_r30 [0:31] bits of the internal PRU register file are mapped to device-level general purpose output pins (PRU<n>_GPO [0:31]). In GPO Direct Output mode, pru<n>_r30 [0:31] feed directly to PRU<n>_GPO [0:31]. There are 32 possible general purpose outputs per PRU core; however, some devices may not pin out all of these signals. See the device-specific datasheet for device specific pin mapping.

NOTE: R30 is not initialized after reset. To avoid unintended output signals, R30 should be initialized before pinmux configuration of PRU signals.

Figure 4-11. PRU R30 (GPO) Direct Output Mode Block Diagram



4.4.1.2.4.2 Shift Out

In shift out mode, data is shifted out of pru<n>_r30[0] (on PRU<n>_DATAOUT) on every rising edge of pru<n>_r30[1] (PRU<n>_CLOCKOUT). The shift rate is controlled by the effective divisor of two cascaded dividers applied to the op_clk. These cascaded dividers can each be configured through the PRU-ICSS CFG register space to a value of {1, 1.5, …, 16}. Table 4-17 shows sample effective clock values and the divisor values that can be used to generate these clocks. The shift out clock is a free-running clock that is always running internally. This clock will be output to PRU<n>_CLOCKOUT (or pr1_pru<n>_pru_r30_1) when the PRU GPO mode is set to shift out mode.

Table 4-17. Effective Clock Values

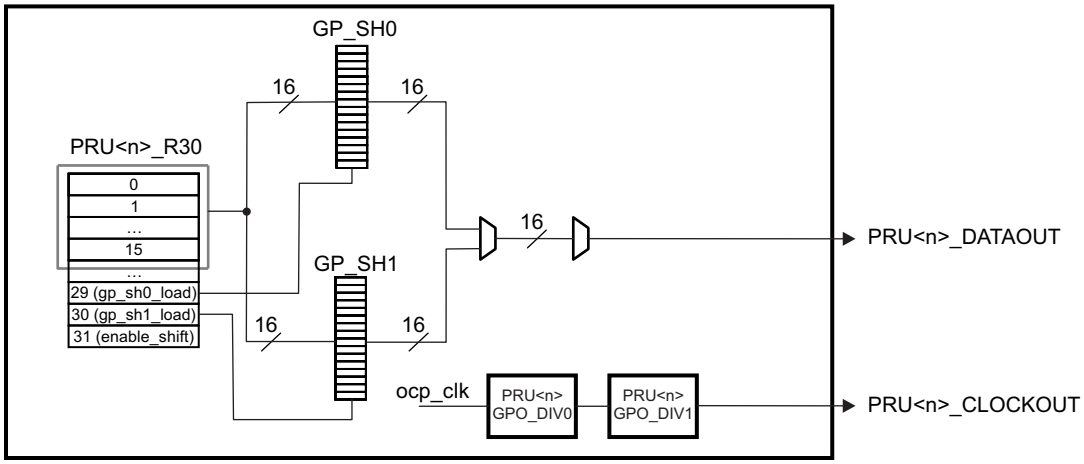
Generated Clock	PRU<n>_GPO_DIV0	PRU<n>_GPO_DIV1
8 MHz	12.5 (0x17)	2 (0x02)
10 MHz	10 (0x12)	2 (0x02)
16 MHz	16 (0x1e)	1 (0x00)
20 MHz	10 (0x12)	1 (0x00)

Shift out mode uses two 16-bit shadow registers (gpo_sh0 and gpo_sh1) to support ping-pong buffers. Each shadow register has independent load controls programmable through pru<n>_r30[29:30] (PRU<n>_LOAD_GPO_SH [0:1]). While PRU<n>_LOAD_GPO_SH [0/1] is set, the contents of pru<n>_r31[0:15] are loaded into gpo_sh0/1.

NOTE: If any device-level pins mapped to pru<n>_r30 [2:15] are configured for the pru<n>_r30 [2:15] pinmux mode, then these pins will reflect the shadow register value written to pru<n>_r30. Any pin configured for a different pinmux setting will not reflect the shadow register value written to pru<n>_r30.

The data shift will start from the lsb of gpo_sh0 when pru<n>_r30[31] (PRU<n>_ENABLE_SHIFT) is set. Note that if no new data is loaded into gpo_shn<n> after shift operation, the shift operation will continue looping and shifting out the pre-loaded data. When PRU<n>_ENABLE_SHIFT is cleared, the shift operation will finish shifting out the current shadow register, stop, and reset.

Figure 4-12. PRU R30 (GPO) Shift Out Mode Block Diagram



Follow these steps to use the GPO shift out mode:

Step One: Initialization

1. Load 16-bits of data into gpo_sh0:
   (a) Set R30[29] = 1 (PRU<n>_LOAD_GPO_SH0)
   (b) Load data in R30[15:0]
   (c) Clear R30[29] to turn off load controller
2. Load 16-bits of data into gpo_sh1:
   (a) Set R30[30] = 1 (PRU<n>_LOAD_GPO_SH1)
   (b) Load data in R30[15:0]
   (c) Clear R30[30] to turn off load controller
3. Start shift operation:
   (a) Set R30[31] = 1 (PRU<n>_ENABLE_SHIFT)

Step Two: Shift Loop

1. Monitor when a shadow register has finished shifting out data and can be loaded with new data:
   (a) Poll PRU<n>_GPI_SH_SEL bit of the GPCFG<n> register
   (b) Load new 16-bits of data into gpo_sh0 if PRU<n>_GPI_SH_SEL = 1
   (c) Load new 16-bits of data into gpo_sh1 if PRU<n>_GPI_SH_SEL = 0
2. If more data to be shifted out, loop to Shift Loop
3. If no more data, exit loop

Exit:

1. End shift operation:
   (a) Clear R30[31] to turn off shift operation

NOTE: Until the shift operation is disabled, the shift loop will continue looping and shifting out the pre-loaded data if no new data has been loaded into gpo_sh<n>.

4.4.1.3 Multiplier With Optional Accumulation (MPY/MAC)

Each PRU core has a designated unsigned multiplier with optional accumulation (MPY/MAC). The MAC has two modes of operation: Multiply Only or Multiply and Accumulate. The MAC is directly connected with the PRU internal registers R25–R29 and uses the broadside load/store PRU interface and XFR instructions to both control the mode of the MAC and import the multiplication results into the PRU.

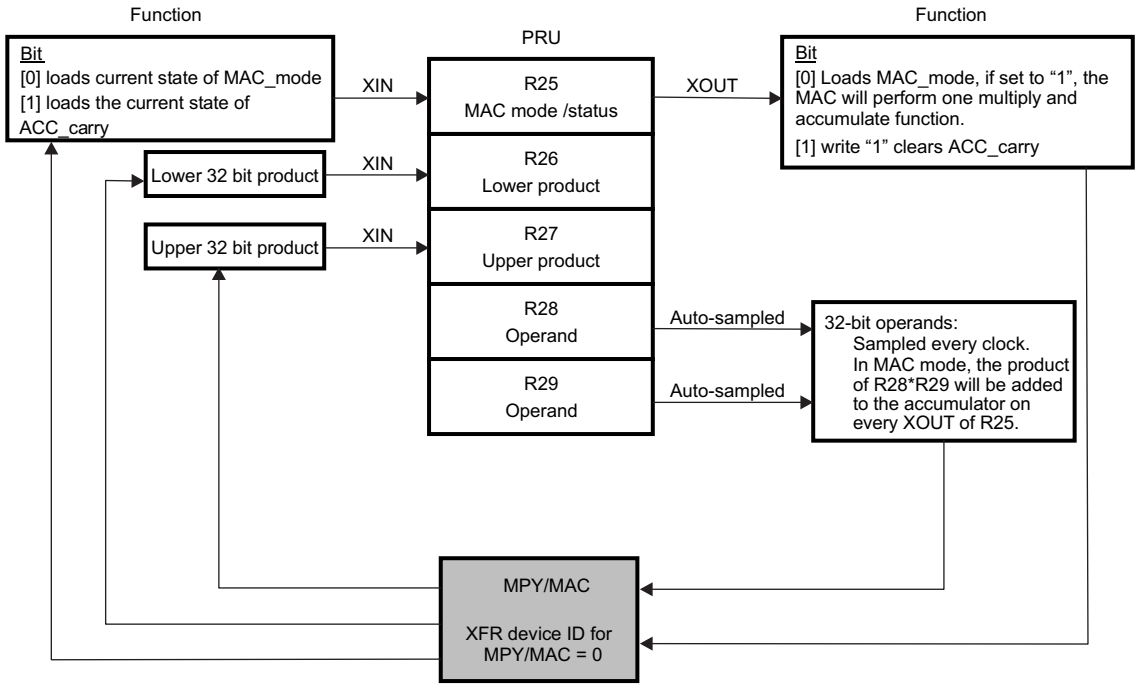
4.4.1.3.1 Features

- Configurable Multiply Only and Multiply and Accumulate functionality via PRU register R25
- 32-bit operands with direct connection to PRU registers R28 and R29
- 64-bit result (with carry flag) with direct connection to PRU registers R26 and R27
- PRU broadside interface and XFR instructions (XIN, XOUT) allow for importing multiplication results and initiating accumulate function

4.4.1.3.2 PRU and MPY/MAC Interface

The MAC directly connects with the PRU internal registers R25–R29 through use of the PRU broadside interface and XFR instructions. Figure 4-13 shows the functionality of each register.

Figure 4-13. Integration of the PRU and MPY/MAC



The XFR instructions (XIN and XOUT) are used to load/store register contents between the PRU core and the MAC. These instructions define the start, size, direction of the operation, and device ID. The device ID number corresponding to the MPY/MAC is shown in Table 4-18.

Table 4-18. MPY/MAC XFR ID

Device ID	Function
0	Selects MPY/MAC

The PRU register R25 is mapped to the MAC_CTRL_STATUS register (Table 4-19). The MAC's current status (MAC_mode and ACC_carry states) is loaded into R25 using the XIN command on R25. The PRU sets the MAC's mode and clears the ACC_carry using the XOUT command on R25.

Table 4-19. MAC_CTRL_STATUS Register (R25) Field Descriptions

Bit	Field	Value	Description
7-2	Reserved		Reserved
1	ACC_carry		Write 1 to clear
		0	64-bit accumulator carry has not occurred
		1	64-bit accumulator carry occurred
0	MAC_mode	0	Accumulation mode disabled and accumulator is cleared
		1	Accumulation mode enabled

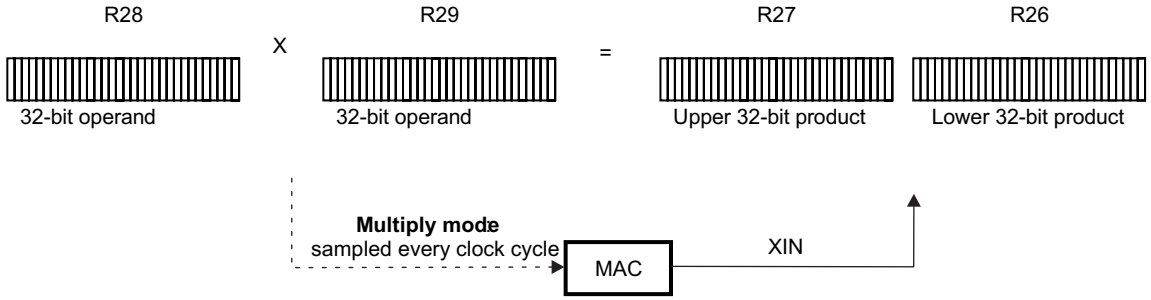
The two 32-bit operands for the multiplication are loaded into R28 and R29. These registers have a direction connection with the MAC, and the MAC samples these registers every clock cycle. Note, XOUT is not required to load the MAC. In multiply and accumulate mode, the product of R28*R29 is added to the accumulator on every XOUT of R25.

The product from the MAC is linked to R26 (lower 32 bits) and R27 (upper 32 bits). The product is loaded into register R26 and R27 using XIN.

4.4.1.3.2.1 Multiply-Only Mode (Default State), MAC_mode = 0

On every clock cycle, the MAC multiplies the contents of R28 and R29.

Figure 4-14. Multiply-Only Mode Functional Diagram



The following steps are performed by the PRU firmware for multiply-only mode:

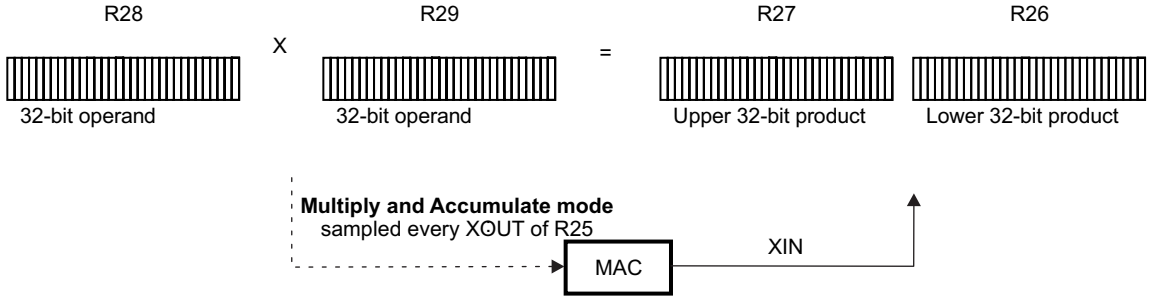
1. Enable multiply only MAC_mode.
   (a) Clear R25[0] for multiply only mode.
   (b) Store MAC_mode to MAC using XOUT instruction with the following parameters:
   - Device ID = 0
   - Base register = R25
   - Size = 1
2. Load operands into R28 and R29.
3. Delay at least 1 PRU cycle before executing XIN in step 4.
4. Load product into PRU using XIN instruction on R26, R27.

Repeat steps 2 through 4 for each new operand.

4.4.1.3.2.2 Multiply and Accumulate Mode, MAC_mode = 1

On every XOUT R25_reg[7:0] transaction, the MAC multiplies the contents of R28 and R29, adds the product to its accumulated result, and sets ACC_carry if an accumulation overflow occurs.

Figure 4-15. Multiply and Accumulate Mode Functional Diagram



The following steps are performed by the PRU firmware for multiply and accumulate mode:

1. Enable multiply and accumulate MAC_mode.
   (a) Set R25[1:0] = 1 for accumulate mode.
   (b) Store MAC_mode to MAC using XOUT instruction with the following parameters:
   - Device ID = 0
   - Base register = R25
   - Size = 1
2. Clear accumulator and carry flag.
   (a) Set R25[1:0] = 3 to clear accumulator (R25[1]=1) and preserve accumulate mode (R25[0]=1).
   (b) Store accumulator to MAC using XOUT instruction on R25.
3. Load operands into R28 and R29.
4. Multiply and accumulate, XOUT R25[1:0] = 1
   Repeat step 4 for each multiply and accumulate using same operands.
   Repeat step 3 and 4 for each multiply and accumulate for new operands.
5. Load the accumulated product into R26, R27 and the ACC_carry status into R25 using the XIN instruction.

Note: Steps one and two are required to set the accumulator mode and clear the accumulator and carry flag.

4.4.1.4 PRU0/1 Scratch Pad

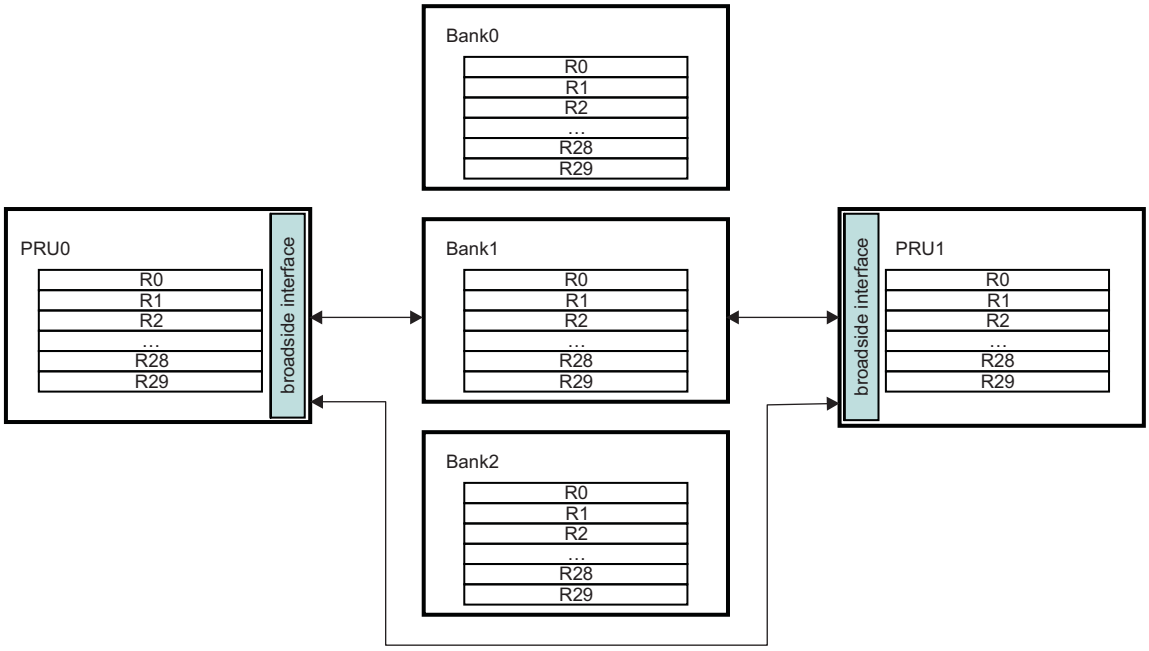
The PRU-ICSS supports a scratch pad with three independent banks accessible by the PRU cores. The PRU cores interact with the scratch pad through broadside load/store PRU interface and XFR instructions. The scratch pad can be used as a temporary place holder for the register contents of the PRU cores. Direct connection between the PRU cores is also supported for transferring register contents directly between the cores.

4.4.1.4.1 Features

The PRU-ICSS scratch pad supports the following features:

- Three scratch pad banks of 30, 32-bit registers (R29:0)
- Flexible load/store options
  - User-defined start byte and length of the transfer
  - Length of transfer ranges from one byte of a register to the entire register content (R29 to R0)
  - Simultaneous transactions supported between PRU0 ↔ Bank<n> and PRU1 ↔ Bank<m>
  - Direct connection of PRU0 → PRU1 or PRU1 → PRU0 for all registers R29–R0
- XFR instructions operate in one clock cycle
- Optional XIN/XOUT shift functionality allows remapping of registers (R<n> → R<m>) during load/store operation

Figure 4-16. Integration of PRU and Scratch Pad



4.4.1.4.2 Implementations and Operations

XFR instructions are used to load/store register contents between the PRU cores and the scratch pad banks. These instructions define the start, size, direction of the operation, and device ID. The device ID corresponds to the external source or destination (either a scratch pad bank or the other PRU core). The device ID numbers are shown in Table 4-20. Note the direct connect mode (device ID 14) can be used to synchronize the PRU cores. This mode requires the transmitting PRU core to execute XOUT and the receiving PRU core to execute XIN.

Table 4-20. Scratch Pad XFR ID

Device ID	Function
10	Selects Bank0
11	Selects Bank1
12	Selects Bank2
13	Reserved
14	Selects other PRU core (Direct connect mode)

A collision occurs when two XOUT commands simultaneously access the same asset or device ID. Table 4-21 shows the priority assigned to each operation when a collision occurs. In direct connect mode (device ID 14), any PRU transaction will be terminated if the stall is greater than 1024 cycles. This will generate the event pr1_xfr_timeout that is connected to INTC.

Table 4-21. Scratch Pad XFR Collision & Stall Conditions

Operation	Collision Handling
PRU<n> XOUT (→) bank[j]	If both PRU cores access the same bank simultaneously, PRU0 is given priority. PRU1 will temporarily stall until the PRU0 operation completes.
PRU<n> XOUT (→) PRU<m>	Direct Connect mode requires the transmitting core (PRU<n>) to execute XOUT and the receiving core (PRU<m>) to execute XIN. If PRU<n> executes XOUT before PRU<m> executes XIN, then PRU<n> will stall until either PRU<m> executes XIN or the stall is greater than 1024 cycles.

Table 4-21. Scratch Pad XFR Collision & Stall Conditions (continued)

PRU<m> XIN (←) PRU<n>	Direct Connect mode requires the transmitting core (PRU<n>) to execute XOUT and the receiving core (PRU<m>) to execute XIN. If PRU<m> executes XIN before PRU<n> executes XOUT, then PRU<m> will stall until either PRU<n> executes XOUT or the stall is greater than 1024 cycles.

4.4.1.4.2.1 Optional XIN/XOUT Shift

The optional XIN/XOUT shift functionality allows register contents to be remapped or shifted within the destination's register space. For example, the contents of PRU0 R6-R8 could be remapped to Bank1 R10-12. The XIN/XOUT shift feature is not supported for direct connect mode, only for transfers between a PRU core and scratch pad bank.

The shift feature is enabled or disabled through the SPP register of the PRU-ICSS CFG register space. When enabled, R0[4:0] (internal to the PRU) defines the number of 32-bit registers in which content is shifted in the scratch pad bank. Note that scratch pad banks do not have registers R30 or R31.

4.4.1.4.2.2 Example Scratch Pad Operations

The following PRU firmware examples demonstrate the shift functionality. Note these assume the SHIFT_EN bit of the SPP register of the PRU-ICSS CFG register space has been set.

XOUT Shift By 4 Registers

Store R4:R7 to R8:R11 in bank0:

- Load 4 into R0.b0
- XOUT using the following parameters:
  - Device ID = 10
  - Base register = R4
  - Size = 16

XOUT Shift By 9 Registers, With Wrap Around

Store R25:R29 to R4:R8 in bank1:

- Load 9 into R0.b0
- XOUT using the following parameters:
  - Device ID = 11
  - Base register = R25
  - Size = 20

XIN Shift By 10 Registers

Store R14:R16 from bank2 to R4:R6:

- Load 10 into R0.b0
- XIN using the following parameters:
  - Device ID = 12
  - Base register = R4
  - Size = 12

4.4.2 Interrupt Controller (INTC)

The PRU-ICSS interrupt controller (INTC) is an interface between interrupts coming from different parts of the system (referred to as system events; see Section 4.4.2.2) and the PRU-ICSS interrupt interface.

The PRU-ICSS INTC has the following features:

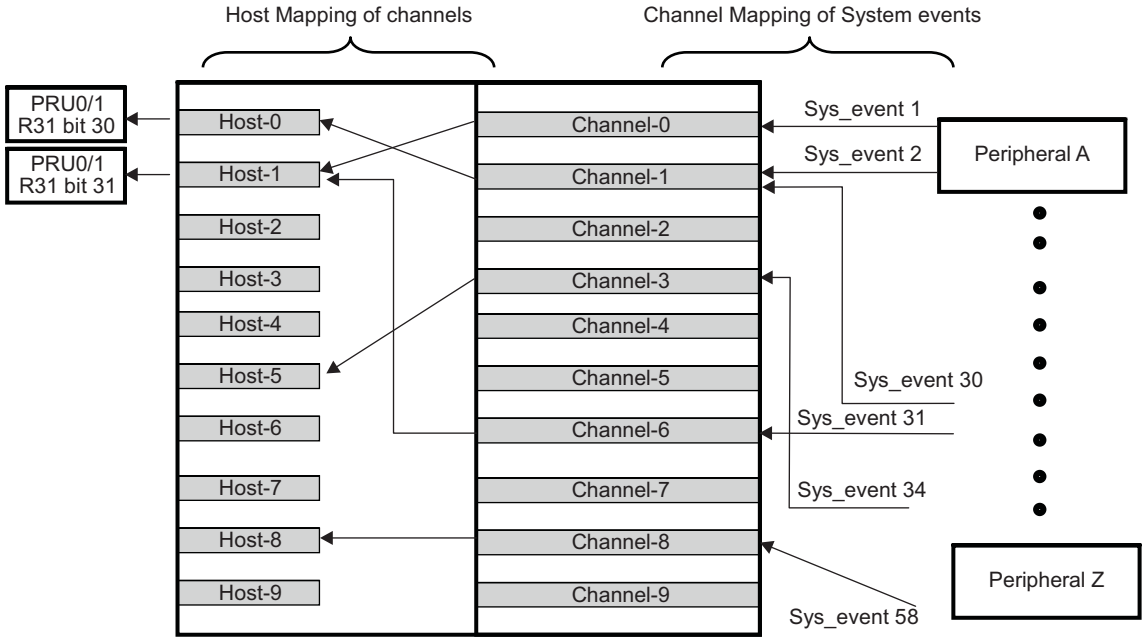
- Capturing up to 64 System Events
- Supports up to 10 interrupt channels.
- Generation of 10 Host Interrupts
  - 2 Host Interrupts for the PRUs.
  - 8 Host Interrupts exported from the PRU-ICSS for signaling the ARM interrupt controllers.
- Each system event can be enabled and disabled.
- Each host event can be enabled and disabled.
- Hardware prioritization of events.

4.4.2.1 INTC Overview

The PRU-ICSS INTC supports up to 64 system events from different peripherals and PRUs to be mapped to 10 channels inside the INTC (see Figure 4-17). Interrupts from these 10 channels are further mapped to 10 Host Interrupts.

- Any of the 64 system events can be mapped to any of the 10 channels.
- Multiple interrupts can be mapped to a single channel.
- An interrupt should not be mapped to more than one channel.
- Any of the 10 channels can be mapped to any of the 10 host interrupts. It is recommended to map channel “x” to host interrupt “x”, where x is from 0 to 9
- A channel should not be mapped to more than one host interrupt
- For channels mapping to the same host interrupt, lower number channels have higher priority.
- For interrupts on same channel, priority is determined by the hardware interrupt number. The lower the interrupt number, the higher the priority.
- Host Interrupt 0 is connected to bit 30 in register 31 of PRU0 and PRU1.
- Host Interrupt 1 is connected to bit 31 in register 31 for PRU0 and PRU1.
- Host Interrupts 2 through 9 exported from PRU-ICSS for signaling ARM interrupt controllers or other machines like EDMA.

Figure 4-17. Interrupt Controller Block Diagram



4.4.2.2 PRU-ICSS System Events

The PRU-ICSS system events can be found in Table 4-22.

Table 4-22. PRU-ICSS System Events

Int Number	Signal Name (Standard Mode)	Source	Signal Name (MII_RT Mode)[1][2]
63	tpcc_int_pend_po1	TPCC (EDMA)	
62	tpcc_errint_pend_po	TPCC (EDMA)	
61	tptc_erint_pend_po	TPTC0 (EDMA)	
60	initiator_sinterrupt_q_n1	Mbox0 - mail_u1_irq (mailbox interrupt for pru0)	
59	initiator_sinterrupt_q_n2	Mbox0 - mail_u2_irq (mailbox interrupt for pru1)	
58	Emulation Suspend Signal (software use)	Debugss	
57	POINTRPEND1	GPIO0	
56	pwm_trip_zone	eHRPWM0/eHRPWM1/eHRPWM2	
55	mcasp_x_intr_pend	McASP0 Tx	(pr1_mii1_col & pr1_mii1_txen) (external)
54	mcasp_r_intr_pend	McASP0 Rx	PRU-ICSS0 PRU1_RX_EOF
53	gen_intr_pend	ADC_TSC	PRU-ICSS0 MDIO_MII_LINK[1]
52	nirq	UART2	PRU-ICSS0 PORT1_TX_OVERFLOW
51	nirq	UART0	PRU-ICSS0 PORT1_TX_UNDERFLOW
50	c0_rx_thresh_pend	3PGSW (GEMAC)	PRU-ICSS0 PRU1_RX_OVERFLOW
49	c0_rx_pend	3PGSW (GEMAC)	PRU-ICSS0 PRU1_RX_NIBBLE_ODD
48	c0_tx_pend	3PGSW (GEMAC)	PRU-ICSS0 PRU1_RX_CRC
47	c0_misc_pend	3PGSW (GEMAC)	PRU-ICSS0 PRU1_RX_SOF
46	epwm_intr_intr_pend	eHRPWM1	PRU-ICSS0 PRU1_RX_SFD
45	eqep_intr_intr_pend	eQEP0	PRU-ICSS0 PRU1_RX_ERR32
44	SINTERRUPTN	McSPI0	PRU-ICSS0 PRU1_RX_ERR
43	epwm_intr_intr_pend	eHRPWM0	(pr1_mii0_col & pr1_mii0_txen) (external)
42	ecap_intr_intr_pend	eCAP0	PRU-ICSS0 PRU0_RX_EOF
41	POINTRPEND	I2C0	PRU-ICSS0 MDIO_MII_LINK[0]
40	dcan_intr	DCAN0	PRU-ICSS0 PORT0_TX_OVERFLOW
39	dcan_int1	DCAN0	PRU-ICSS0 PORT0_TX_UNDERFLOW
38	dcan_uerr	DCAN0	PRU-ICSS0 PRU0_RX_OVERFLOW
37	epwm_intr_intr_pend	eHRPWM2	PRU-ICSS0 PRU0_RX_NIBBLE_ODD
36	ecap_intr_intr_pend	eCAP2	PRU-ICSS0 PRU0_RX_CRC
35	ecap_intr_intr_pend	eCAP1	PRU-ICSS0 PRU0_RX_SOF
34	mcasp_r_intr_pend	McASP1 Rx	PRU-ICSS0 PRU0_RX_SFD
33	mcasp_x_intr_pend	McASP1 Tx	PRU-ICSS0 PRU0_RX_ERR32
32	nirq	UART1	PRU-ICSS0 PRU0_RX_ERR

(1) MII_RT mode is selected through the MII_RT register in the PRU-ICSS0 CFG register space.
(2) Signals 63-56 and 31-0 for MII_RT Mode are the same as for Standard Mode.

Table 4-22. PRU-ICSS System Events (continued)

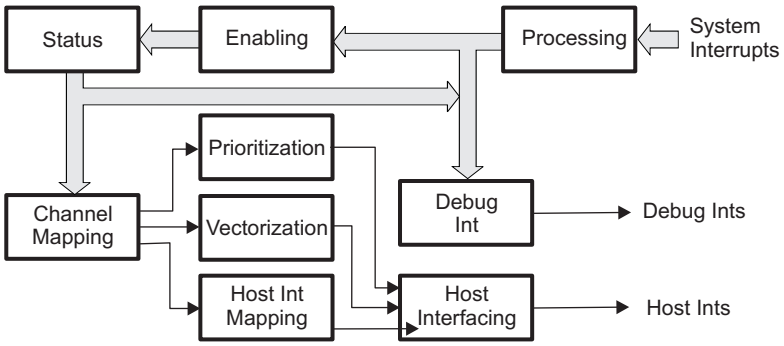
Int Number	Signal Name (Standard Mode)	Source	Signal Name (MII_RT Mode)[1][2]
31	pr1_pru_mst_intr[15]_intr_req	pru0 or pru1	PRU-ICSS0 Internal Interrupts
30	pr1_pru_mst_intr[14]_intr_req	pru0 or pru1	
29	pr1_pru_mst_intr[13]_intr_req	pru0 or pru1	
28	pr1_pru_mst_intr[12]_intr_req	pru0 or pru1	
27	pr1_pru_mst_intr[11]_intr_req	pru0 or pru1	
26	pr1_pru_mst_intr[10]_intr_req	pru0 or pru1	
25	pr1_pru_mst_intr[9]_intr_req	pru0 or pru1	
24	pr1_pru_mst_intr[8]_intr_req	pru0 or pru1	
23	pr1_pru_mst_intr[7]_intr_req	pru0 or pru1	
22	pr1_pru_mst_intr[6]_intr_req	pru0 or pru1	
21	pr1_pru_mst_intr[5]_intr_req	pru0 or pru1	
20	pr1_pru_mst_intr[4]_intr_req	pru0 or pru1	
19	pr1_pru_mst_intr[3]_intr_req	pru0 or pru1	
18	pr1_pru_mst_intr[2]_intr_req	pru0 or pru1	
17	pr1_pru_mst_intr[1]_intr_req	pru0 or pru1	
16	pr1_pru_mst_intr[0]_intr_req	pru0 or pru1	
15	pr1_ecap_intr_req	PRU-ICSS eCAP	
14	sync0_out_pend	PRU-ICSS IEP	
13	sync1_out_pend	PRU-ICSS IEP	
12	latch0_in (input to PRU-ICSS)	PRU-ICSS IEP	
11	latch1_in (input to PRU-ICSS)	PRU-ICSS IEP	
10	pdi_wd_exp_pend	PRU-ICSS IEP	
9	pd_wd_exp_pend	PRU-ICSS IEP	
8	pr1_digio_event_req	PRU-ICSS IEP	
7	pr1_iep_tim_cap_cmp_pend	PRU-ICSS IEP	
6	pr1_uart_uint_intr_req	PRU-ICSS UART	
5	pr1_uart_utxevt_intr_req	PRU-ICSS UART	
4	pr1_uart_urxevt_intr_req	PRU-ICSS UART	
3	pr1_xfr_timeout	PRU-ICSS Scratch Pad	
2	pr1_pru1_r31_status_cnt16	PRU-ICSS PRU1 (Shift Capture)	
1	pr1_pru0_r31_status_cnt16	PRU-ICSS PRU0 (Shift Capture)	
0	pr1_parity_err_intr_pend	PRU-ICSS Parity Logic	

4.4.2.3 INTC Methodology

The INTC module controls the system event mapping to the host interrupt interface. System events are generated by the device peripherals or PRUs. The INTC receives the system events and maps them to internal channels. The channels are used to group interrupts together and to prioritize them. These channels are then mapped onto the host interrupts. Interrupts from the system side are active high in polarity. They are also pulse type of interrupts.

The INTC encompasses many functions to process the system events and prepare them for the host interface. These functions are: processing, enabling, status, channel mapping, host interrupt mapping, prioritization, and host interfacing. Figure 4-18 illustrates the flow of system events through the functions to the host. The following subsections describe each part of the flow.

Figure 4-18. Flow of System Events to Host



4.4.2.3.1 Interrupt Processing

This block does the following tasks:

- Synchronization of slower and asynchronous interrupts
- Conversion of polarity to active high
- Conversion of interrupt type to pulse interrupts

After the processing block, all interrupts will be active high pulses.

4.4.2.3.2 Interrupt Enabling

The next stage of INTC is to enable system events based on programmed settings. The following sequence is to be followed to enable interrupts:

- Enable required system events: System events that are required to get propagated to host are to be enabled individually by writing to IDX field in the system event enable indexed set register (EISR). The event to enable is the index value written. This sets the Enable Register bit of the given index.
- Enable required host interrupts: By writing to the IDX field in the host interrupt enable indexed set register (HIEISR), enable the required host interrupts. The host interrupt to enable is the index value written. This enables the host interrupt output or triggers the output again if that host interrupt is already enabled.
- Enable all host interrupts: By setting the EN bit in the global enable register (GER) to 1, all host interrupts will be enabled. Individual host interrupts are still enabled or disabled from their individual enables and are not overridden by the global enable.

4.4.2.3.3 Interrupt Status Checking

The next stage is to capture which system events are pending. There are two kinds of pending status: raw status and enabled status. Raw status is the pending status of the system event without regards to the enable bit for the system event. Enabled status is the pending status of the system events with the enable bits active. When the enable bit is inactive, the enabled status will always be inactive. The enabled status of system events is captured in system event status enabled/clear registers (SECR1-SECR2).

Status of system event 'N' is indicated by the Nth bit of SECR1-SECR2. Since there are 64 system events, two 32-bit registers are used to capture the enabled status of events. The pending status reflects whether the system event occurred since the last time the status register bit was cleared. Each bit in the status register can be individually cleared.

4.4.2.3.4 Interrupt Channel Mapping

The INTC has 10 internal channels to which enabled system events can be mapped. Channel 0 has highest priority and channel 9 has the lowest priority. Channels are used to group the system events into a smaller number of priorities that can be given to a host interface with a very small number of interrupt inputs.

When multiple system events are mapped to the same channel their interrupts are ORed together so that when one or more is active the output is active. The channel map registers (CMRm) define the channel for each system event. There is one register per 4 system events; therefore, there are16 channel map registers for a system of 64 events. The channel for each system event can be set using these registers.

4.4.2.3.4.1 Host Interrupt Mapping

The hosts can be the PRUs or ARM CPU. The 10 channels from the INTC can mapped to any of the 10 Host events. The Host map registers (HMR*m*) define the channel for each system event. There is one register per 4 channels; therefore, there are 3 host map registers for 10 channels. When multiple channels are mapped to the same host interrupt, then prioritization is done to select which interrupt is in the highest-priority channel and which should be sent first to the host.

4.4.2.3.4.2 Interrupt Prioritization

The next stage of the INTC is prioritization. Since multiple events can feed into a single channel and multiple channels can feed into a single host interrupt, it is to read the status of all system events to determine the highest priority event that is pending. The INTC provides hardware to perform this prioritization with a given scheme so that software does not have to do this. There are two levels of prioritizations:

- The first level of prioritization is between the active channels for a host interrupt. Channel 0 has the highest priority and channel 9 has the lowest. So the first level of prioritization picks the lowest numbered active channel.
- The second level of prioritization is between the active system events for the prioritized channel. The system event in position 0 has the highest priority and system event 63 has the lowest priority. So the second level of prioritization picks the lowest position active system event.

This is the final prioritized system event for the host interrupt and is stored in the global prioritized index register (GPIR). The highest priority pending event with respect to each host interrupt can be obtained using the host interrupt prioritized index registers (HIPIRn).

4.4.2.3.5 Interrupt Nesting

The INTC can also perform a nesting function in its prioritization. Nesting is a method of enabling certain interrupts (usually higher-priority interrupts) when an interrupt is taken so that only those desired interrupts can trigger to the host while it is servicing the current interrupt. The typical usage is to nest on the current interrupt and disable all interrupts of the same or lower priority (or channel). Then the host will only be interrupted from a higher priority interrupt.

The nesting is done in one of three methods:

1. Nesting for all host interrupts, based on channel priority: When an interrupt is taken, the nesting level is set to its channel priority. From then, that channel priority and all lower priority channels will be disabled from generating host interrupts and only higher priority channels are allowed. When the interrupt is completely serviced, the nesting level is returned to its original value. When there is no interrupt being serviced, there are no channels disabled due to nesting. The global nesting level register (GNLR) allows the checking and setting of the global nesting level across all host interrupts. The nesting level is the channel (and all of lower priority channels) that are nested out because of a current interrupt.
2. Nesting for individual host interrupts, based on channel priority: Always nest based on channel priority for each host interrupt individually. When an interrupt is taken on a host interrupt, then, the nesting level is set to its channel priority for just that host interrupt, and other host interrupts do not have their nesting affected. Then for that host interrupt, equal or lower priority channels will not interrupt the host but may on other host interrupts if programmed. When the interrupt is completely serviced the nesting level for the host interrupt is returned to its original value. The host interrupt nesting level registers (HINLR1 and HINLR2) display and control the nesting level for each host interrupt. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.
3. Software manually performs the nesting of interrupts. When an interrupt is taken, the software will disable all the host interrupts, manually update the enables for any or all the system events, and then re-enables all the host interrupts. This now allows only the system events that are still enabled to trigger to the host. When the interrupt is completely serviced the software must reverse the changes to re-enable the nested out system events. This method requires the most software interaction but gives the most flexibility if simple channel based nesting mechanisms are not adequate.

4.4.2.3.6 Interrupt Status Clearing

After servicing the event (after execution of the ISR), event status is to be cleared. If a system event status is not cleared, then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. It is also essential to clear all system events before the PRU is halted as the PRU does not power down unless all the event status are cleared. For clearing the status of an event, whose event number is N, write a 1 to the Nth bit position in the system event status enabled/clear registers (SECR1-SECR2). System event N can also be cleared by writing the value N into the system event status indexed clear register (SICR).

4.4.2.4 Interrupt Disabling

At any time, if any event is not to be propagated to the host, then that event should be disabled. For disabling an event whose event number is N, write a 1 to the Nth bit in the system event enable clear registers (ECR1-ECR2). System event N can also be disabled by writing the value N in the system event enable indexed clear register (EICR).

4.4.2.5 INTC Basic Programming Model

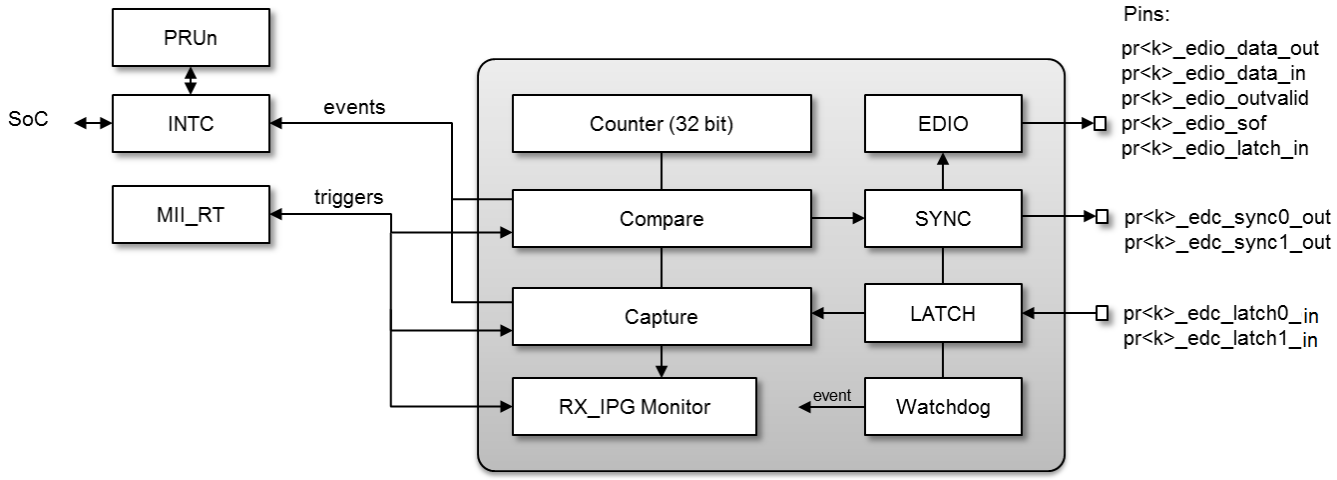
Follow these steps to configure the interrupt controller.

1. Set polarity and type of system event through the System Event Polarity Registers (SIPR1 and SPIR2) and the System Event Type Registers (SITR1 and SITR2). Polarity of all system events is always high. Type of all system events is always pulse.
2. Map system event to INTC channel through CHANMAP registers.
3. Map channel to host interrupt through HOSTMAP registers. Recommend channel “x” be mapped to host interrupt “x”.
4. Clear system event by writing 1s to SECR registers.
5. Enable host interrupt by writing index value to HIER register.
6. Enable interrupt nesting if desired.
7. Globally enable all interrupts through GER register.

4.4.3 Industrial Ethernet Peripheral (IEP)

The industrial ethernet peripheral (IEP) performs hardware work required for industrial ethernet functions. The IEP module features an industrial ethernet timer with 8 compare events, industrial ethernet sync generator and latch capture, industrial ethernet watchdog timer, and a digital I/O port (DIGIO). The IEP functional block diagram is shown in Figure 4-19.

Figure 4-19. Industrial Ethernet Peripheral Block Diagram



4.4.3.1 IEP Clock Source

The IEP has a selectable module input clock. The clock source is selected by the state of the IEPCLK.OCP_EN bit within the PRU-ICSS CFG register space. Two clock sources are supported for the IEP input clock:

- iep_clk (default): Runs at 200 MHz
- ocp_clk

Switching from iep_clk to ocp_clk is done by writing 1 to the IEPCLK.OCP_EN bit. This is a one time configuration step before enabling the IEP function. Switching back from ocp_clk to iep_clk is only supported through a hardware reset of the PRU-ICSS.

4.4.3.2 Industrial Ethernet Timer

The industrial ethernet timer is a simple 32-bit timer. This timer is intended for use by industrial ethernet functions but can also be leveraged as a generic timer in other applications.

4.4.3.2.1 Features

The industrial ethernet timer supports the following features:

- One master 32-bit count-up counter with an overflow status bit
  - Runs on iep_clk or ocp_clk
  - Write 1 to clear status
  - Supports a programmable increment value from 1 to 16 (default 5)
  - An optional compensation method allows the increment value to apply a compensation increment value from 1 to 16, counting up to 2^24 iep_clk/ocp_clk events
- Ten 32-bit capture registers (CAPR[5:0], CAPR[7:6], CAPF[7:6])
  - Eight capture inputs with optional synchronous or asynchronous mode
    - Six rise-only capture inputs (CAPR[5:0])
    - Two rise-and-fall capture inputs:
      - CAPR[7] and CAPF[7]

        - CAPR[6] and CAPF[6]
      - One input signal will be used by two capture registers:
        - One register for rising edge
        - One register for falling edge
      - One global event (any capture event) output for interrupt generation, triggered by any capture event
- Eight 32-bit compare registers (CMP[7:0], CMP_STAT)
  - Eight status bits, write 1 to clear
  - Eight individual event outputs
  - One global event (any compare event) output for interrupt generation triggered by any compare event
- Sixteen outputs, one high level and one high pulse for each compare hit event
- CMP[0], if enabled, will reset the counter on the next iep_clk or ocp_clk cycle

4.4.3.2.2 Basic Programming Model

Follow these basic steps to configure the IEP Timer.

1. Initialize timer to known state (default values)
   (a) Disable counter (GLB_CFG.CNT_ENABLE)
   (b) Reset Count Register (CNT) by writing 0xFFFFFFFF to clear
   (c) Clear overflow status register (GLB_STS.CNT_OVF)
   (d) Clear compare status (CMP_STS)
2. Set compare values (CMP0-CMPx)
3. Enable compare event (CMP_CFG.CMP_EN)
4. Set increment value (GLB_CFG.DEFAULT_INC)
5. Set compensation value (COMPEN.COMPEN_CNT)
6. Enable counter (GLB_CFG.CNT_ENABLE)

4.4.3.3 Industrial Ethernet Mapping

Some of the capture inputs and compare registers are mapped to specific industrial ethernet functions in hardware, shown in Table 4-23. All capture inputs are mapped to industrial ethernet functions, and these inputs are not available for any other application. The cmp1 and cmp2 compare registers also function as the start time triggers for SYNC0 and SYNC1, respectively.

Table 4-23. Industrial Ethernet Timer Mode Mapping

Capture Input	IEP Line/Function
cap0, rise only	PRU0_RX_SOF
cap1, rise only	PRU0_RX_SFD
cap2, rise only	PRU1_RX_SOF
cap3, rise only	PRU1_RX_SFD
cap4, rise only	PORT0_TX_SOF
cap5, rise only	PORT1_TX_SOF
cap6, rise and fall	latch0_in, SOC IO inputs
cap7, rise and fall	latch1_in, SOC IO inputs
cmp1	For SYNC0 trigger of start time
cmp2	For SYNC1 trigger of start time; only valid in the independent mode
cmp3	For MII TX0 start trigger, if MII register TXCFG0/1[TX_EN_MODE] is enabled.

Table 4-23. Industrial Ethernet Timer Mode Mapping (continued)

Capture Input	IEP Line/Function
cmp4	For MII TX1 start trigger, if MII register TXCFG0/1[TX_EN_MODE] is enabled.

4.4.3.4 Industrial Ethernet SYNC0/SYNC1

The industrial ethernet sync block supports the generation of two synchronization signals: SYNC0 and SYNC1. SYNC0 and SYNC1 can be directly mapped to output signals (pr1_edc_sycn0_out and pr1_edc_sync1_out) for external devices to use. They can also be used for internal synchronization within the PRU-ICSS. These signals are also mapped as system events and can therefore be mapped to the ARM core's Host interrupts.

4.4.3.4.1 Features

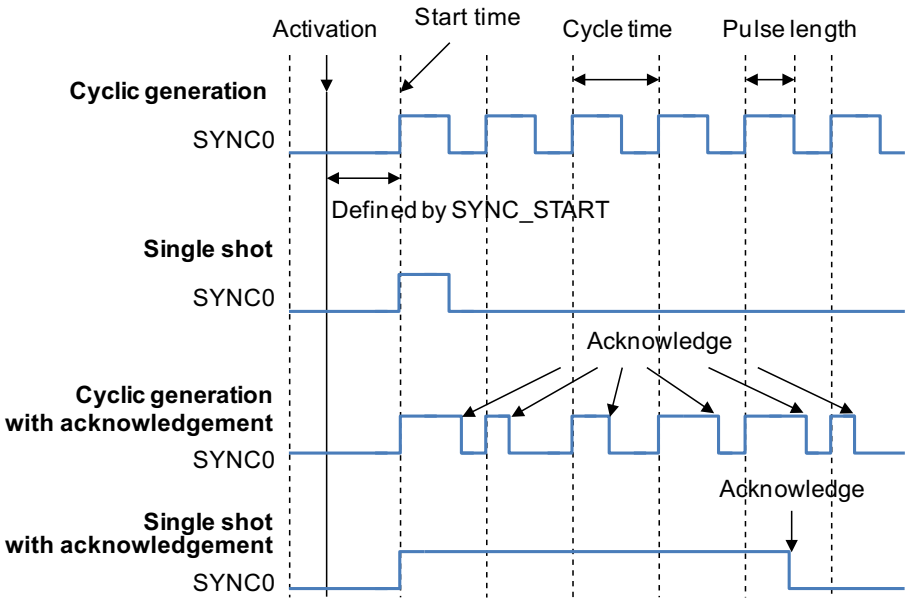
The industrial ethernet sync block supports the following features:

- Two synchronize generation signals (SYNC0, SYNC1)
  - Activation time synchronized with IEP Timer
    - CMP[1] triggers SYNC0 activation time
    - CMP[2] triggers SYNC1 activation time (only valid in the independent mode)
  - Pulse width defined by registers or ack mode (remain asserted until software acknowledged)
  - Cyclic or single-shot operation
  - Option to enable or disable sync generation
- Programmable number of clock cycles between the start of SYNC0 to the start of SYNC1

4.4.3.4.2 Sync Generation Modes

There are four modes of operation for the sync signals: cyclic mode, single shot mode, cyclic with acknowledge mode, and single shot with acknowledge mode. Figure 4-20 shows examples of these modes. The start time is set by the SYNC_START register. The cycle time is configured by the SYNC0_PERIOD register. The pulse length is defined by SYNC_PWIDTH register.

Figure 4-20. Sync Signal Generation Mode



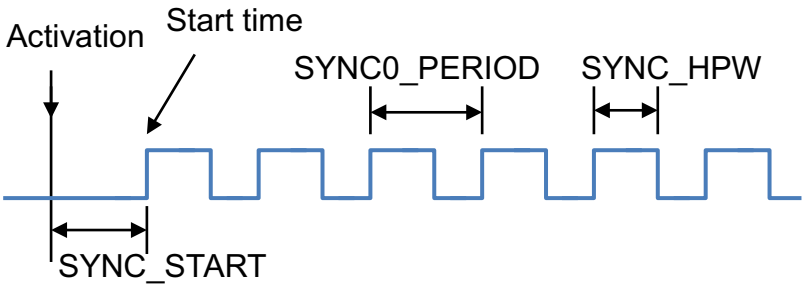
In SYNC1 dependent mode (SYNC_CTRL.sync1_ind_en = 0), SYNC1 depends on SYNC0 and the start time of the SYNC1 can be defined by the SYNC1_DELAY register. Figure 4-21 shows different examples when changing the value in the SYNC1_DELAY register. Note if the SYNC1 delay time is 0, SYNC1 reflects SYNC0.

Cyclic generation cannot be used for network time synchronized applications because only the CMP1/CMP2 hit occurs in the compensated time domain.

Figure 4-21. Examples of the Dependent Mode of SYNC1

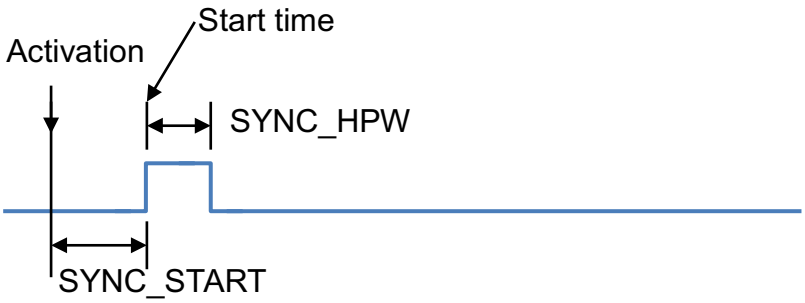
SYNC0 : **Cyclic generation**

Field	Value
SYNC0_CYCLIC_EN	1
SYNC0_ACK_EN	0



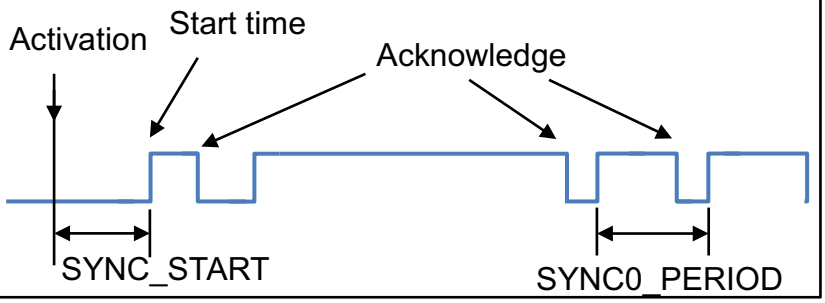
SYNC0 : **Single shot**

Field	Value
SYNC0_CYCLIC_EN	0
SYNC0_ACK_EN	0



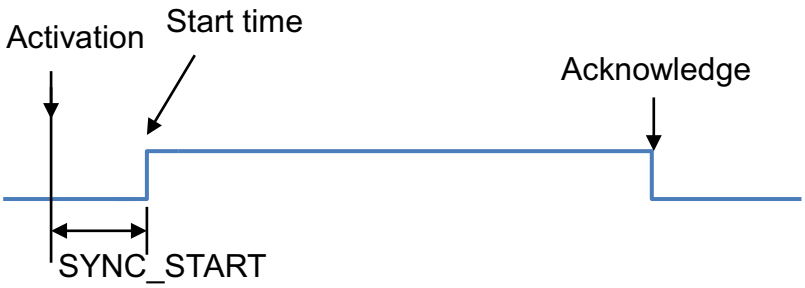
SYNC0 : **Cyclic generation with Acknowledge**

Field	Value
SYNC0_CYCLIC_EN	1
SYNC0_ACK_EN	1



SYNC0 : **Single shot with Acknowledge**

Field	Value
SYNC0_CYCLIC_EN	0
SYNC0_ACK_EN	1



4.4.3.5 Industrial Ethernet Watchdog Timer

In industrial ethernet applications, the watchdog timer (WD) is used as a safety feature to monitor process data communication and to turn off the outputs of the digital input/output (DIGIO) functional block after a set time. The WD will thereby protect the system from errors or faults by timeout or expiration. The expiration is used to initiate corrective action in order to keep the system in a safe state and restore normal operation based on configuration. Therefore, if the system is stable, the watchdog timer should be regularly reset or cleared to avoid timeout or expiration.

4.4.3.5.1 Features

The industrial ethernet watchdog timer supports the following features:

- One 32-bit pre-divider for generating a WD clock (default 100µs) based on iep_clk input
- Two 16-bit Watchdog Timers:
  - PDI_WD for Sync Managers WD, used in conjunction with digital input/output (DIGIO)
  - PD_WD for data link layer user WD, used in conjunction with data link layer or application layer interface actions

4.4.3.6 Industrial Ethernet Digital I/O (DIGIO)

The IEP Digital I/O (DIGIO) block provides dedicated I/Os intended for industrial ethernet protocols, but they can also be used as generic I/Os in other applications. The digital inputs can be sampled when specific events occur, or continuously as a raw input. Likewise, driving the digital outputs can be triggered by specific events, or controlled by software. The timing, delay cycle clocks, data sources, and data valid of the digital input and outputs are controlled by the DIGIO_CTRL and DIGIO_EXT registers.

4.4.3.6.1 Features

The industrial ethernet digital I/O supports the following features:

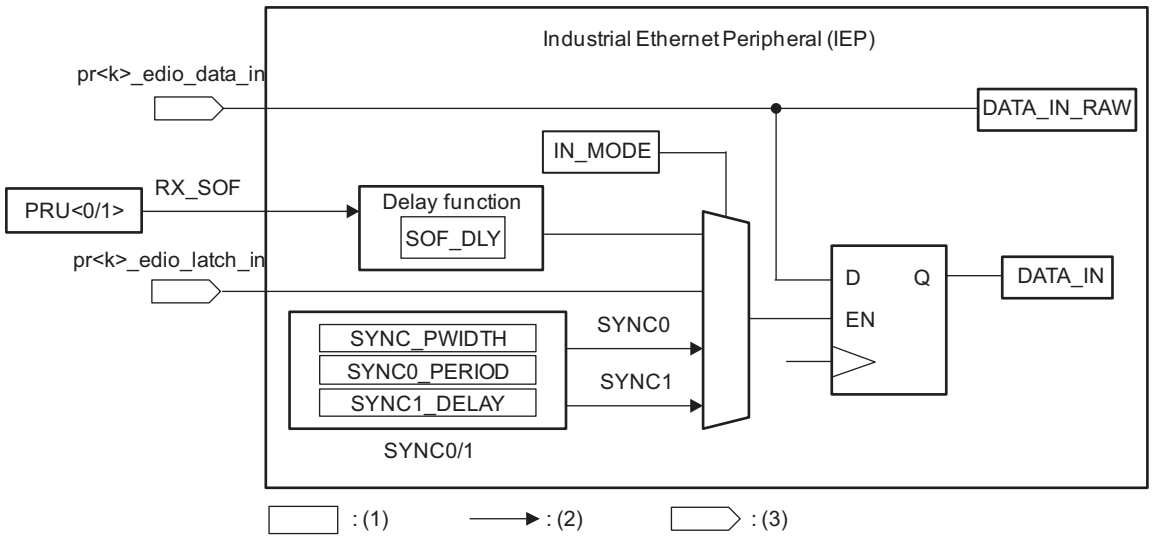
- Digital data output
  - 32 channels (pr1_edio_data_out[31:0])
  - Five event options for driving output data output:
    - End of frame event (PRU0/1_RX_EOF)
    - SYNC0 events
    - SYNC1 events
    - Watchdog trigger
    - Software enable
- Digital data out enable (optional tri-state control)
- Digital data input
  - 32 channels (pr1_edio_data_in[31:0])
  - DIGIO_DATA_IN_RAW supports direct sampling of pr1_edio_data_in
  - DIGIO_DATA_IN supports four event options to trigger sampling of pr1_edio_data_in:
    - Start of frame event in start of frame (SOF) mode
    - pr1_edio_latch_in event
    - SYNC0 events
    - SYNC1 events

NOTE: Some devices may not pin out all 32 data I/O signals. For which data pins are available on this device, see Table 4-3, *PRU-ICSS Pin List*.

4.4.3.6.2 DIGIO Block Diagrams

Figure 4-22 shows the signals and registers for capturing the DIGIO data in. In PRU0/1_RX_SOF mode, the delay time of capturing pr1_edio_data_in is programmable through the SOF_DLY bit of the DIGIO_EXT register.

Figure 4-22. IEP DIGIO Data In



(1) Register

(2) Internal signal wire

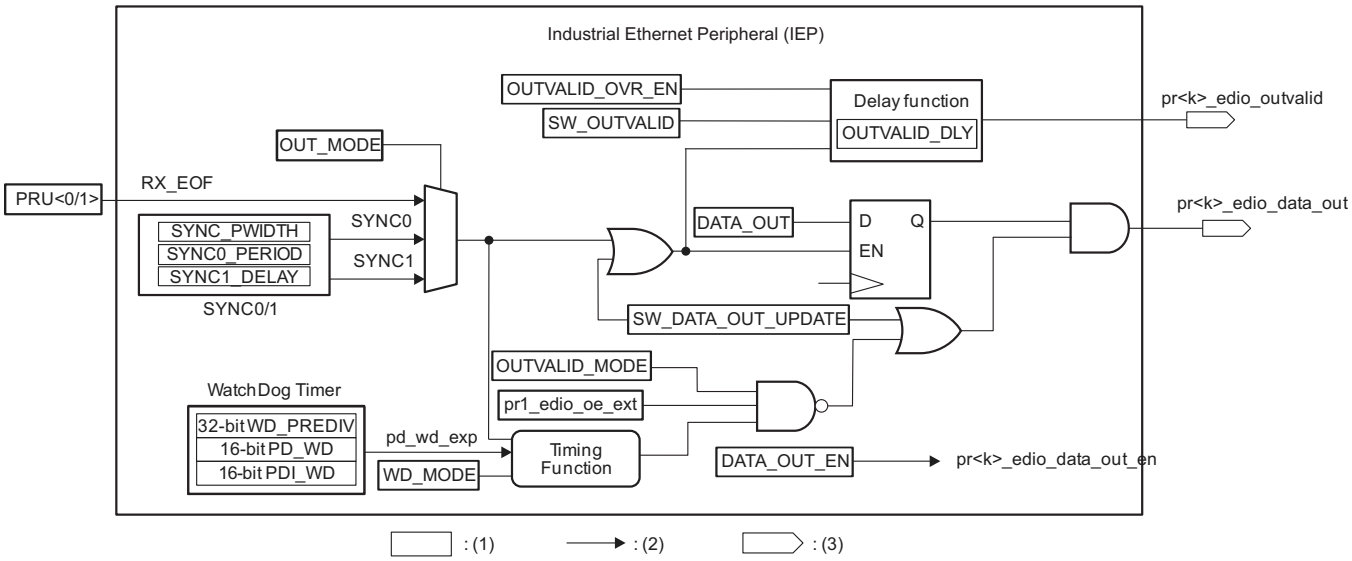
(3) External pin input/output

(1) Register

(2) External pin input/output

Figure 4-23 shows the signals and registers for driving the DIGIO data out. The pr1_edio_data_out is immediately forced to zero when OUTVALID_MODE = 1, pr1_edio_oe_ext = 1, and PD_WD_EXP = 1, or the next update hardware post PD_WD_EXP. Delay assertion of pr1_edio_outvalid from pr1_edio_data_out update events are controlled by software (SW_OUTVALID).

Figure 4-23. IEP DIGIO Data Out



(1) Register

(2) Internal signal wire

(3) External pin input/output

(1) Register

(2) External pin input/output

4.4.3.6.3 Basic Programming Model

Follow these steps to configure and read the DIGIO Data Input.

1. Read DIGIO_DATA_IN_RAW for raw input data

or

1. Enable sampling of pr1_edio_data_in[31:0] by setting DIGIO_CTRL.IN_MODE
2. Read DIGIO_DATA_IN for data sampled by the selected trigger source

Follow these steps to configure and write to the DIGIO Data Output.

1. Pre-configure DIGIO by setting DIGIO_EXP.OUTVALID_OVR_EN and DIGIO_EXP.SW_DATA_OUT_UPDATE
2. Write to DIGIO_DATA_OUT to configure output data
3. To Hi-Z output, set DIGIO_DATA_OUT_EN (clear DIGIO_DATA_OUT_EN to drive value stored in DIGIO_DATA_OUT)

4.4.4 Universal Asynchronous Receiver/Transmitter (UART)

4.4.4.1 Introduction

4.4.4.1.1 Purpose of the Peripheral

The PRU UART peripheral within the PRU-ICSS is based on the industry standard TL16C550 asynchronous communications element, which in turn is a functional upgrade of the TL16C450. Functionally similar to the TL16C450 on power up (single character or TL16C450 mode), the UART can be placed in an alternate FIFO (TL16C550) mode. This relieves the CPU of excessive software overhead by buffering received and transmitted characters. The receiver and transmitter FIFOs store up to 16 bytes including three additional bits of error status per byte for the receiver FIFO.

The UART performs serial-to-parallel conversions on data received from a peripheral device and parallel-to-serial conversion on data received from the CPU. The CPU can read the UART status at any time. The UART includes control capability and a processor interrupt system that can be tailored to minimize software management of the communications link.

The UART includes a programmable baud rate generator capable of dividing the PRU_ICSS_UART_CLK input clock by divisors from 1 to 65535 and producing a 16× reference clock or a 13× reference clock for the internal transmitter and receiver logic. For detailed timing and electrical specifications for the UART, see your device-specific data manual.

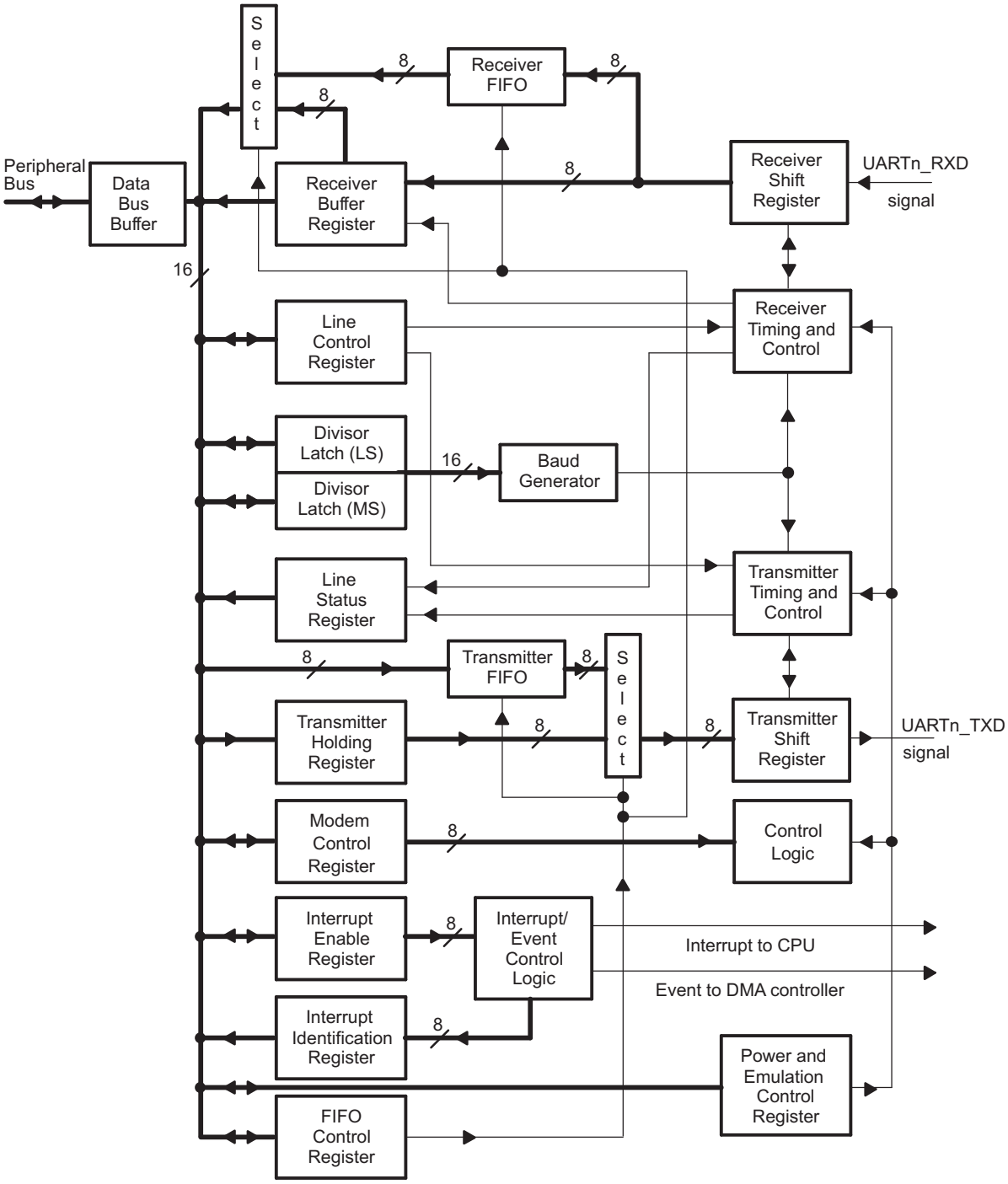
4.4.4.1.2 Functional Block Diagram

A functional block diagram of the UART is shown in Figure 4-24.

4.4.4.1.3 Industry Standard(s) Compliance Statement

The UART peripheral is based on the industry standard TL16C550 asynchronous communications element, which is a functional upgrade of the TL16C450. The information in this chapter assumes you are familiar with these standards.

Figure 4-24. UART Block Diagram



NOTE: The value *n* indicates the applicable UART where there are multiple instances. For the PRU-ICSS, there is only one instance and all UART signals should reflect this (e.g., UART0_TXD instead of UARTn_TXD).

4.4.4.2 Functional Description

4.4.4.2.1 Clock Generation and Control

The UART bit clock is derived from an input clock to the UART. See your device-specific data manual to check the maximum data rate supported by the UART.

Figure 4-25 is a conceptual clock generation diagram for the UART. The PRU_ICSS_UART_CLK is input to the Clock Generator, which uses a programmable divider to produce the UART input clock. The UART contains a programmable baud generator that takes the UART input clock and divides it by a divisor in the range between 1 and (2^{16} - 1) to produce a baud clock (BCLK). The frequency of BCLK is sixteen times (16×) the baud rate (each received or transmitted bit lasts 16 BCLK cycles) or thirteen times (13×) the baud rate (each received or transmitted bit lasts 13 BCLK cycles). When the UART is receiving, the bit is sampled in the 8th BCLK cycle for 16× over sampling mode and on the 6th BCLK cycle for 13× over-sampling mode. The 16× or 13× reference clock is selected by configuring the OSM_SEL bit in the mode definition register (MDR). The formula to calculate the divisor is:

$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 16} \quad [\text{MDR.OSM\_SEL} = 0]$$

$$\text{Divisor} = \frac{\text{UART input clock frequency}}{\text{Desired baud rate} \times 13} \quad [\text{MDR.OSM\_SEL} = 1]$$

Two 8-bit register fields (DLH and DLL), called divisor latches, hold this 16-bit divisor. DLH holds the most significant bits of the divisor, and DLL holds the least significant bits of the divisor. For information about these register fields, see the UART register descriptions. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

Figure 4-26 summarizes the relationship between the transferred data bit, BCLK, and the UART input clock. Note that the timing relationship depicted in Figure 4-26 shows that each bit lasts for 16 BCLK cycles . This is in case of 16x over-sampling mode. For 13× over-sampling mode each bit lasts for 13 BCLK cycles .

Example baud rates and divisor values relative to a 192-MHz UART input clock and 16× over-sampling mode are shown in Table 4-24.

Figure 4-25. UART Clock Generation Diagram

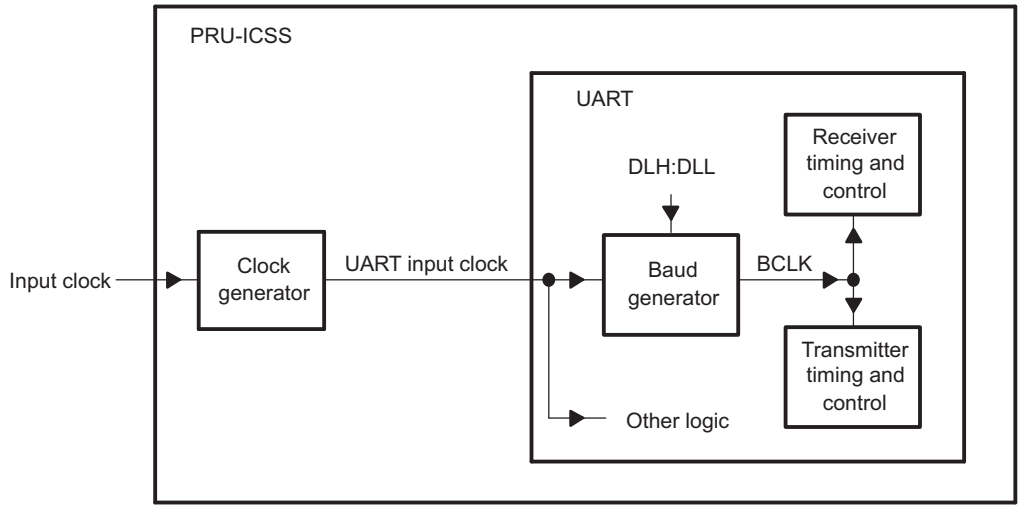


Figure 4-26. Relationships Between Data Bit, BCLK, and UART Input Clock

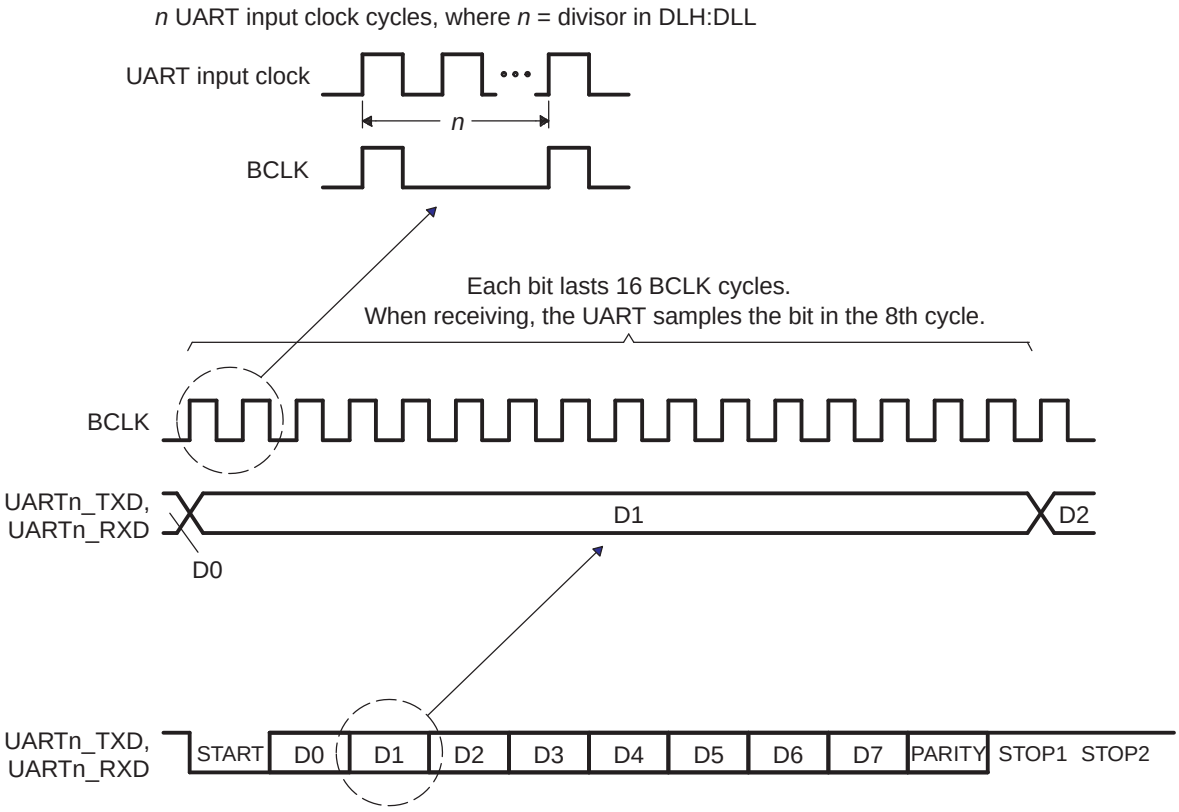


Table 4-24. Baud Rate Examples for 192-MHZ UART Input Clock and 16× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	5000	2400	0.00
4800	2500	4800	0.00
9600	1250	9600	0.00
19200	625	19200	0.00
38400	313	38338.658	-0.16
56000	214	56074.766	0.13
128000	94	127659.574	-0.27
300000	40	300000	0.00

Table 4-25. Baud Rate Examples for 192-MHZ UART Input Clock and 13× Over-sampling Mode

Baud Rate	Divisor Value	Actual Baud Rate	Error (%)
2400	6154	2399.940	-0.0025
4800	3077	4799.880	-0.0025
9600	1538	9602.881	0.03
19200	769	19205.762	0.03
38400	385	38361.638	-0.10
56000	264	55944.056	-0.10
128000	115	128428.094	0.33
300000	49	301412.873	0.47

4.4.4.2.2 Signal Descriptions

The UARTs utilize a minimal number of signal connections to interface with external devices. The UART signal descriptions are included in Table 4-26. Note that the number of UARTs and their supported features vary on each device. See your device-specific data manual for more details.

Table 4-26. UART Signal Descriptions

Signal Name[1]	Signal Type	Function
UARTn_TXD	Output	Serial data transmit
UARTn_RXD	Input	Serial data receive
$\overline{\text{UARTn\_CTS}}$ [2]	Input	Clear-to-Send handshaking signal
$\overline{\text{UARTn\_RTS}}$ [2]	Output	Request-to-Send handshaking signal

(1) The value *n* indicates the applicable UART; that is, UART0, UART1, etc.
(2) This signal is not supported in all UARTs. See your device-specific data manual to check if it is supported.

4.4.4.2.3 Pin Multiplexing

Extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. See your device-specific data manual to determine how pin multiplexing affects the UART.

4.4.4.2.4 Protocol Description

4.4.4.2.4.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

4.4.4.2.4.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

4.4.4.2.4.3 Data Format

The UART transmits in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + STOP bit (1, 1.5, 2)

It transmits 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1, 1.5, or 2 STOP bits, depending on the STOP bit selection.

The UART receives in the following format:

1 START bit + data bits (5, 6, 7, 8) + 1 PARITY bit (optional) + 1 STOP bit

It receives 1 START bit; 5, 6, 7, or 8 data bits, depending on the data width selection; 1 PARITY bit, if parity is selected; and 1 STOP bit.

The protocol formats are shown in Figure 4-27.

Figure 4-27. UART Protocol Formats

Transmit/Receive for 5-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	PARITY	STOP1

Transmit/Receive for 6-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	PARITY	STOP1

Transmit/Receive for 7-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	D6	PARITY	STOP1

Transmit/Receive for 8-bit data, parity Enable, 1 STOP bit

		D0	D1	D2	D3	D4	D5	D6	D7	PARITY	STOP1

4.4.4.2.5 Operation

4.4.4.2.5.1 Transmission

The UART transmitter section includes a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART transmitter sends the following to the receiving device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1, 1.5, or 2 STOP bits

THR receives data from the internal data bus, and when TSR is ready, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the UARTn_TXD pin.

In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR or the interrupt identification register (IIR) is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or IIR is read.

4.4.4.2.5.2 Reception

The UART receiver section includes a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16× receiver clock. Receiver section control is a function of the UART line control register (LCR). Based on the settings chosen in LCR, the UART receiver accepts the following from the transmitting device:

- 1 START bit
- 5, 6, 7, or 8 data bits
- 1 PARITY bit (optional)
- 1 STOP bit (any other STOP bits transferred with the above data are not detected)

RSR receives the data bits from the UARTn_RXD pin. Then RSR concatenates the data bits and moves the resulting value into RBR (or the receiver FIFO). The UART also stores three bits of error status information next to each received character, to record a parity error, framing error, or break.

In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled in the interrupt enable register (IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

4.4.4.2.5.3 FIFO Modes

The following two modes can be used for servicing the receiver and transmitter FIFOs:

- FIFO interrupt mode. The FIFO is enabled and the associated interrupts are enabled. Interrupts are sent to the CPU to indicate when specific events occur.
- FIFO poll mode. The FIFO is enabled but the associated interrupts are disabled. The CPU polls status bits to detect specific events.

Because the receiver FIFO and the transmitter FIFO are controlled separately, either one or both can be placed into the interrupt mode or the poll mode.

4.4.4.2.5.3.1 FIFO Interrupt Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are enabled in the interrupt enable register (IER), the interrupt mode is selected for the receiver FIFO. The following are important points about the receiver interrupts:

- The receiver data-ready interrupt is issued to the CPU when the FIFO has reached the trigger level that is programmed in FCR. It is cleared when the CPU or the DMA controller reads enough characters from the FIFO such that the FIFO drops below its programmed trigger level.
- The receiver line status interrupt is generated in response to an overrun error, a parity error, a framing error, or a break. This interrupt has higher priority than the receiver data-ready interrupt. For details, see Section 4.4.4.2.8.
- The data-ready (DR) bit in the line status register (LSR) indicates the presence or absence of characters in the receiver FIFO. The DR bit is set when a character is transferred from the receiver shift register (RSR) to the empty receiver FIFO. The DR bit remains set until the FIFO is empty again.
- A receiver time-out interrupt occurs if all of the following conditions exist:
  - At least one character is in the FIFO,
  - The most recent character was received more than four continuous character times ago. A character time is the time allotted for 1 START bit, *n* data bits, 1 PARITY bit, and 1 STOP bit, where *n* depends on the word length selected with the WLS bits in the line control register (LCR). See Table 4-27.
  - The most recent read of the FIFO has occurred more than four continuous character times before.
- Character times are calculated by using the baud rate.
- When a receiver time-out interrupt has occurred, it is cleared and the time-out timer is cleared when the CPU or the EDMA controller reads one character from the receiver FIFO. The interrupt is also cleared if a new character is received in the FIFO or if the URRST bit is cleared in the power and emulation management register (PWREMU_MGMT).
- If a receiver time-out interrupt has not occurred, the time-out timer is cleared after a new character is received or after the CPU or EDMA reads the receiver FIFO.

When the transmitter FIFO is enabled in FCR and the transmitter holding register empty (THRE) interrupt is enabled in IER, the interrupt mode is selected for the transmitter FIFO. The THRE interrupt occurs when the transmitter FIFO is empty. It is cleared when the transmitter hold register (THR) is loaded (1 to 16 characters may be written to the transmitter FIFO while servicing this interrupt) or the interrupt identification register (IIR) is read.

Table 4-27. Character Time for Word Lengths

Word Length (*n*)	Character Time	Four Character Times
5	Time for 8 bits	Time for 32 bits
6	Time for 9 bits	Time for 36 bits
7	Time for 10 bits	Time for 40 bits
8	Time for 11 bits	Time for 44 bits

4.4.4.2.5.3.2 FIFO Poll Mode

When the receiver FIFO is enabled in the FIFO control register (FCR) and the receiver interrupts are disabled in the interrupt enable register (IER), the poll mode is selected for the receiver FIFO. Similarly, when the transmitter FIFO is enabled and the transmitter interrupts are disabled, the transmitted FIFO is in the poll mode. In the poll mode, the CPU detects events by checking bits in the line status register (LSR):

- The RXFIFOE bit indicates whether there are any errors in the receiver FIFO.
- The TEMT bit indicates that both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
- The THRE bit indicates when THR is empty.
- The BI (break), FE (framing error), PE (parity error), and OE (overrun error) bits specify which error or errors have occurred.
- The DR (data-ready) bit is set as long as there is at least one byte in the receiver FIFO.

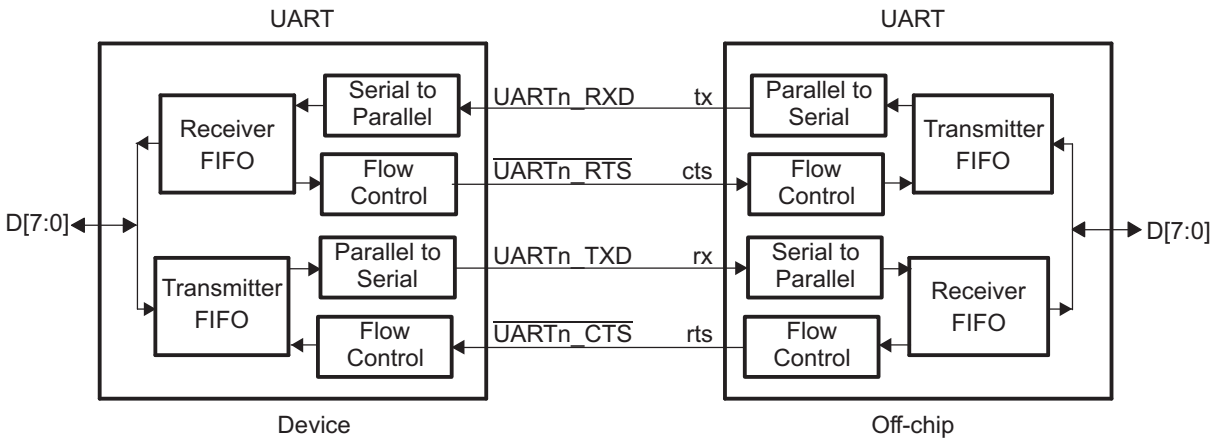
Also, in the FIFO poll mode:

- The interrupt identification register (IIR) is not affected by any events because the interrupts are disabled.
- The UART does not indicate when the receiver FIFO trigger level is reached or when a receiver time-out occurs.

4.4.4.2.5.4 Autoflow Control

The UART can employ autoflow control by connecting the $\overline{\text{UARTn\_CTS}}$ and $\overline{\text{UARTn\_RTS}}$ signals. Note that all UARTs do not support autoflow control; see your device-specific data manual for supported features. The $\overline{\text{UARTn\_CTS}}$ input must be active before the transmitter FIFO can transmit data. The $\overline{\text{UARTn\_RTS}}$ output becomes active when the receiver needs more data and notifies the sending device. When $\overline{\text{UARTn\_RTS}}$ is connected to $\overline{\text{UARTn\_CTS}}$, data transmission does not occur unless the receiver FIFO has space for the data. Therefore, when two UARTs are connected as shown in Figure 4-28 with autoflow enabled, overrun errors are eliminated.

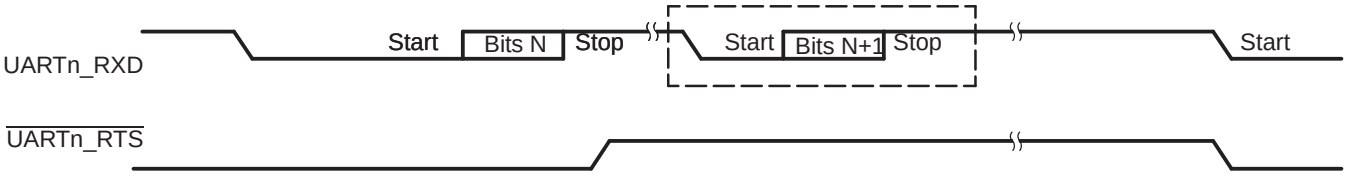
Figure 4-28. UART Interface Using Autoflow Diagram



4.4.4.2.5.4.1 UARTn_RTS Behavior

UARTn_RTS data flow control originates in the receiver block (see Figure 4-24). When the receiver FIFO level reaches a trigger level of 1, 4, 8, or 14 (see Figure 4-29), UARTn_RTS is deasserted. The sending UART may send an additional byte after the trigger level is reached (assuming the sending UART has another byte to send), because it may not recognize the deassertion of UARTn_RTS until after it has begun sending the additional byte. For trigger level 1, 4, and 8, UARTn_RTS is automatically reasserted once the receiver FIFO is emptied. For trigger level 14, UARTn_RTS is automatically reasserted once the receiver FIFO drops below the trigger level.

Figure 4-29. Autoflow Functional Timing Waveforms for UARTn_RTS



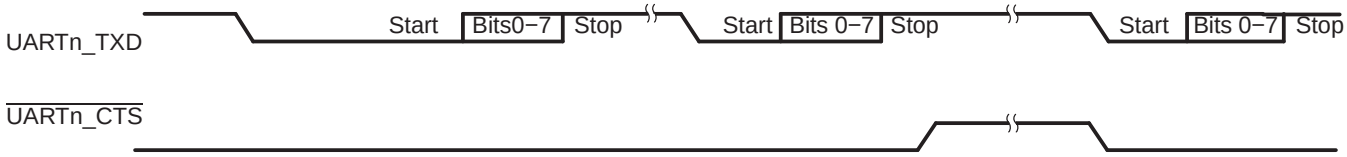
(1) N = Receiver FIFO trigger level.

(2) The block in dashed lines covers the case where an additional byte is sent.

4.4.4.2.5.4.2 UARTn_CTS Behavior

The transmitter checks UARTn_CTS before sending the next data byte. If UARTn_CTS is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, UARTn_CTS must be released before the middle of the last STOP bit that is currently being sent (see Figure 4-30). When flow control is enabled, UARTn_CTS level changes do not trigger interrupts because the device automatically controls its own transmitter. Without autoflow control, the transmitter sends any data present in the transmitter FIFO and a receiver overrun error may result.

Figure 4-30. Autoflow Functional Timing Waveforms for UARTn_CTS



(1) When UARTn_CTS is active (low), the transmitter keeps sending serial data out.

(2) When UARTn_CTS goes high before the middle of the last STOP bit of the current byte, the transmitter finishes sending the current byte but it does not send the next byte.

(3) When UARTn_CTS goes from high to low, the transmitter begins sending data again.

4.4.4.2.5.5 Loopback Control

The UART can be placed in the diagnostic mode using the LOOP bit in the modem control register (MCR), which internally connects the UART output back to the UART input. In this mode, the transmit and receive data paths, the transmitter and receiver interrupts, and the modem control interrupts can be verified without connecting to another UART.

4.4.4.2.6 Reset Considerations

4.4.4.2.6.1 Software Reset Considerations

Two bits in the power and emulation management register (PWREMU_MGMT) control resetting the parts of the UART:

- The UTRST bit controls resetting the transmitter only. If UTRST = 1, the transmitter is active; if UTRST = 0, the transmitter is in reset.
- The URRST bit controls resetting the receiver only. If URRST = 1, the receiver is active; if URRST = 0, the receiver is in reset.

In each case, putting the receiver and/or transmitter in reset will reset the state machine of the affected portion but does not affect the UART registers.

4.4.4.2.6.2 Hardware Reset Considerations

When the processor RESET pin is asserted, the entire processor is reset and is held in the reset state until the RESET pin is released. As part of a device reset, the UART state machine is reset and the UART registers are forced to their default states. The default states of the registers are shown in Section 4.5.5.

4.4.4.2.7 Initialization

The following steps are required to initialize the UART:

1. Perform the necessary device pin multiplexing setup (see your device-specific data manual).
2. Set the desired baud rate by writing the appropriate clock divisor values to the divisor latch registers (DLL and DLH).
3. If the FIFOs will be used, select the desired trigger level and enable the FIFOs by writing the appropriate values to the FIFO control register (FCR). The FIFOEN bit in FCR must be set first, before the other bits in FCR are configured.
4. Choose the desired protocol settings by writing the appropriate values to the line control register (LCR).
5. If autoflow control is desired, write appropriate values to the modem control register (MCR). Note that all UARTs do not support autoflow control; see your device-specific data manual for supported features.
6. Choose the desired response to emulation suspend events by configuring the FREE bit, and enable the UART by setting the UTRST and URRST bits in the power and emulation management register (PWREMU_MGMT).

4.4.4.2.8 Interrupt Support

4.4.4.2.8.1 Interrupt Events and Requests

The UART generates the interrupt requests described in Table 4-28. All requests are multiplexed through an arbiter to a single UART interrupt request to the CPU, as shown in Figure 4-31. Each of the interrupt requests has an enable bit in the interrupt enable register (IER) and is recorded in the interrupt identification register (IIR).

If an interrupt occurs and the corresponding enable bit is set to 1, the interrupt request is recorded in IIR and is forwarded to the CPU. If an interrupt occurs and the corresponding enable bit is cleared to 0, the interrupt request is blocked. The interrupt request is neither recorded in IIR nor forwarded to the CPU.

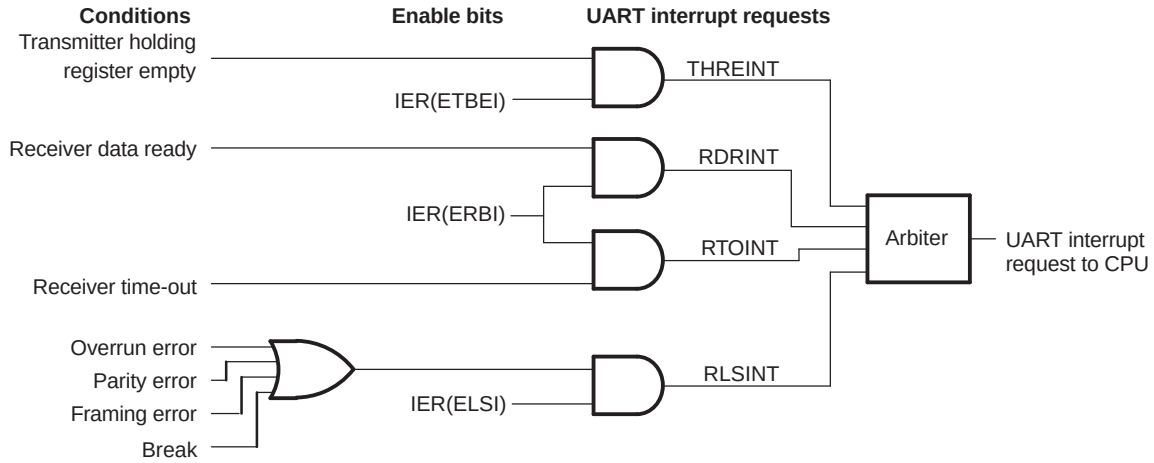
4.4.4.2.8.2 Interrupt Multiplexing

The UARTs have dedicated interrupt signals to the CPU and the interrupts are not multiplexed with any other interrupt source.

Table 4-28. UART Interrupt Requests Descriptions

UART Interrupt Request	Interrupt Source	Comment
THREINT	THR-empty condition: The transmitter holding register (THR) or the transmitter FIFO is empty. All of the data has been copied from THR to the transmitter shift register (TSR).	If THREINT is enabled in IER, by setting the ETBEI bit, it is recorded in IIR. As an alternative to using THREINT, the CPU can poll the THRE bit in the line status register (LSR).
RDAINT	Receive data available in non-FIFO mode or trigger level reached in the FIFO mode.	If RDAINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. As an alternative to using RDAINT, the CPU can poll the DR bit in the line status register (LSR). In the FIFO mode, this is not a functionally equivalent alternative because the DR bit does not respond to the FIFO trigger level. The DR bit only indicates the presence or absence of unread characters.
RTOINT	Receiver time-out condition (in the FIFO mode only): No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 4-27), and there is at least one character in the receiver FIFO during this time.	The receiver time-out interrupt prevents the UART from waiting indefinitely, in the case when the receiver FIFO level is below the trigger level and thus does not generate a receiver data-ready interrupt. If RTOINT is enabled in IER, by setting the ERBI bit, it is recorded in IIR. There is no status bit to reflect the occurrence of a time-out condition.
RLSINT	Receiver line status condition: An overrun error, parity error, framing error, or break has occurred.	If RLSINT is enabled in IER, by setting the ELSI bit, it is recorded in IIR. As an alternative to using RLSINT, the CPU can poll the following bits in the line status register (LSR): overrun error indicator (OE), parity error indicator (PE), framing error indicator (FE), and break indicator (BI).

Figure 4-31. UART Interrupt Request Enable Paths



4.4.4.2.9 DMA Event Support

In the FIFO mode, the UART generates the following two DMA events:

- **Receive event (URXEVT):** The trigger level for the receiver FIFO (1, 4, 8, or 14 characters) is set with the RXFIFTL bit in the FIFO control register (FCR). Every time the trigger level is reached or a receiver time-out occurs, the UART sends a receive event to the EDMA controller. In response, the EDMA controller reads the data from the receiver FIFO by way of the receiver buffer register (RBR). Note that the receive event is not asserted if the data at the top of the receiver FIFO is erroneous even if the trigger level has been reached.
- **Transmit event (UTXEVT):** When the transmitter FIFO is empty (when the last byte in the transmitter FIFO has been copied to the transmitter shift register), the UART sends a UTXEVT signal to the EDMA controller. In response, the EDMA controller refills the transmitter FIFO by way of the transmitter holding register (THR). The UTXEVT signal is also sent to the DMA controller when the UART is taken out of reset using the UTRST bit in the power and emulation management register (PWREMU_MGMT).

Activity in DMA channels can be synchronized to these events. In the non-FIFO mode, the UART generates no DMA events. Any DMA channel synchronized to either of these events must be enabled at the time the UART event is generated. Otherwise, the DMA channel will miss the event and, unless the UART generates a new event, no data transfer will occur.

4.4.4.2.10 Power Management

The UART peripheral can be placed in reduced-power modes to conserve power during periods of low activity. The power management of the UART peripheral is controlled by the processor Power and Clock Management (PRCM). The PRCM acts as a master controller for power management for all of the peripherals on the device. For detailed information on power management procedures using the PSC, see the chapter *Power, Reset, and Clock Management (PRCM)* in the device reference manual.

4.4.4.2.11 Emulation Considerations

The FREE bit in the power and emulation management register (PWREMU_MGMT) determines how the UART responds to an emulation suspend event such as an emulator halt or breakpoint. If FREE = 0 and a transmission is in progress, the UART halts after completing the one-word transmission; if FREE = 0 and a transmission is not in progress, the UART halts immediately. If FREE = 1, the UART does not halt and continues operating normally.

Note also that most emulator accesses are transparent to UART operation. Emulator read operations do not affect any register contents, status bits, or operating states, with the exception of the interrupt identification register (IIR). Emulator writes, however, may affect register contents and may affect UART operation, depending on which register is accessed and what value is written.

The UART registers can be read from or written to during emulation suspend events, even if the UART activity has stopped.

4.4.4.2.12 Exception Processing

4.4.4.2.12.1 Divisor Latch Not Programmed

Since the processor reset signal has no effect on the divisor latch, the divisor latch will have an unknown value after power up. If the divisor latch is not programmed after power up, the baud clock (BCLK) will not operate and will instead be set to a constant logic 1 state.

The divisor latch values should always be reinitialized following a processor reset.

4.4.4.2.12.2 Changing Operating Mode During Busy Serial Communication

Since the serial link characteristics are based on how the control registers are programmed, the UART will expect the control registers to be static while it is busy engaging in a serial communication. Therefore, changing the control registers while the module is still busy communicating with another serial device will most likely cause an error condition and should be avoided.

4.4.5 ECAP

The PRU ECAP module within the PRU-ICSS is identical to the ECAP module in the AM335x PWMSS. For additional details about the ECAP module, see Section 15.3, *Pulse-Width Modulation Subsystem (PWMSS)*.

4.4.6 MII_RT

4.4.6.1 Introduction

The Real-time Media Independent Interface (MII_RT) provides a programmable I/O interface for the PRUs to access and control up to two MII ports. The MII_RT module can also be configured to push and pull data independent of the PRU cores.

NOTE: To ensure the MII_RT IO timing values published in the device data manual, the ocp_clk must be configured for 200 MHz (default value) and TXCFG<n>_TX_CLK_DELAY must be set to 6h (non-default value).

4.4.6.1.1 Features

The PRU-ICSS MII_RT module supports:

- Two MII ports
  - Each MII port has:
    - 32-byte RX L1 FIFO
    - 64-byte RX L2 buffer
    - 64-byte TX L1 FIFO
  - Rate decoupling on TX L1 FIFO
  - Configurable pre-amble removal on RX L1 FIFO and insertion on TX L1 FIFO
  - Configurable TX L1 FIFO trigger (10 bits with 40 ns ticks)
- MII port multiplexer per direction to support line/ring structure
  - Link detection through RX_ERR
- Cyclic redundancy check (CRC)
  - CRC32 generation on TX path
  - CRC32 checker on RX path

4.4.6.1.2 Unsupported Features

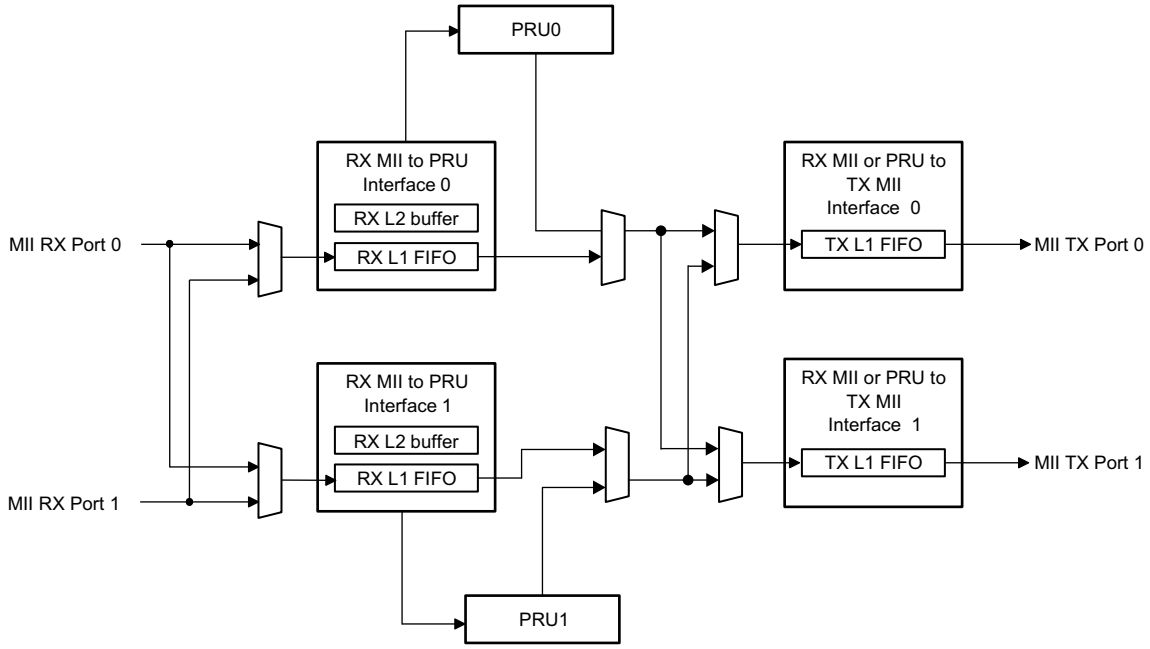
The PRU-ICSS MII_RT module does not support:

- Auto padding in TX L1 FIFO
- Dynamic TX multiplexer switching during packet handling
  - Can allow one PRU to handle both MII interfaces and a second PRU to manage the host and switch functions.

4.4.6.1.3 Block Diagram

Figure 4-32 shows the MII_RT in context of the PRU-ICSS. This diagram is a conceptual block diagram and does not necessarily reflect actual topologies.

Figure 4-32. MII_RT Block Diagram



4.4.6.2 Functional Description

4.4.6.2.1 Data Path Configuration

The MII_RT module supports three basic data path configurations. These configurations are compared in Table 4-29 and described in the following sections.

Table 4-29. Data Path Configuration Comparison

Configuration	PRU Dependency	Data Servicing	Port-to-Port Latency
Auto-forward	Snoop only	One word in flight	Low
8- or 16-bit processing with on-the-fly modifications (RX L1)	Yes	One word or byte in flight	Low
32-byte double buffer or ping-pong processing (RX L2)	Yes	Multi-words in flight	Medium (application-dependent)

4.4.6.2.1.1 Auto-forward with Optional PRU Snoop

Data is automatically forwarded from the MII RX port to the MII TX port without manipulations, as shown in Figure 4-33. This configuration does not depend on the PRU core. However, it does support an option for PRU to snoop or monitor the received data through the RX L2, shown in Figure 4-34. The PRU does not access data and status bits through R31, and it does not modify and push data.

Figure 4-33. Auto-forward

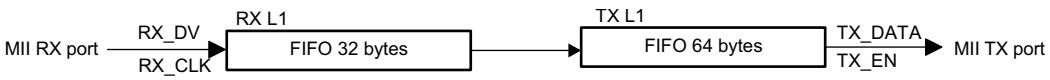
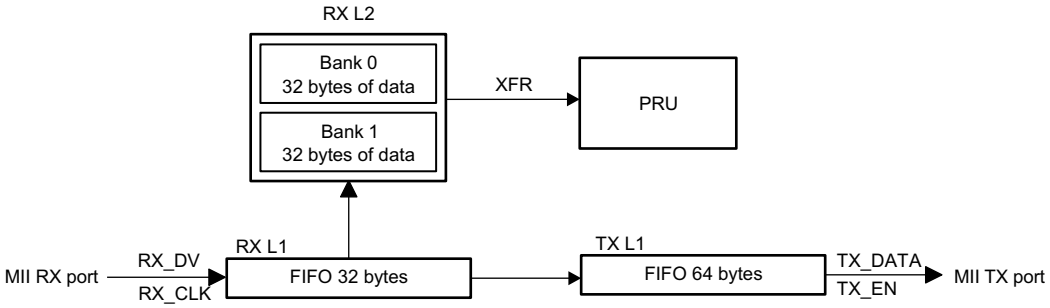


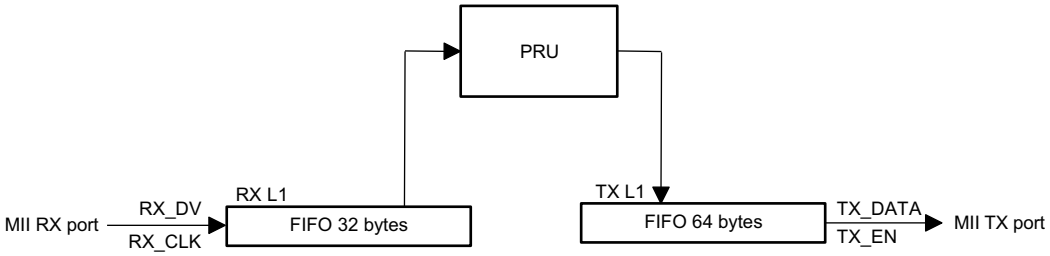
Figure 4-34. Auto-forward with PRU Snoop



4.4.6.2.1.2 8- or 16-bit Processing with On-the-Fly Modifications

This configuration services one byte or word in flight and has low latency. The PRU has the option to manipulate the received word and control popping data from the RX L1 FIFO and pushing it on the TX L1 FIFO.

Figure 4-35. 8- or 16-bit Processing with On-the-Fly Modifications

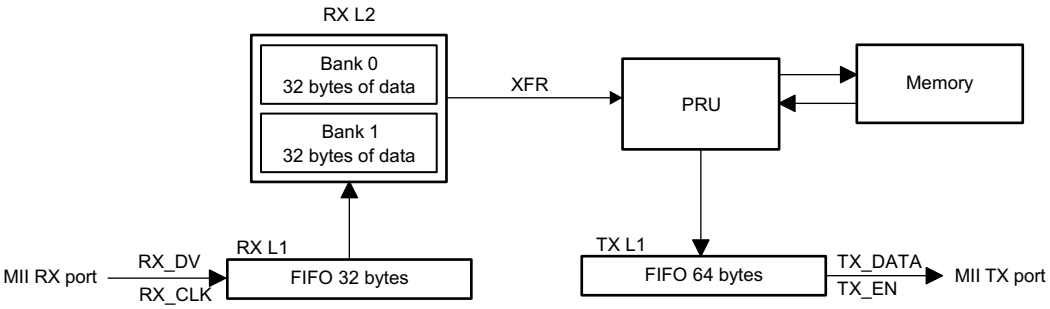


4.4.6.2.1.3 32-byte Double Buffer or Ping-Pong Processing

This configuration supports high bandwidth, high efficiency transactions. Often implementations using this mode permit relaxed servicing requirements allowing the PRU to manipulate the received data before transmitting.

Data received in this configuration is passed into the RX L2 buffer. The PRU reads multiple bytes of data from one of the RX L2 banks through the high bandwidth broadside interface and XFR instructions. The PRU can then store or manipulate data before pushing it to the TX L1 FIFO for transmission on the MII TX port.

Figure 4-36. 32-byte Double Buffer or Ping-Pong Processing



4.4.6.2.2 Definition and Terms

4.4.6.2.2.1 Data Frame Structure

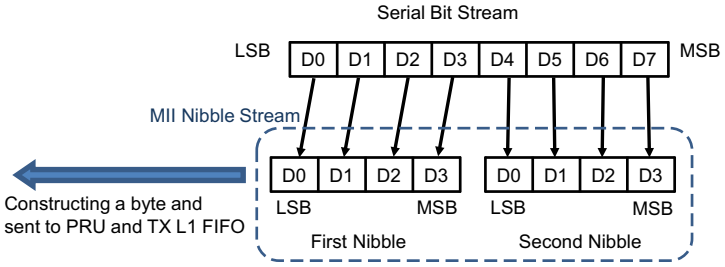
The data received and transmitted over MII conforms with the following frame structure:

Table 4-30. Frame Structure

Inter-frame	Preamble	Start of Frame Delimiter (SFD)	Data	Cyclic Redundancy Check (CRC)

The data following the SFD is formatted in a 4-bit nibble structure. Figure 4-37 illustrates the nibble order. The MSB arriving first is on the LSB side of a nibble. When receiving data, the MII_RT receive logic will wait for the next nibble to arrive before constructing a byte and delivering to the PRU.

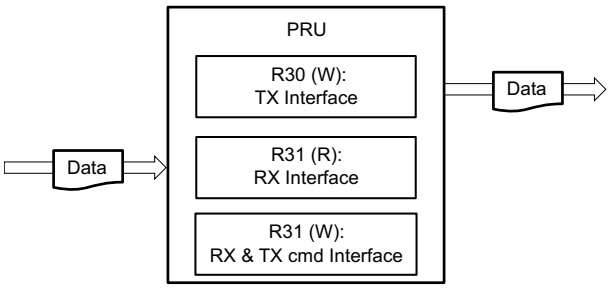
Figure 4-37. Data Nibble Structure



4.4.6.2.2.2 PRU R30 and R31

The PRU registers R30 and R31 are used to receive, transmit, and control the data for the PRU. As shown in Figure 4-38, the R31 is used to access data in the RX L1 FIFO, the R30 is used to transmit data from the PRU, and the R31 output is used the control the flow of receive and transmit. For more details about these registers, see the following sections.

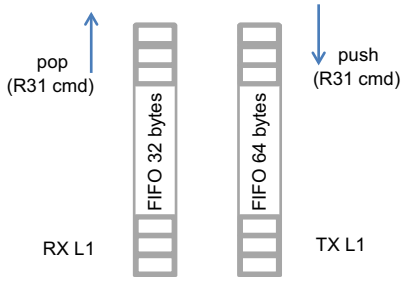
Figure 4-38. PRU R30, R31 Operations



4.4.6.2.2.3 RX and TX L1 FIFO Data Movement

To advance the next data byte seen by R31, the PRU must pop the data from the RX L1 FIFO. Likewise, the PRU can push the data from R30 to the TX L1 FIFO. These operations are illustrated in Figure 4-39.

Figure 4-39. Reading and Writing FIFO Data



4.4.6.2.2.4 CRC Computation

4.4.6.2.2.4.1 Receive CRC Computation

For the incoming data, the MII_RT calculates CRC32 and then compares against the value provided in the incoming frame. If there is a mismatch, the MII RT signals ERROR_CRC to the PRU. If a previous node or Ethernet device appended an error nibble, the CRC calculation of received packet will be wrong because the longer frame and the frame length will end at a 4-bit boundary instead of the usual 8-bit boundary. When RX_DV goes inactive on the 4-bit boundary, the interface will assert DATA_RDY and BYTE_RDY flag with the ERROR NIBBLE. The error event is also mapped into the PRU-ICSS INTC.

4.4.6.2.2.4.2 Transmit CRC Computation

For the outgoing data, the MII_RT calculates the CRC32 value and inserts it into outgoing packets. The CRC value computed on each MII transmit path is also available in memory map registers (MMRs) that can be read by the PRU and used primarily for debug and diagnostic purposes. The CRC is inserted into the outgoing packet based on the commands received through the R31 register of the PRU. The CRC will be inserted into the TX L1 FIFO, and there must be enough room to store the CRC value in the FIFO or else the FIFO will overflow. As Table 4-31 shows, the CRC programming model supports three sequences that provide more flexibility. Note “cmdR31” indicates write to the mentioned bits of the R31 command interface.

Table 4-31. TX CRC Programming Models

Option 1	Step 1: cmdR31 [TX_CRC_HIGH + TX_CRC_LOW + TX_EOF]
Option 2	Step 1: cmdR31 [TX_CRC_HIGH] Step 2: wait > 6 clocks (PRU cycles) Step 3: cmdR31 [TX_CRC_LOW + TX_EOF]
Option 3	Step 1: cmdR31 [TX_CRC_HIGH] Step 2: wait > 6 clocks (PRU cycles) Step 3: read TXCRC0/1 Step 4: modify CRC[15:0] Step 5: cmdR31 [TX_PUSH16 + TX_EOF + TX_ERROR_NIBBLE]

4.4.6.2.3 RX MII Interface

4.4.6.2.3.1 RX MII Submodule Overview

The RX MII interface is composed of multiple submodules that process the incoming frames and pass receive data and status information into the PRU register R31. These submodules include:

- Latch received data
- Start of frame detection
- Start frame delimiter detection

- CRC calculation and error detection
- Enhanced link detection through RX error detection

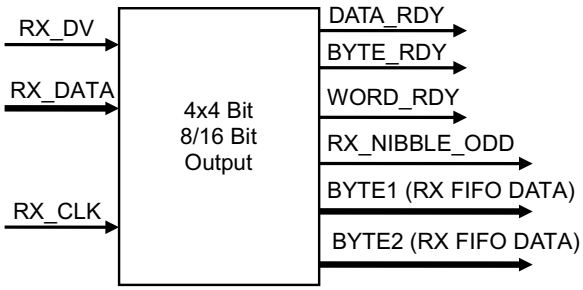
Table 4-32 includes more details about the internal signals and output of these submodules.

4.4.6.2.3.1.1 Receive Data Latch

The receive data from the MII interface is stored in the receive data FIFO which is 32 bytes. The PRU can access this data through the register R31. Depending on the configuration settings, the data can be latched on reception of one or two bytes. In each scheme, the configured number of nibbles is assembled before being copied into the PRU registers. Figure 4-40 shows the inputs and outputs of the data latch logic block.

The receiver logic in MII_RT can be programmed through the RXCFG0 and RXCFG1 registers to remove or retain the preamble + SFD from incoming frames.

Figure 4-40. RX Data Latch



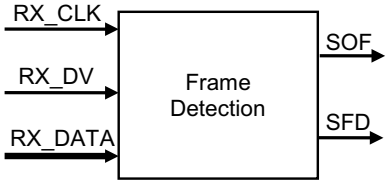
4.4.6.2.3.1.1.1 Start of Frame Detection

The start of frame detection logic tracks the frame boundaries and signals the beginning of a frame to other components of the PRU-ICSS. This logic detects two events:

- Start of Frame (SOF) event that occurs when Receive Data Valid MII signal is sampled high.
- Start of Frame Delimiter (SFD) event is seen on MII Receive Data bus.

These event triggers can be used to add timestamp to the frames. The notification for these events is available through R31 as well as through INTC which is integrated in the PRU-ICSS. Figure 4-41 shows the inputs and outputs of the start of frame detection logic block.

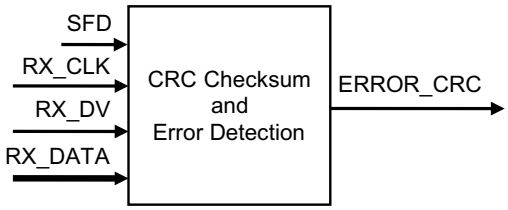
Figure 4-41. Start of Frame Detection



4.4.6.2.3.1.1.2 CRC Error Detection

For each incoming frame, the CRC is calculated by the MII_RT and compared against the CRC included in the frame. When the two values do not match, a CRC error is flagged. The ERROR_CRC indication is available in the register interface (PRU R31 Receive Interface) as well as in the FIFO interface (RX L2 Status Interface). It is also provided to the INTC which is integrated in the PRU-ICSS. Figure 4-42 shows the inputs and outputs of CRC error detection logic block.

Figure 4-42. CRC Error Detection

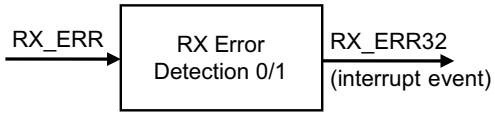


4.4.6.2.3.1.1.3 RX Error Detection and Action

The RX error detection logic tracks the receive error signaled by the physical layer and informs the PRU-ICSS INTC whenever an error is detected. Figure 4-43 shows the inputs and outputs of the RX error detection logic block. Note the following dependencies:

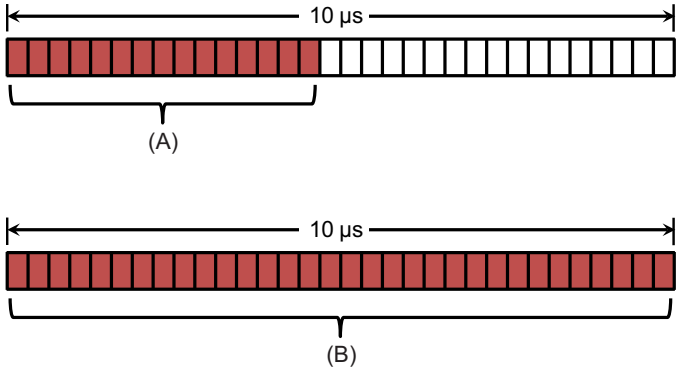
- RX_ERR signal is only sampled when RX_DV is asserted.
- All nibbles are discarded post RX_ERR event, including the nibble which had RX_ERR asserted. This state will remain until EOF occurs.
- Due to this fact, RX L1 FIFO and RX L2 FIFO will never receive any data with RX_ERR or post RX_ERR during that frame.

Figure 4-43. RX Error Detection



This submodule also keeps track of a running count of receive error events within a 10 µs error detection window, as shown in Figure 4-44. The INTC is notified when 32 or more events have occurred in a 10 µs error detection window. The error detection window is not a sliding window but a non-overlapping window with no specific initialization time with respect to incoming traffic. The timer starts its 10 µs counts immediately after de-assertion of reset to the MII_RT module.

Figure 4-44. Error Detection Window with Running Counter



A There are fewer than 32 consecutive error events in the 10 µs window. The detection module will not forward to the interrupt controller (INTC).

B There are more or equal to 32 error events in the 10 µs window. The detection module will notify the interrupt controller (INTC).

4.4.6.2.3.1.2 RX Data Path Options to PRU

There are two data path options for delivering received data to the PRU, described further in the subsequent sections:

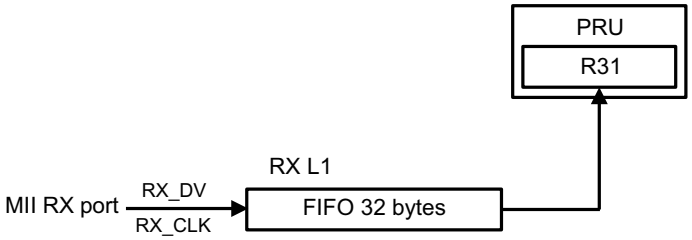
1. RX MII port → RX L1 FIFO → PRU (one word in flight)
2. RX MII port → RX L1 FIFO → RX L2 buffer → PRU (multi-word in flight)

Once the PRU has received RX data, the PRU can both manipulate received data or send data to the TX MII Interface.

4.4.6.2.3.1.2.1 RX MII Port → RX L1 FIFO → PRU

The RX L1 FIFO to PRU interface is depicted in Figure 4-45. In this mode, the data received from the MII interface is fed into the 32-byte RX L1 FIFO. The first data byte into the FIFO is automatically available in R31 of the PRU. Therefore, the PRU firmware can directly operate on this data without having to read it in a separate instruction. This allows the PRU to access receive data with low latency.

Figure 4-45. RX L1 to PRU Interface



When the new data is received, the PRU is provided with up to two bytes at a time in the R31 register, as shown in Figure 4-46. Once the PRU processes the incoming data, it instructs the MII_RT by writing to the R31 command interface bits to pop one or two bytes of data from the 32-byte RX FIFO. The pop operation causes current contents of R31 to be refreshed with new data from the incoming packet. Each time the data is popped, the status bits change to indicate so. If the pop is completed and there is no new data, the status bits immediately change to indicate no new data. Note the current R31 content, including data, will be lost after issuing the pop operation. If this information needs to be accessed later, the PRU should store the existing R31 content before popping new data.

Figure 4-46. MII RX Data to PRU R31 (R) and RX FIFO

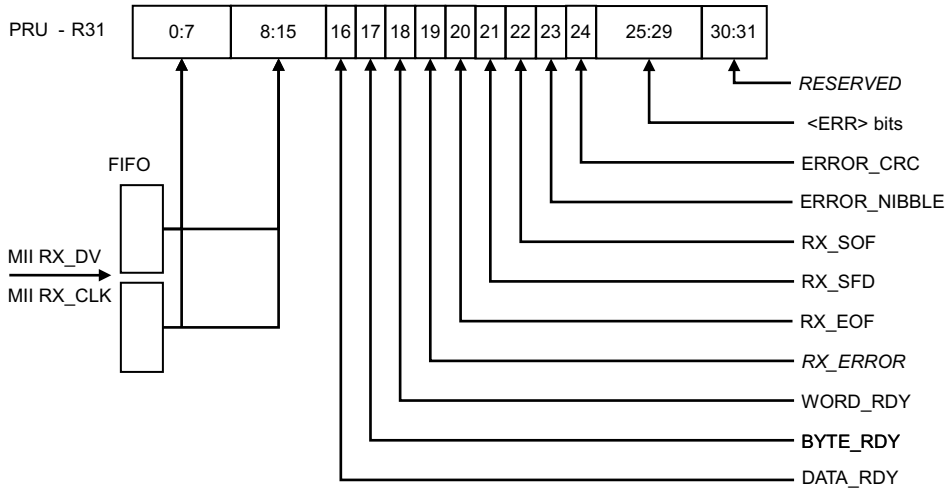


Table 4-32 describes the receive interface data and status contents provided by the R31 register. These contents are available when R31 is read. To configure this register, the PRU GPI mode should be set for MII_RT mode in the CFG register space. Note the following:

1. If the data from the receive path is not read in time, it could cause an overflow event because the data is still continuously provided to the 32-byte receive FIFO. Due to the receive FIFO overflow, the data gets automatically discarded to avoid lack of space in the FIFO. At the same time, an interrupt is raised to the INTC through a system event (PRU<n>_RX_OVERFLOW). To detect an overflow condition, the PRU should poll for this system event condition, and a RX RESET command through the R31 command interface is required to clear out from this condition. The received Ethernet frame is corrupted and should not be used for further processing, as bytes have been dropped due to the overflow condition. A FIFO reset is recommended.

2. The receive data in the R31 register is available following synchronization to the PRU clock domain. So, there is a finite delay (120 ns) when data is available from MII interface and it is accessible to the PRU.
3. The receive FIFO also has the capability to be reset through software. When reset, all contents of receive FIFO are purged and it may result in the current frame not being received as expected. When a frame is being received and the PRU resets the RX FIFO, the remaining frame is not placed into the RX FIFO. However, any new frame arriving on the receive MII port will be stored in the FIFO.

Table 4-32. PRU R31: Receive Interface Data and Status (Read Mode)

Bits	Field Name	Description
31:30	RESERVED	In case of register interface, these bits are provided to PRU by other modules in PRU-ICSS. From the MII_RT module point of view, these bits are always zero.
29	RX_MIN_FRM_CNT_ERR	RX_MIN_FRM_CNT_ERR is set to 1 when the count of total bytes of incoming frame is less than the value defined by RX_MIN_FRM_CNT. RX_MIN_FRM_CNT_ERR is cleared by RX_ERROR_CLR.
28	RX_MAX_FRM_CNT_ERR	RX_MAX_FRM_CNT_ERR is set to 1 when the count of total bytes of incoming frame is more than the value defined by RX_MAX_FRM_CNT_ERR. RX_MAX_FRM_CNT_ERR is cleared by RX_ERROR_CLR.
27	RX_EOF_ERROR	RX_EOF_ERROR is set to 1 when an RX_EOF event or RX_ERROR event occurs. RX_EOF_ERROR is cleared by RX_ EOF_CLR and/or RX_ ERROR_CLR.
26	RX_MAX_PRE_CNT_ERR	RX_MAX_PRE_CNT_ERR is set to 1 when the number of nibbles equaling 0x5 before SFD event (0x5D) is more than the value defined by RXPCNT0/1 [RX_MAX_PRE_CNT]. RX_MAX_PRE_CNT_ERR is cleared by RX_ERROR_CLR.
25	RX_ERR	RX_ERR is set to 1 when pr1_mii0/1_rxer is asserted while pr1_mii0/1_rxdv bit is set. RX_ERR is cleared by RX_ERROR_CLR.
24	ERROR_CRC	ERROR_CRC indicates that the frame has a CRC mismatch. This bit is valid when the RX_EOF bit is set. It should be noted that ERROR_CRC bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_CRC is cleared by RX_ERROR_CLR.
23	ERROR_NIBBLE	ERROR_NIBBLE indicates that the frame ended in odd nibble. It should be considered valid only when the RX_EOF bit and pr1_mii0/1_rxdv are set. Nibble counter is enabled post SFD event. It should be noted that ERROR_NIBBLE bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_NIBBLE is cleared by RX_ERROR_CLR.
22	RX_SOF	RX_SOF transitions from low to high when the frame data starts to arrive and pr1_mii0/1_rxdv is asserted. Note there will be a small sync delay of 0ns – 5ns. The PRU must write one to this bit through the R31 command interface to clear it. The recommended time to clear this bit is at the end of frame (EOF). It should be noted that RX_SOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO.
21	RX_SFD	RX_SFD transitions from low to high when the SFD sequence (0x5D) post RX_SOF is observed on the receive MII data. The PRU must write one to this bit through the R31 command interface to clear it. The recommended time to clear this bit is at the end of frame (EOF). It should be noted that RX_SFD bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO.
20	RX_EOF	RX_EOF indicates that the frame has ended and pr1_mii0/1_rxdv is de-asserted. It also validates the CRC match bit. Note there will be a small sync delay of 0ns – 5ns. The PRU must write one to clear this bit in the R31 command interface at the end of the frame. It should be noted that RX_EOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO.
19	RX_ERROR	RX_ERROR indicates one or more of the following errors occurred: • RX_MAX/MIN_FRM_CNT_ERR • RX_MAX/MIN_PRE_CNT_ERR • RX_ERR RX_ERROR is cleared by RX_ERROR_CLR.
18	WORD_RDY	WORD_RDY indicates that all four nibbles in R31 have valid data. There is a 2 clock cycle latency from the command RX_POP16 to WORD_RDY update. Therefore, firmware needs to insure it does not read WORD_RDY until 2 clock cycles after RX_POP16.
17	BYTE_RDY	BYTE_RDY indicates that the lower two nibbles in R31 have valid data. There is a 2 clock cycle latency from the command RX_POP8 to BYTE_RDY update. Therefore, PRU firmware needs to insure it does not read BYTE_RDY until 2 clock cycles after RX_POP8.

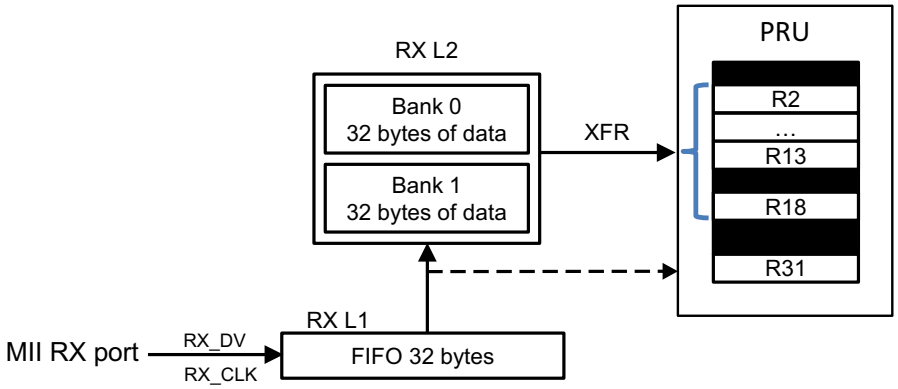
Table 4-32. PRU R31: Receive Interface Data and Status (Read Mode) (continued)

Bits	Field Name	Description
16	DATA_RDY	DATA_RDY indicates there is valid data in R31 ready to be read. This bit goes to zero when the PRU does a POP8/16 and there is no new data left in the receive MII port. This bit is high if there is more receive data for PRU to read. There is a 2 clock cycle latency from the command RX_POP16/8 to WORD_RDY/BYTE_RDY update. Therefore, PRU firmware must einsure it does not read BYTE_RDY/WORD_RDY until 2 clock cycles after RX_POP16/8.
15:8	BYTE1	Data Byte 1. This data is available such that it is safe to read by the PRU when the DATA_RDY/BYTE_RDY/WORD_RDY bits are asserted.
7:0	BYTE0	Data Byte 0. This data is available such that it is safe to read by the PRU when the DATA_RDY/BYTE_RDY/WORD_RDY bits are asserted.

4.4.6.2.3.1.2.2 RX MII Port → RX L1 FIFO → RX L2 Buffer → PRU

The RX L2 is an optional high performance buffer between the RX L1 FIFO and the PRU. Figure 4-47 illustrates the receive data path using RX L2 buffer. This data path is characterized by multi-word in flight transactions.

Figure 4-47. RX L2 to PRU Interface



The 64-byte RX L2 buffer is divided into two 32 byte banks, or ping/pong buffers. When the RX L2 is enabled, the incoming data from the MII RX port will transmit first to the 32 byte RX L1 FIFO. RX L1 pushes data into RX L2, starting when the first byte is ready until the final EOF marker. The RX L2 buffer does not apply any backpressure to the RX L1 FIFO. Therefore, it is the PRU firmware's responsibility to fetch the data in RX L2 before it is overwritten by the cyclic buffer. The RX L1 will remain near empty, with only one byte (nibble) stored.

Each RX L2 bank holds up to 32 bytes of data, and every four nibbles (or 16 bits) of data has a corresponding 8-bit status. The data and status information are stored in packed arrays. In each bank, R2 to R9 contains the data packed array and R10 to R13 contains the status packed array. Figure 4-48 shows the relationship of the data registers and status registers. The RX L2 status registers record status information about the received data, such as ERROR_CRC, RX_ERROR, STATUS_RDY, etc. The RX L2 status register details are described in Table 4-33. Note RX_RESET clears all Data and Status elements and resets R18.

Figure 4-48. Data and Status Register Dependency

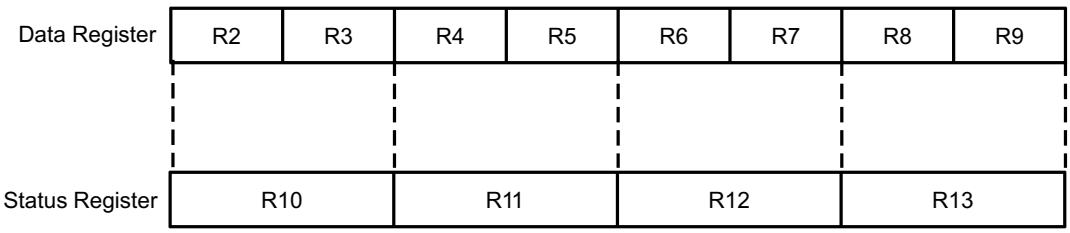


Table 4-33. RX L2 Status

Bit	Field Name	Description
7	ERROR_CRC	ERROR_CRC indicates that the frame has a CRC mismatch. This bit is valid when the RX_EOF bit is set. It should be noted that ERROR_CRC bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_CRC will only be set for one entry, self clear on next entry.
6	ERROR_NIBBLE	ERROR_NIBBLE indicates that the frame ended in odd nibble. It should be considered valid only when the RX_EOF bit and pr1_mii0/1_rxdv are set. Nibble counter is enabled post SFD event. It should be noted that ERROR_NIBBLE bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. ERROR_NIBBLE will only be set for one entry, self clear on next entry.
5	RX_SOF	RX_SOF transitions from low to high when the frame data starts to arrive and pr1_mii0/1_rxdv is asserted. Note there will be a small sync delay of 0ns – 5ns. The recommended time to clear this bit is at the end of frame (EOF). It should be noted that RX_SOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. RX_SOF will only be set for one entry, self clear on next entry.
4	RX_SFD	RX_SFD transitions from low to high when the SFD sequence (0x5D) post RX_SOF is observed on the receive MII data. The recommended time to clear this bit is at the end of frame (EOF). It should be noted that RX_SFD bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. RX_SOF will only be set for one entry, self clear on next entry.
3	RX_EOF	RX_EOF indicates that the frame has ended and pr1_mii0/1_rxdv is de-asserted. It also validates the CRC match bit. Note there will be a small sync delay of 0ns – 5ns. It should be noted that RX_EOF bit is ready in early status, which means it is calculated before data is available in RXL1 FIFO. RX_EOF will only be set for one entry, self clear on next entry.
2	RX_ERROR	RX_ERROR indicates one or more of the following errors occurred: • RX_MAX/MIN_FRM_CNT_ERR • RX_MAX/MIN_PRE_CNT_ERR • RX_ERR RX_ERROR is cleared by RX_ERROR_CLR.
1	STATUS_RDY	STATUS_RDY is set when RX_EOF or write pointer advanced by 2. This is a simple method for software to determine if RX_EOF event has occurred or new data is available. If RX_EOF is not set, all status bits are static.
0	RX_ERR	RX_ERR is set to 1 when pr1_mii0/1_rxer is asserted while pr1_mii0/1_rxdv bit is set. It will get set for first pr1_mii0/1_rxer event and self clear on SOF for the next FRAME.

Bank 0 and Bank 1 are used as ping/pong buffers. RX L2 supports the reading of a write pointer in R18 that allows software to determine which bank has active write transactions, as well as the specific write address within packed data arrays.

The PRU interacts with the RX L2 buffer using the high performance XFR read instructions and broadside interface. Table 4-34 shows the device XFR ID numbers for each bank.

Table 4-34. RX L2 XFR ID

Device ID	Function	Description
20	Selects RX L2 Bank0	R2:R9 Data packed array R10:R13 Status packed array
21	Selects RX L2 Bank1	R2:R9 Data packed array R10:R13 Status packed array

Table 4-34. RX L2 XFR ID (continued)

Device ID	Function	Description
20/21	Byte pointer of current write	R18[5:0] Pointer indicating location of current write in data packed array. 0 = Bank0.R2.Byte0 (default and reset value) 1 = Bank0.R2.Byte1 2 = Bank0.R2.Byte2 3 = Bank0.R2.Byte3 4 = Bank0.R3.Byte0 … 63=Bank1.R9.Byte3

XFR read transactions are passive and have no effect on any status or other states in RX L2. The firmware can also read R18 to determine which bank has active write transactions, and the location of the transaction. With this information, the firmware can read multiple times the stable preserved data. When RX L1 data is written to RX L2, the next status byte gets cleared at the same time the current status byte gets updated. The rest of the status buffer is persistent. When the software accesses any register of the ping/pong buffer, the software must issue an XFER read transaction to fetch the latest or current state of the ping/pong buffer. The PRU registers do not reflect the current snapshot of L2 unless an XFER is issued by the software.

4.4.6.2.4 TX MII Interface

Data to be transmitted is loaded into the TX L1 FIFO. The transmit FIFO (TX L1) stores up to 64 bytes of transmit data. From the FIFO, the data is sent to the MII TX port of the PHY by the MII_RT transmit logic.

The transmit FIFO also has the capability to be reset through software (TX_RESET). When reset, all contents of transmit FIFO are purged and this may result in a frame not getting transmitted as expected, if the transmission is already ongoing. Any new data written in the transmit FIFO results in a new frame being composed and transmitted. An overflow event will require a TX_RESET to recover from this condition.

There are four dependencies that must be true for TX_EN to assert:

1. TX L1 FIFO not empty
2. Interpacket gap (IPG) timer expiration
3. RX_DV to TX_EN timer expiration
4. TX_EN compare timer expiration

The transmit interface also provides an underflow error signal if there was no data loaded when TX_EN triggered. The transmit underflow signal is mapped to the INTC in PRU-ICSS. The PRU firmware must track the FIFO fill level, such as with a timer or the PRU cycle count register (PRU_ICSS_CTRL_CYCLE).The current FIFO fill level cannot be accessed by PRU firmware. The firmware can issue an R31 command through R31 bit 29 (TX_EOF) to indicate that the last byte has been written into the TX FIFO.

4.4.6.2.4.1 TX Data Path Options to TX L1 FIFO

There are two data path options for delivering data to the TX L1 FIFO and transmit port, described further in the subsequent sections:

1. PRU → TX L1 FIFO → TX MII port
2. RX L1 FIFO → TX L1 FIFO → TX MII port

4.4.6.2.4.1.1 PRU → TX L1 FIFO → TX MII Port

The PRU can be used to feed data into the TX L1 FIFO using the R30 and R31 registers, shown in Figure 4-49. The PRU writes up to two bytes of data into R30 and then pushes the data into the TX L1 FIFO by writing to the R31 command interface.

Figure 4-49. PRU to TX L1 Interface

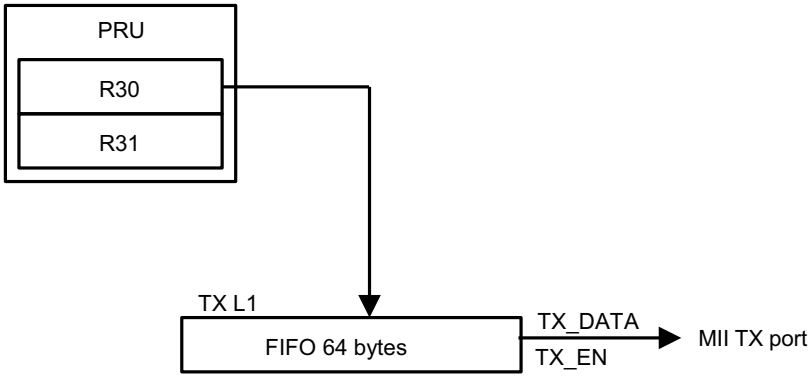


Figure 4-50 shows the R30 transmit interface. The lower 16 bits of the R30 (or FIFO transmit word) contain transmit data nibbles. The upper 16 bits contain mask information. The operation to be performed on the transmit interface is controlled by PRU writes to the R31 command interface. Table 4-35 describes the TXMASK and TXDATA bit fields of the R30 transmit interface.

Figure 4-50. PRU to TX MII Interface

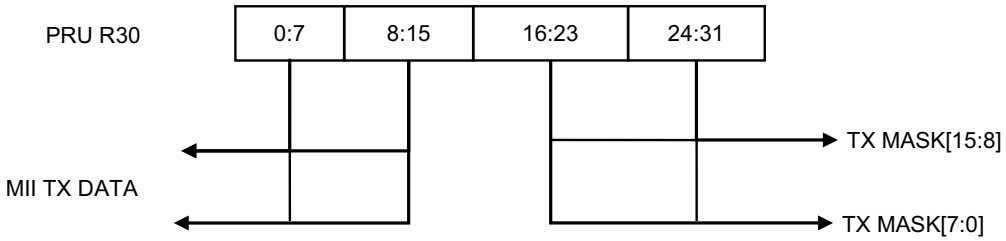


Table 4-35. PRU R30: Transmit Interface

Bits	Field Name	Description
31:16	TXMASK	The TXMASK is used to determine which of RX L1 FIFO received data and R30 data is sent to transmit FIFO. It must to be applied to TXDATA and RXDATA before it is transmitted. To disable TXMASK and transmit only TXDATA through R30, set to 0xFFFF. Note software should not pop the RXDATA from the RX L1 FIFO before pushing the TXDATA. This will cause new data to propagate before the push. Otherwise, software can pop and push on the same command for bytes only or delay the pop after the push for words or bytes.
15:0	TXDATA	Data provided by the PRU to be sent to transmit path after applying the mask. When 16 bits are to be transmitted, all bits of this and TXMASK field are used. When 8 bits are to be transmitted, the bits [7:0] of this and TXMASK field are used.

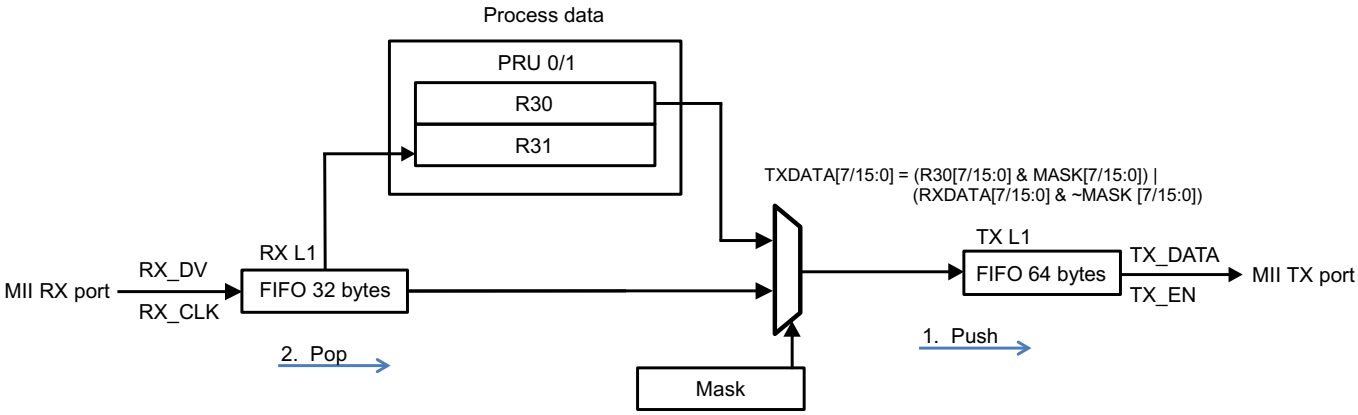
Using the TX mask, the PRU can send a mix of R30 and RX L1 FIFO data to the TX L1 FIFO. Note the TX mask is only available when the PRU is fed one word or byte at a time by the RX L1 FIFO. It is not applicable when the RX L2 buffer is enabled. To disable TX mask, set TXMASK to 0xFFFF.

As shown in Figure 4-51, the PRU drives the MII transmit interface through its R30 register. The contents of R30 and RX data from the receive interface are taken and fed into a 64 byte transmit FIFO.

Before transmission, a mask is applied to the data portion of the R30 register. By using the mask, the PRU firmware can control whether received data from the RX L1 FIFO is sent to transmit, R30 data is sent to transmit, or a mix of the two is sent. The Boolean equation that is used by MII_RT to compose TX data is:

TXDATA[7/15:0] = (R30[7/15:0] & MASK[7/15:0]) | (RXDATA[7/15:0] & ~MASK [7/15:0])

Figure 4-51. TX Mask Mode

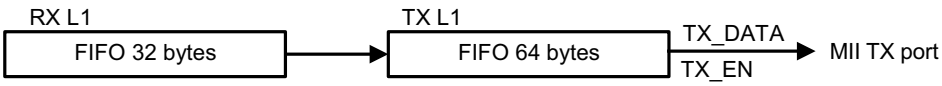


4.4.6.2.4.1.2 RX L1 FIFO → TX L1 FIFO → TX MII Port

When TXCFG0/1[TX_AUTO_SEQUENCE] is set, the data frame is passed from the RX to TX FIFOs without the any interaction of the PRU. This mode of operations is shown in Figure 4-52. The RX L1 will push into TX L1 as long as it is enabled and not full.

There is no PRU dependency in this mode and no option for the PRU to perform any operation to the TX L1 FIFO. RX_RESET clears all data and status elements.

Figure 4-52. RX L1 to TX L1 Interface



4.4.6.2.5 PRU R31 Command Interface

The PRU uses writes to R31[31:16] to control the reception and transmission of packets in register mode. Table 4-36 lists the available commands. Each bit in the table is a single clock pulse output from the PRU. When more than one action is to be performed in the same instant, the PRU firmware must set those command bits in one instruction.

Table 4-36. PRU R31: Command Interface (Write Mode)

Bit	Command	Description
31	TX_CRC_ERR	TX_CRC_ERR command when set will add 0xa5 byte to the TX L1 FIFO if the current FCS is valid. This bit can only be set with the TX_EOF command and optionally with the TX_ERROR_NIBBLE command. It cannot get set with any other commands. Note for proper operations auto-forward preamble must be enabled.
30	TX_RESET	TX_RESET command is used to reset the transmit FIFO and clear all its contents. This is required to recover from a TX FIFO overrun.
29	TX_EOF	TX_EOF command is used to indicate that the data loaded is considered last for the current frame
28	TX_ERROR_NIBBLE	TX_ERROR_NIBBLE command is used to insert an error nibble. This makes the frame invalid. Also, it will add 0x0 after the 32-bit CRC.
27	TX_CRC_HIGH	TX_CRC_HIGH command ends the CRC calculations and pushes CRC[31:16] to append to the outgoing frame in the TX L1 FIFO. Note TXCRC0/1 will become valid after 6 clock cycles.
26	TX_CRC_LOW	TX_CRC_LOW command pushes CRC[15:0] to append to the outgoing frame in the TX L1 FIFO.

Table 4-36. PRU R31: Command Interface (Write Mode) (continued)

Bit	Command	Description
25	TX_PUSH16	TX_PUSH16 command applies mask to two bytes from receive path and transmit. Note TX_PUSH16 needs to occur before TX_POP16 if data is not fully masked. TX CRC requires the data to be valid for 2 clock cycles.
24	TX_PUSH8	TX_PUSH8 command applies mask to one byte from receive path and transmit. Note TX_PUSH8 needs to occur before TX_POP8 if data is not fully masked.
23	RX_ ERROR_CLR	RX_ERROR_CLR command is used to clear RX_ ERROR indicator bit by writing 1.
22	RX_EOF_CLR	RX_EOF_CLR command is used to clear RX_EOF status indicator bit by writing 1.
21	RX_SFD_CLR	RX_SFD_CLR command is used to clear RX_SFD indicator bit by writing 1.
20	RX_SOF_CLR	RX_SOF_CLR command is used to clear RX_SOF indicator bit by writing 1.
19	Reserved	Reserved
18	RX_RESET	RX_RESET is used to reset the receive FIFO and clear all contents. This is required to recover from a RX FIFO overrun, if software does not want to undrain. The typical use case is assertion after RX_EOF. If asserted during an active frame, the following actions will occur: 1. Terminate the current frame 2. Block/terminate all new data 3. Flush/clear all FIFO elements 4. Cause RX state machine into an idle state 5. Cause EOF event 6. Cause minimum frame error, if you abort before minimum size reached
17	RX_POP16	RX_POP16 command advances the receive traffic by two bytes. This is only required when you are using R31 to read the data. After R31[15:0] is ready to read by PRU, it will set 1 to WORD_RDY, and the next new data will be allowed to advance. RX_POP16 to WORD_RDY update has 2 clock cycles latency. Firmware needs to insure it does not read WORD_RDY/BYTE_RDY until 2 clock cycles after RX_POP16.
16	RX_POP8	RX_POP8 command advances the receive traffic by one bytes. This is only required when you are using R31 to read the data. After R31[7:0] is ready to read by PRU, it will set 1 to BYTE_RDY, and the next new data will be allowed to advance. RX_POP8 to BYTE_RDY update has 2 clock cycles latency. Firmware needs to insure it does not read WORD_RDY/BYTE_RDY until 2 clock cycles after RX_POP8.

4.4.6.2.6 Other Configuration Options

4.4.6.2.6.1 Nibble and Byte Order

The PRU core is little endian. To support this difference, the MII_RT supports optional nibble swapping on both the RX and TX side.

On the receive side, the order of the two data bytes in RX R31 and the RX L2 buffer are configurable through the RX_BYTE_SWAP bit in the RXCFG0/1 registers, as shown in Table 4-37. Note the Nibble0 is the first nibble received.

Table 4-37. RX Nibble and Byte Order

Configuration	Order
RXCFG0/1[RX_BYTE_SWAP] = 0 (default)	R31[15:8] / RXL2[15:8] = Byte1{Nibble3,Nibble2} R31[7:0] / RXL2[7:0] = Byte0{Nibble1,Nibble0}

Table 4-37. RX Nibble and Byte Order (continued)

Configuration	Order
RXCFG0/1[RX_BYTE_SWAP] = 1	R31[15:8] / RXL2[15:8] = Byte0{Nibble1,Nibble0} R31[7:0] / RXL2[7:0] = Byte1{Nibble3,Nibble2}

On the transmit side, the order of the two data bytes and mask bytes in TX R30 are configurable through the TX_BYTE_SWAP bit in the TXCFG0/1 registers, as shown in Table 4-38. Note the Nibble0 is the first nibble received.

Table 4-38. TX Nibble and Byte Order

Configuration	Order
TXCFG0/1[TX_BYTE_SWAP] = 0 (default)	R30[15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30[31:24] = TX_MASK[15:8] R30[23:16] = TX_MASK[7:0]
TXCFG0/1[TX_BYTE_SWAP] = 1	R30[15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30[31:24] = TX_MASK[7:0] R30[23:16] = TX_MASK[15:8]

4.4.6.2.6.2 Preamble Source

The MII_RT module has the option to preserve and forward a received preamble in the TX data stream, use a preamble provided by the PRU, or auto-generate a preamble. These configurations are highlighted in Table 4-39.

Table 4-39. Preamble Configuration Options

RX_CUT_PREAMBLE	Determines whether RX preamble is passed onto RX L1/L2 FIFO
RX_AUTO_FWD_PRE	Determines whether RX preamble is automatically passed to TX L1 FIFO
TX_AUTO_PREAMBLE	TX interface logic auto-generates and appends preamble to TX data stream with the first push of data into the TX L1 FIFO. Enabling this option fills the TX FIFO with the preamble length, thus software must consider this to not overrun the TX FIFO.

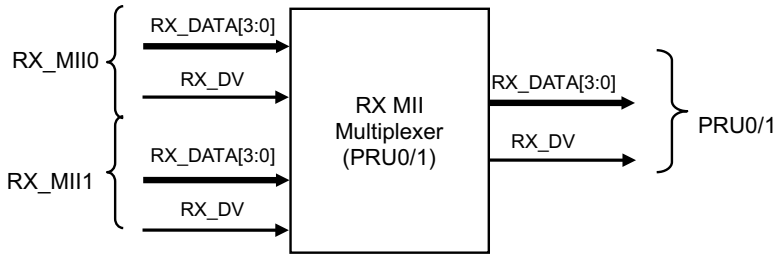
4.4.6.2.6.3 PRU and MII Port Multiplexer

The MII_RT module supports configurable PRU core to MII TXn / RXn port mapping. By default, PRU0 is mapped to TX1 and RX0 and PRU1 is mapped to TX0 and RX1. However, the system supports the flexibility to map any PRU core to any TX and RX port. Note the mapping options are destination fixed. For example, the input to PRU0 can be either RX_MII0 or RX_MII1. Similarly, the input to TX_MII0 can be either PRU0 or PRU1.

4.4.6.2.6.3.1 Receive Multiplexer

A multiplexer is provided to allow selecting either of the two MII interfaces for the receive data that is sent to PRU. Figure 4-53 shows the symbol of receive multiplexer of PRU.

Figure 4-53. MII Receive Multiplexer

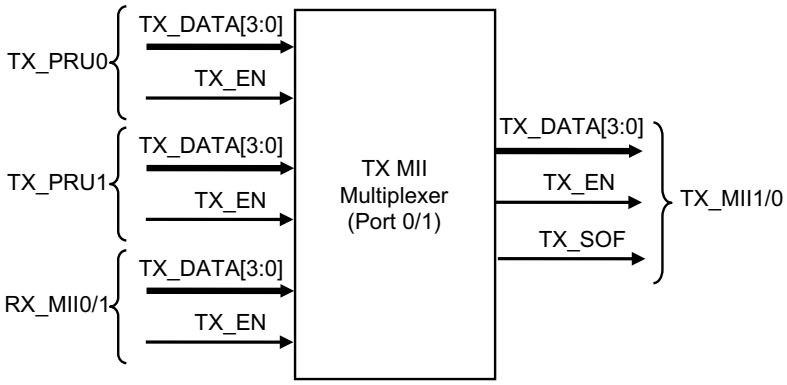


There are two receive multiplexer instances to enable selection of RX MII path for each PRU. The select lines of the RX multiplexers are driven from the PRU-ICSS programmable registers (RXCFG0/1).

4.4.6.2.6.3.2 Transmit Multiplexer

On the MII transmit ports, there is a multiplexer for each MII transmit port that enables selection of either the transmit data from the PRUs or from the RX MII interface of the other MII interface. Figure 4-54 shows the symbol of transmit multiplexer of PRU.

Figure 4-54. MII Transmit Multiplexer

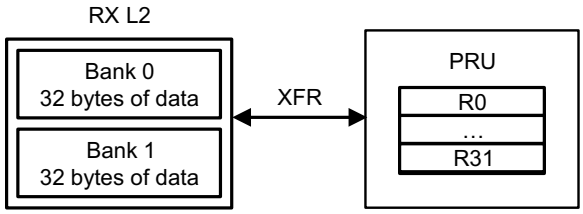


The transmit multiplexers enable the PRU-ICSS to either operate in a bypass mode where the PRU is not involved in processing MII traffic or use of one of the PRU cores for transmitting data into the MII interface. There are two instances of the TX MII multiplexer and the select lines for each TX multiplexer are provided by the PRU-ICSS. The select lines are common between register and FIFO interface. It is expected that the select lines will not change during the course of a frame so that can avoid data exchange error.

4.4.6.2.6.4 RX L2 Scratch Pad

When the RX L2 is disabled (RXCFG0/1[RX_L2_ENABLE] = 0), the RX L2 banks can be used as a generic scratch pad. In scrach pad mode, RX L2 Bank0 and RX L2 Bank1 operate like simple write/read memory mapped registers (MMRs). All XFR size and start operations are supported. RX_RESET has no effect in this mode. This mode is shown in Figure 4-55.

Figure 4-55. Scratch Pad Mode



4.4.7 MDIO

The MDIO module within the PRU-ICSS is identical to the MDIO module in Section 14.3.8.

4.5 Registers

4.5.1 PRU_ICSS_PRU_CTRL Registers

Table 4-40 lists the memory-mapped registers for the PRU_ICSS_PRU_CTRL. All register offset addresses not listed in Table 4-40 should be considered as reserved locations and the register contents should not be modified.

Table 4-40. PRU_ICSS_PRU_CTRL Registers

Offset	Acronym	Register Name	Section
0h	CTRL		Section 4.5.1.1
4h	STS		Section 4.5.1.2
8h	WAKEUP_EN		Section 4.5.1.3
Ch	CYCLE		Section 4.5.1.4
10h	STALL		Section 4.5.1.5
20h	CTBIR0		Section 4.5.1.6
24h	CTBIR1		Section 4.5.1.7
28h	CTPPR0		Section 4.5.1.8
2Ch	CTPPR1		Section 4.5.1.9

4.5.1.1 CTRL Register (offset = 0h) [reset = 1h]

CTRL is shown in Figure 4-56 and described in Table 4-41.

CONTROL REGISTER

Figure 4-56. CTRL Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-41. CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PCTR_RST_VAL	R/W	0h	Program Counter Reset Value: This field controls the address where the PRU will start executing code from after it is taken out of reset.
15	RUNSTATE	R	0h	Run State: This bit indicates whether the PRU is currently executing an instruction or is halted. 0 = PRU is halted and host has access to the instruction RAM and debug registers regions. 1 = PRU is currently running and the host is locked out of the instruction RAM and debug registers regions. This bit is used by an external debug agent to know when the PRU has actually halted when waiting for a HALT instruction to execute, a single step to finish, or any other time when the pru_enable has been cleared.
14	RESERVED	R	0h	Reserved.
13-9	RESERVED	R/W	0h	
8	SINGLE_STEP	R/W	0h	Single Step Enable: This bit controls whether or not the PRU will only execute a single instruction when enabled. 0 = PRU will free run when enabled. 1 = PRU will execute a single instruction and then the pru_enable bit will be cleared. Note that this bit does not actually enable the PRU, it only sets the policy for how much code will be run after the PRU is enabled. The pru_enable bit must be explicitly asserted. It is legal to initialize both the single_step and pru_enable bits simultaneously. (Two independent writes are not required to cause the stated functionality.)
7-4	RESERVED	R/W	0h	
3	CTR_EN	R/W	0h	PRU Cycle Counter Enable: Enables PRU cycle counters. 0 = Counters not enabled 1 = Counters enabled
2	SLEEPING	R/W	0h	PRU Sleep Indicator: This bit indicates whether or not the PRU is currently asleep. 0 = PRU is not asleep 1 = PRU is asleep If this bit is written to a 0, the PRU will be forced to power up from sleep mode.

Table 4-41. CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EN	R/W	0h	Processor Enable: This bit controls whether or not the PRU is allowed to fetch new instructions. 0 = PRU is disabled. 1 = PRU is enabled. If this bit is de-asserted while the PRU is currently running and has completed the initial cycle of a multi-cycle instruction (LBxO,SBxO,SCAN, etc.), the current instruction will be allowed to complete before the PRU pauses execution. Otherwise, the PRU will halt immediately. Because of the unpredictability/timing sensitivity of the instruction execution loop, this bit is not a reliable indication of whether or not the PRU is currently running. The pru_state bit should be consulted for an absolute indication of the run state of the core. When the PRU is halted, its internal state remains coherent therefore this bit can be reasserted without issuing a software reset and the PRU will resume processing exactly where it left off in the instruction stream.
0	SOFT_RST_N	R	1h	Soft Reset: When this bit is cleared, the PRU will be reset. This bit is set back to 1 on the next cycle after it has been cleared.

4.5.1.2 STS Register (offset = 4h) [reset = 0h]

STS is shown in Figure 4-57 and described in Table 4-42.

STATUS REGISTER

Figure 4-57. STS Register

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
RESERVED																PCTR															
R-0h																R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-42. STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	
15-0	PCTR	R	0h	Program Counter: This field is a registered (1 cycle delayed) reflection of the PRU program counter. Note that the PC is an instruction address where each instruction is a 32 bit word. This is not a byte address and to compute the byte address just multiply the PC by 4 (PC of 2 = byte address of 0x8, or PC of 8 = byte address of 0x20).

4.5.1.3 WAKEUP_EN Register (offset = 8h) [reset = 0h]

WAKEUP_EN is shown in Figure 4-58 and described in Table 4-43.

WAKEUP ENABLE REGISTER

Figure 4-58. WAKEUP_EN Register

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
BITWISE_ENS																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-43. WAKEUP_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITWISE_ENS	R/W	0h	Wakeup Enables: This field is ANDed with the incoming R31 status inputs (whose bit positions were specified in the stmap parameter) to produce a vector which is unary ORed to produce the status_wakeup source for the core. Setting any bit in this vector will allow the corresponding status input to wake up the core when it is asserted high. The PRU should set this enable vector prior to executing a SLP (sleep) instruction to ensure that the desired sources can wake up the core.

4.5.1.4 CYCLE Register (offset = Ch) [reset = 0h]

CYCLE is shown in Figure 4-59 and described in Table 4-44.

CYCLE COUNT. This register counts the number of cycles for which the PRU has been enabled.

Figure 4-59. CYCLE Register

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
CYCLECOUNT																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-44. CYCLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CYCLECOUNT		0h	This value is incremented by 1 for every cycle during which the PRU is enabled and the counter is enabled (both bits EN and CTR_EN set in the PRU control register). Counting halts while the PRU is disabled or counter is disabled, and resumes when re-eneabled. Counter clears the CTR_EN bit in the PRU control register when the count reaches 0xFFFFFFFF. (Count does does not wrap). The register can be read at any time. The register can be cleared when the counter or PRU is disabled. Clearing this register also clears the PRU Stall Count Register.

4.5.1.5 STALL Register (offset = 10h) [reset = 0h]

STALL is shown in Figure 4-60 and described in Table 4-45.

STALL COUNT. This register counts the number of cycles for which the PRU has been enabled, but unable to fetch a new instruction. It is linked to the Cycle Count Register (0x0C) such that this register reflects the stall cycles measured over the same cycles as counted by the cycle count register. Thus the value of this register is always less than or equal to cycle count.

Figure 4-60. STALL Register

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
STALLCOUNT																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-45. STALL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	STALLCOUNT		0h	This value is incremented by 1 for every cycle during which the PRU is enabled and the counter is enabled (both bits EN and CTR_EN set in the PRU control register), and the PRU was unable to fetch a new instruction for any reason.

4.5.1.6 CTBIR0 Register (offset = 20h) [reset = 0h]

CTBIR0 is shown in Figure 4-61 and described in Table 4-46.

CONSTANT TABLE BLOCK INDEX REGISTER 0. This register is used to set the block indices which are used to modify entries 24 and 25 in the PRU Constant Table. This register can be written by the PRU whenever it needs to change to a new base pointer for a block in the State / Scratchpad RAM. This function is useful since the PRU is often processing multiple processing threads which require it to change contexts. The PRU can use this register to avoid requiring excessive amounts of code for repetitive context switching. The format of this register is as follows:

Figure 4-61. CTBIR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								C25_BLK_IDX							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								C24_BLK_IDX							
R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-46. CTBIR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	
23-16	C25_BLK_IDX	R/W	0h	PRU Constant Entry 25 Block Index: This field sets the value that will appear in bits 11:8 of entry 25 in the PRU Constant Table.
15-8	RESERVED	R/W	0h	
7-0	C24_BLK_IDX	R/W	0h	PRU Constant Entry 24 Block Index: This field sets the value that will appear in bits 11:8 of entry 24 in the PRU Constant Table.

4.5.1.7 CTBIR1 Register (offset = 24h) [reset = 0h]

CTBIR1 is shown in Figure 4-62 and described in Table 4-47.

CONSTANT TABLE BLOCK INDEX REGISTER 1. This register is used to set the block indices which are used to modify entries 24 and 25 in the PRU Constant Table. This register can be written by the PRU whenever it needs to change to a new base pointer for a block in the State / Scratchpad RAM. This function is useful since the PRU is often processing multiple processing threads which require it to change contexts. The PRU can use this register to avoid requiring excessive amounts of code for repetitive context switching. The format of this register is as follows:

Figure 4-62. CTBIR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								C27_BLK_IDX							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								C26_BLK_IDX							
R/W-0h								R/W-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-47. CTBIR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R/W	0h	
23-16	C27_BLK_IDX	R/W	0h	PRU Constant Entry 27 Block Index: This field sets the value that will appear in bits 11:8 of entry 27 in the PRU Constant Table.
15-8	RESERVED	R/W	0h	
7-0	C26_BLK_IDX	R/W	0h	PRU Constant Entry 26 Block Index: This field sets the value that will appear in bits 11:8 of entry 26 in the PRU Constant Table.

4.5.1.8 CTPPR0 Register (offset = 28h) [reset = 0h]

CTPPR0 is shown in Figure 4-63 and described in Table 4-48.

CONSTANT TABLE PROGRAMMABLE POINTER REGISTER 0. This register allows the PRU to set up the 256-byte page index for entries 28 and 29 in the PRU Constant Table which serve as general purpose pointers which can be configured to point to any locations inside the session router address map. This register is useful when the PRU needs to frequently access certain structures inside the session router address space whose locations are not hard coded such as tables in scratchpad memory. This register is formatted as follows:

Figure 4-63. CTPPR0 Register

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
C29_POINTER																C28_POINTER															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-48. CTPPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	C29_POINTER	R/W	0h	PRU Constant Entry 29 Pointer: This field sets the value that will appear in bits 23:8 of entry 29 in the PRU Constant Table.
15-0	C28_POINTER	R/W	0h	PRU Constant Entry 28 Pointer: This field sets the value that will appear in bits 23:8 of entry 28 in the PRU Constant Table.

4.5.1.9 CTPPR1 Register (offset = 2Ch) [reset = 0h]

CTPPR1 is shown in Figure 4-64 and described in Table 4-49.

CONSTANT TABLE PROGRAMMABLE POINTER REGISTER 1. This register functions the same as the PRU Constant Table Programmable Pointer Register 0 but allows the PRU to control entries 30 and 31 in the PRU Constant Table. This register is formatted as follows:

Figure 4-64. CTPPR1 Register

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
C31_POINTER																C30_POINTER															
R/W-0h																R/W-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-49. CTPPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	C31_POINTER	R/W	0h	PRU Constant Entry 31 Pointer: This field sets the value that will appear in bits 23:8 of entry 31 in the PRU Constant Table.
15-0	C30_POINTER	R/W	0h	PRU Constant Entry 30 Pointer: This field sets the value that will appear in bits 23:8 of entry 30 in the PRU Constant Table.

4.5.2 PRU_ICSS_PRU_DEBUG Registers

Table 4-50 lists the memory-mapped registers for the PRU_ICSS_PRU_DEBUG. All register offset addresses not listed in Table 4-50 should be considered as reserved locations and the register contents should not be modified.

Table 4-50. PRU_ICSS_PRU_DEBUG Registers

Offset	Acronym	Register Name	Section
0h	GPREG0		Section 4.5.2.1
4h	GPREG1		Section 4.5.2.2
8h	GPREG2		Section 4.5.2.3
Ch	GPREG3		Section 4.5.2.4
10h	GPREG4		Section 4.5.2.5
14h	GPREG5		Section 4.5.2.6
18h	GPREG6		Section 4.5.2.7
1Ch	GPREG7		Section 4.5.2.8
20h	GPREG8		Section 4.5.2.9
24h	GPREG9		Section 4.5.2.10
28h	GPREG10		Section 4.5.2.11
2Ch	GPREG11		Section 4.5.2.12
30h	GPREG12		Section 4.5.2.13
34h	GPREG13		Section 4.5.2.14
38h	GPREG14		Section 4.5.2.15
3Ch	GPREG15		Section 4.5.2.16
40h	GPREG16		Section 4.5.2.17
44h	GPREG17		Section 4.5.2.18
48h	GPREG18		Section 4.5.2.19
4Ch	GPREG19		Section 4.5.2.20
50h	GPREG20		Section 4.5.2.21
54h	GPREG21		Section 4.5.2.22
58h	GPREG22		Section 4.5.2.23

Table 4-50. PRU_ICSS_PRU_DEBUG Registers (continued)

Offset	Acronym	Register Name	Section
5Ch	GPREG23		Section 4.5.2.24
60h	GPREG24		Section 4.5.2.25
64h	GPREG25		Section 4.5.2.26
68h	GPREG26		Section 4.5.2.27
6Ch	GPREG27		Section 4.5.2.28
70h	GPREG28		Section 4.5.2.29
74h	GPREG29		Section 4.5.2.30
78h	GPREG30		Section 4.5.2.31
7Ch	GPREG31		Section 4.5.2.32
80h	CT_REG0		Section 4.5.2.33
84h	CT_REG1		Section 4.5.2.34
88h	CT_REG2		Section 4.5.2.35
8Ch	CT_REG3		Section 4.5.2.36
90h	CT_REG4		Section 4.5.2.37
94h	CT_REG5		Section 4.5.2.38
98h	CT_REG6		Section 4.5.2.39
9Ch	CT_REG7		Section 4.5.2.40
A0h	CT_REG8		Section 4.5.2.41
A4h	CT_REG9		Section 4.5.2.42
A8h	CT_REG10		Section 4.5.2.43
ACh	CT_REG11		Section 4.5.2.44
B0h	CT_REG12		Section 4.5.2.45
B4h	CT_REG13		Section 4.5.2.46
B8h	CT_REG14		Section 4.5.2.47
BCh	CT_REG15		Section 4.5.2.48
C0h	CT_REG16		Section 4.5.2.49
C4h	CT_REG17		Section 4.5.2.50
C8h	CT_REG18		Section 4.5.2.51
CCh	CT_REG19		Section 4.5.2.52
D0h	CT_REG20		Section 4.5.2.53
D4h	CT_REG21		Section 4.5.2.54
D8h	CT_REG22		Section 4.5.2.55
DCh	CT_REG23		Section 4.5.2.56
E0h	CT_REG24		Section 4.5.2.57
E4h	CT_REG25		Section 4.5.2.58
E8h	CT_REG26		Section 4.5.2.59
ECh	CT_REG27		Section 4.5.2.60
F0h	CT_REG28		Section 4.5.2.61
F4h	CT_REG29		Section 4.5.2.62
F8h	CT_REG30		Section 4.5.2.63
FCh	CT_REG31		Section 4.5.2.64

4.5.2.1 GPREG0 Register (offset = 0h) [reset = 0h]

GPREG0 is shown in Figure 4-65 and described in Table 4-51.

DEBUG PRU GENERAL PURPOSE REGISTER 0. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-65. GPREG0 Register

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
GP_REG0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-51. GPREG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG0	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.2 GPREG1 Register (offset = 4h) [reset = 0h]

GPREG1 is shown in Figure 4-66 and described in Table 4-52.

DEBUG PRU GENERAL PURPOSE REGISTER 1. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-66. GPREG1 Register

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
GP_REG1																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-52. GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG1	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.3 GPREG2 Register (offset = 8h) [reset = 0h]

GPREG2 is shown in Figure 4-67 and described in Table 4-53.

DEBUG PRU GENERAL PURPOSE REGISTER 2. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-67. GPREG2 Register

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
GP_REG2																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-53. GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG2	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.4 GPREG3 Register (offset = Ch) [reset = 0h]

GPREG3 is shown in Figure 4-68 and described in Table 4-54.

DEBUG PRU GENERAL PURPOSE REGISTER 3. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-68. GPREG3 Register

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
GP_REG3																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-54. GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG3	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.5 GPREG4 Register (offset = 10h) [reset = 0h]

GPREG4 is shown in Figure 4-69 and described in Table 4-55.

DEBUG PRU GENERAL PURPOSE REGISTER 4. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-69. GPREG4 Register

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
GP_REG4																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-55. GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG4	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.6 GPREG5 Register (offset = 14h) [reset = 0h]

GPREG5 is shown in Figure 4-70 and described in Table 4-56.

DEBUG PRU GENERAL PURPOSE REGISTER 5. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-70. GPREG5 Register

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
GP_REG5																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-56. GPREG5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG5	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.7 GPREG6 Register (offset = 18h) [reset = 0h]

GPREG6 is shown in Figure 4-71 and described in Table 4-57.

DEBUG PRU GENERAL PURPOSE REGISTER 6. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-71. GPREG6 Register

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
GP_REG6																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-57. GPREG6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG6	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.8 GPREG7 Register (offset = 1Ch) [reset = 0h]

GPREG7 is shown in Figure 4-72 and described in Table 4-58.

DEBUG PRU GENERAL PURPOSE REGISTER 7. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-72. GPREG7 Register

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
GP_REG7																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-58. GPREG7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG7	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.9 GPREG8 Register (offset = 20h) [reset = 0h]

GPREG8 is shown in Figure 4-73 and described in Table 4-59.

DEBUG PRU GENERAL PURPOSE REGISTER 8. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-73. GPREG8 Register

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
GP_REG8																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-59. GPREG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG8	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.10 GPREG9 Register (offset = 24h) [reset = 0h]

GPREG9 is shown in Figure 4-74 and described in Table 4-60.

DEBUG PRU GENERAL PURPOSE REGISTER 9. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-74. GPREG9 Register

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
GP_REG9																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-60. GPREG9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG9	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.11 GPREG10 Register (offset = 28h) [reset = 0h]

GPREG10 is shown in Figure 4-75 and described in Table 4-61.

DEBUG PRU GENERAL PURPOSE REGISTER 10. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-75. GPREG10 Register

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
GP_REG10																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-61. GPREG10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG10	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.12 GPREG11 Register (offset = 2Ch) [reset = 0h]

GPREG11 is shown in Figure 4-76 and described in Table 4-62.

DEBUG PRU GENERAL PURPOSE REGISTER 11. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-76. GPREG11 Register

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
GP_REG11																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-62. GPREG11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG11	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.13 GPREG12 Register (offset = 30h) [reset = 0h]

GPREG12 is shown in Figure 4-77 and described in Table 4-63.

DEBUG PRU GENERAL PURPOSE REGISTER 12. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-77. GPREG12 Register

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
GP_REG12																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-63. GPREG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG12	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.14 GPREG13 Register (offset = 34h) [reset = 0h]

GPREG13 is shown in Figure 4-78 and described in Table 4-64.

DEBUG PRU GENERAL PURPOSE REGISTER 13. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-78. GPREG13 Register

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
GP_REG13																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-64. GPREG13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG13	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.15 GPREG14 Register (offset = 38h) [reset = 0h]

GPREG14 is shown in Figure 4-79 and described in Table 4-65.

DEBUG PRU GENERAL PURPOSE REGISTER 14. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-79. GPREG14 Register

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
GP_REG14																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-65. GPREG14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG14	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.16 GPREG15 Register (offset = 3Ch) [reset = 0h]

GPREG15 is shown in Figure 4-80 and described in Table 4-66.

DEBUG PRU GENERAL PURPOSE REGISTER 15. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-80. GPREG15 Register

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
GP_REG15																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-66. GPREG15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG15	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.17 GPREG16 Register (offset = 40h) [reset = 0h]

GPREG16 is shown in Figure 4-81 and described in Table 4-67.

DEBUG PRU GENERAL PURPOSE REGISTER 16. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-81. GPREG16 Register

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
GP_REG16																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-67. GPREG16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG16	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.18 GPREG17 Register (offset = 44h) [reset = 0h]

GPREG17 is shown in Figure 4-82 and described in Table 4-68.

DEBUG PRU GENERAL PURPOSE REGISTER 17. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-82. GPREG17 Register

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
GP_REG17																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-68. GPREG17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG17	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.19 GPREG18 Register (offset = 48h) [reset = 0h]

GPREG18 is shown in Figure 4-83 and described in Table 4-69.

DEBUG PRU GENERAL PURPOSE REGISTER 18. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-83. GPREG18 Register

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
GP_REG18																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-69. GPREG18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG18	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.20 GPREG19 Register (offset = 4Ch) [reset = 0h]

GPREG19 is shown in Figure 4-84 and described in Table 4-70.

DEBUG PRU GENERAL PURPOSE REGISTER 19. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-84. GPREG19 Register

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
GP_REG19																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-70. GPREG19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG19	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.21 GPREG20 Register (offset = 50h) [reset = 0h]

GPREG20 is shown in Figure 4-85 and described in Table 4-71.

DEBUG PRU GENERAL PURPOSE REGISTER 20. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-85. GPREG20 Register

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
GP_REG20																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-71. GPREG20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG20	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.22 GPREG21 Register (offset = 54h) [reset = 0h]

GPREG21 is shown in Figure 4-86 and described in Table 4-72.

DEBUG PRU GENERAL PURPOSE REGISTER 21. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-86. GPREG21 Register

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
GP_REG21																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-72. GPREG21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG21	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.23 GPREG22 Register (offset = 58h) [reset = 0h]

GPREG22 is shown in Figure 4-87 and described in Table 4-73.

DEBUG PRU GENERAL PURPOSE REGISTER 22. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-87. GPREG22 Register

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
GP_REG22																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-73. GPREG22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG22	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.24 GPREG23 Register (offset = 5Ch) [reset = 0h]

GPREG23 is shown in Figure 4-88 and described in Table 4-74.

DEBUG PRU GENERAL PURPOSE REGISTER 23. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-88. GPREG23 Register

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
GP_REG23																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-74. GPREG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG23	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.25 GPREG24 Register (offset = 60h) [reset = 0h]

GPREG24 is shown in Figure 4-89 and described in Table 4-75.

DEBUG PRU GENERAL PURPOSE REGISTER 24. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-89. GPREG24 Register

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
GP_REG24																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-75. GPREG24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG24	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.26 GPREG25 Register (offset = 64h) [reset = 0h]

GPREG25 is shown in Figure 4-90 and described in Table 4-76.

DEBUG PRU GENERAL PURPOSE REGISTER 25. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-90. GPREG25 Register

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
GP_REG25																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-76. GPREG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG25	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.27 GPREG26 Register (offset = 68h) [reset = 0h]

GPREG26 is shown in Figure 4-91 and described in Table 4-77.

DEBUG PRU GENERAL PURPOSE REGISTER 26. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-91. GPREG26 Register

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
GP_REG26																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-77. GPREG26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG26	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.28 GPREG27 Register (offset = 6Ch) [reset = 0h]

GPREG27 is shown in Figure 4-92 and described in Table 4-78.

DEBUG PRU GENERAL PURPOSE REGISTER 27. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-92. GPREG27 Register

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
GP_REG27																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-78. GPREG27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG27	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.29 GPREG28 Register (offset = 70h) [reset = 0h]

GPREG28 is shown in Figure 4-93 and described in Table 4-79.

DEBUG PRU GENERAL PURPOSE REGISTER 28. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-93. GPREG28 Register

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
GP_REG28																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-79. GPREG28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG28	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.30 GPREG29 Register (offset = 74h) [reset = 0h]

GPREG29 is shown in Figure 4-94 and described in Table 4-80.

DEBUG PRU GENERAL PURPOSE REGISTER 29. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-94. GPREG29 Register

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
GP_REG29																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-80. GPREG29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG29	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.31 GPREG30 Register (offset = 78h) [reset = 0h]

GPREG30 is shown in Figure 4-95 and described in Table 4-81.

DEBUG PRU GENERAL PURPOSE REGISTER 30. This register allows an external agent to debug the PRU while it is disabled. Reading or writing to these registers will have the same effect as a read or write to these registers from an internal instruction in the PRU. For R30, this includes generation of the pulse outputs whenever the register is written.

Figure 4-95. GPREG30 Register

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
GP_REG30																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-81. GPREG30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG30	R/W	0h	PRU Internal GP Register n: Reading / writing this field directly inspects/modifies the corresponding internal register in the PRU internal regfile.

4.5.2.32 GPREG31 Register (offset = 7Ch) [reset = 0h]

GPREG31 is shown in Figure 4-96 and described in Table 4-82.

Figure 4-96. GPREG31 Register

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
GP_REG31																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-82. GPREG31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GP_REG31	R/W	0h	

4.5.2.33 CT_REG0 Register (offset = 80h) [reset = 20000h]

CT_REG0 is shown in Figure 4-97 and described in Table 4-83.

DEBUG PRU CONSTANTS TABLE ENTRY 0. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-97. CT_REG0 Register

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
CT_REG0																															
R-20000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-83. CT_REG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG0	R	20000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.34 CT_REG1 Register (offset = 84h) [reset = 48040000h]

CT_REG1 is shown in Figure 4-98 and described in Table 4-84.

DEBUG PRU CONSTANTS TABLE ENTRY 1. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-98. CT_REG1 Register

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
CT_REG1																															
R-48040000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-84. CT_REG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG1	R	48040000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.35 CT_REG2 Register (offset = 88h) [reset = 4802A000h]

CT_REG2 is shown in Figure 4-99 and described in Table 4-85.

DEBUG PRU CONSTANTS TABLE ENTRY 2. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-99. CT_REG2 Register

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
CT_REG2																															
R-4802A000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-85. CT_REG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG2	R	4802A000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.36 CT_REG3 Register (offset = 8Ch) [reset = 30000h]

CT_REG3 is shown in Figure 4-100 and described in Table 4-86.

DEBUG PRU CONSTANTS TABLE ENTRY 3. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-100. CT_REG3 Register

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
CT_REG3																															
R-30000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-86. CT_REG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG3	R	30000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.37 CT_REG4 Register (offset = 90h) [reset = 26000h]

CT_REG4 is shown in Figure 4-101 and described in Table 4-87.

DEBUG PRU CONSTANTS TABLE ENTRY 4. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-101. CT_REG4 Register

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
CT_REG4																															
R-26000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-87. CT_REG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG4	R	26000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.38 CT_REG5 Register (offset = 94h) [reset = 48060000h]

CT_REG5 is shown in Figure 4-102 and described in Table 4-88.

DEBUG PRU CONSTANTS TABLE ENTRY 5. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-102. CT_REG5 Register

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
CT_REG5																															
R-48060000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-88. CT_REG5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG5	R	48060000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.39 CT_REG6 Register (offset = 98h) [reset = 48030000h]

CT_REG6 is shown in Figure 4-103 and described in Table 4-89.

DEBUG PRU CONSTANTS TABLE ENTRY 6. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-103. CT_REG6 Register

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
CT_REG6																															
R-48030000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-89. CT_REG6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG6	R	48030000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.40 CT_REG7 Register (offset = 9Ch) [reset = 28000h]

CT_REG7 is shown in Figure 4-104 and described in Table 4-90.

DEBUG PRU CONSTANTS TABLE ENTRY 7. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-104. CT_REG7 Register

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
CT_REG7																															
R-28000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-90. CT_REG7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG7	R	28000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.41 CT_REG8 Register (offset = A0h) [reset = 46000000h]

CT_REG8 is shown in Figure 4-105 and described in Table 4-91.

DEBUG PRU CONSTANTS TABLE ENTRY 8. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-105. CT_REG8 Register

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
CT_REG8																															
R-46000000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-91. CT_REG8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG8	R	46000000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.42 CT_REG9 Register (offset = A4h) [reset = 4A100000h]

CT_REG9 is shown in Figure 4-106 and described in Table 4-92.

DEBUG PRU CONSTANTS TABLE ENTRY 9. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-106. CT_REG9 Register

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
CT_REG9																															
R-4A100000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-92. CT_REG9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG9	R	4A100000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.43 CT_REG10 Register (offset = A8h) [reset = 48318000h]

CT_REG10 is shown in Figure 4-107 and described in Table 4-93.

DEBUG PRU CONSTANTS TABLE ENTRY 10. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-107. CT_REG10 Register

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
CT_REG10																															
R-48318000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-93. CT_REG10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG10	R	48318000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.44 CT_REG11 Register (offset = ACh) [reset = 48022000h]

CT_REG11 is shown in Figure 4-108 and described in Table 4-94.

DEBUG PRU CONSTANTS TABLE ENTRY 11. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-108. CT_REG11 Register

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
CT_REG11																															
R-48022000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-94. CT_REG11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG11	R	48022000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.45 CT_REG12 Register (offset = B0h) [reset = 48024000h]

CT_REG12 is shown in Figure 4-109 and described in Table 4-95.

DEBUG PRU CONSTANTS TABLE ENTRY 12. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-109. CT_REG12 Register

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
CT_REG12																															
R-48024000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-95. CT_REG12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG12	R	48024000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.46 CT_REG13 Register (offset = B4h) [reset = 48310000h]

CT_REG13 is shown in Figure 4-110 and described in Table 4-96.

DEBUG PRU CONSTANTS TABLE ENTRY 13. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-110. CT_REG13 Register

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
CT_REG13																															
R-48310000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-96. CT_REG13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG13	R	48310000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.47 CT_REG14 Register (offset = B8h) [reset = 481CC000h]

CT_REG14 is shown in Figure 4-111 and described in Table 4-97.

DEBUG PRU CONSTANTS TABLE ENTRY 14. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-111. CT_REG14 Register

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
CT_REG14																															
R-481CC000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-97. CT_REG14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG14	R	481CC000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.48 CT_REG15 Register (offset = BCh) [reset = 481D0000h]

CT_REG15 is shown in Figure 4-112 and described in Table 4-98.

DEBUG PRU CONSTANTS TABLE ENTRY 15. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-112. CT_REG15 Register

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
CT_REG15																															
R-481D0000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-98. CT_REG15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG15	R	481D0000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.49 CT_REG16 Register (offset = C0h) [reset = 481A0000h]

CT_REG16 is shown in Figure 4-113 and described in Table 4-99.

DEBUG PRU CONSTANTS TABLE ENTRY 16. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-113. CT_REG16 Register

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
CT_REG16																															
R-481A0000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-99. CT_REG16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG16	R	481A0000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.50 CT_REG17 Register (offset = C4h) [reset = 4819C000h]

CT_REG17 is shown in Figure 4-114 and described in Table 4-100.

DEBUG PRU CONSTANTS TABLE ENTRY 17. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-114. CT_REG17 Register

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
CT_REG17																															
R-4819C000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-100. CT_REG17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG17	R	4819C000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.51 CT_REG18 Register (offset = C8h) [reset = 48300000h]

CT_REG18 is shown in Figure 4-115 and described in Table 4-101.

DEBUG PRU CONSTANTS TABLE ENTRY 18. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-115. CT_REG18 Register

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
CT_REG18																															
R-48300000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-101. CT_REG18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG18	R	48300000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.52 CT_REG19 Register (offset = CCh) [reset = 48302000h]

CT_REG19 is shown in Figure 4-116 and described in Table 4-102.

DEBUG PRU CONSTANTS TABLE ENTRY 19. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-116. CT_REG19 Register

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
CT_REG19																															
R-48302000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-102. CT_REG19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG19	R	48302000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.53 CT_REG20 Register (offset = D0h) [reset = 48304000h]

CT_REG20 is shown in Figure 4-117 and described in Table 4-103.

DEBUG PRU CONSTANTS TABLE ENTRY 20. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-117. CT_REG20 Register

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
CT_REG20																															
R-48304000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-103. CT_REG20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG20	R	48304000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.54 CT_REG21 Register (offset = D4h) [reset = 32400h]

CT_REG21 is shown in Figure 4-118 and described in Table 4-104.

DEBUG PRU CONSTANTS TABLE ENTRY 21. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-118. CT_REG21 Register

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
CT_REG21																															
R-32400h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-104. CT_REG21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG21	R	32400h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.55 CT_REG22 Register (offset = D8h) [reset = 480C8000h]

CT_REG22 is shown in Figure 4-119 and described in Table 4-105.

DEBUG PRU CONSTANTS TABLE ENTRY 22. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-119. CT_REG22 Register

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
CT_REG22																															
R-480C8000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-105. CT_REG22 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG22	R	480C8000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.56 CT_REG23 Register (offset = DCh) [reset = 480CA000h]

CT_REG23 is shown in Figure 4-120 and described in Table 4-106.

DEBUG PRU CONSTANTS TABLE ENTRY 23. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-120. CT_REG23 Register

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
CT_REG23																															
R-480CA000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-106. CT_REG23 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG23	R	480CA000h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table.

4.5.2.57 CT_REG24 Register (offset = E0h) [reset = 0h]

CT_REG24 is shown in Figure 4-121 and described in Table 4-107.

DEBUG PRU CONSTANTS TABLE ENTRY 24. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-121. CT_REG24 Register

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
CT_REG24																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-107. CT_REG24 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG24	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C24_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00000n00, n=C24_BLK_INDEX[3:0].

4.5.2.58 CT_REG25 Register (offset = E4h) [reset = 0h]

CT_REG25 is shown in Figure 4-122 and described in Table 4-108.

DEBUG PRU CONSTANTS TABLE ENTRY 25. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-122. CT_REG25 Register

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
CT_REG25																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-108. CT_REG25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG25	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C25_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00002n00, n=C25_BLK_INDEX[3:0].

4.5.2.59 CT_REG26 Register (offset = E8h) [reset = 0h]

CT_REG26 is shown in Figure 4-123 and described in Table 4-109.

DEBUG PRU CONSTANTS TABLE ENTRY 26. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-123. CT_REG26 Register

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
CT_REG26																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-109. CT_REG26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG26	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C26_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x0002En00, n=C26_BLK_INDEX[3:0].

4.5.2.60 CT_REG27 Register (offset = ECh) [reset = 0h]

CT_REG27 is shown in Figure 4-124 and described in Table 4-110.

DEBUG PRU CONSTANTS TABLE ENTRY 27. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-124. CT_REG27 Register

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
CT_REG27																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-110. CT_REG27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG27	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C27_BLK_INDEX in the PRU Control register. The reset value for this Constant Table Entry is 0x00032n00, n=C27_BLK_INDEX[3:0].

4.5.2.61 CT_REG28 Register (offset = F0h) [reset = 0h]

CT_REG28 is shown in Figure 4-125 and described in Table 4-111.

DEBUG PRU CONSTANTS TABLE ENTRY 28. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-125. CT_REG28 Register

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
CT_REG28																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-111. CT_REG28 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG28	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C28_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x00nnnn00, nnnn=C28_POINTER[15:0].

4.5.2.62 CT_REG29 Register (offset = F4h) [reset = 0h]

CT_REG29 is shown in Figure 4-126 and described in Table 4-112.

DEBUG PRU CONSTANTS TABLE ENTRY 29. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-126. CT_REG29 Register

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
CT_REG29																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-112. CT_REG29 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG29	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C29_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x49nnnn00, nnnn=C29_POINTER[15:0].

4.5.2.63 CT_REG30 Register (offset = F8h) [reset = 0h]

CT_REG30 is shown in Figure 4-127 and described in Table 4-113.

DEBUG PRU CONSTANTS TABLE ENTRY 30. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-127. CT_REG30 Register

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
CT_REG30																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-113. CT_REG30 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG30	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C30_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x40nnnn00, nnnn=C30_POINTER[15:0].

4.5.2.64 CT_REG31 Register (offset = FCh) [reset = 0h]

CT_REG31 is shown in Figure 4-128 and described in Table 4-114.

DEBUG PRU CONSTANTS TABLE ENTRY 31. This register allows an external agent to debug the PRU while it is disabled. Since some of the constants table entries may actually depend on system inputs / and or the internal state of the PRU, these registers are provided to allow an external agent to easily determine the state of the constants table.

Figure 4-128. CT_REG31 Register

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
CT_REG31																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-114. CT_REG31 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CT_REG31	R	0h	PRU Internal Constants Table Entry n: Reading this field directly inspects the corresponding entry in the PRU internal constants table. This entry is partially programmable through the C31_POINTER in the PRU Control register. The reset value for this Constant Table Entry is 0x80nnnn00, nnnn=C31_POINTER[15:0].

4.5.3 PRU_ICSS_INTC Registers

Table 4-115 lists the memory-mapped registers for the PRU_ICSS_INTC. All register offset addresses not listed in Table 4-115 should be considered as reserved locations and the register contents should not be modified.

Table 4-115. PRU_ICSS_INTC Registers

Offset	Acronym	Register Name	Section
0h	REVID		Section 4.5.3.1
4h	CR		Section 4.5.3.2
10h	GER		Section 4.5.3.3
1Ch	GNLR		Section 4.5.3.4
20h	SISR		Section 4.5.3.5
24h	SICR		Section 4.5.3.6
28h	EISR		Section 4.5.3.7
2Ch	EICR		Section 4.5.3.8
34h	HIEISR		Section 4.5.3.9
38h	HIDISR		Section 4.5.3.10
80h	GPIR		Section 4.5.3.11
200h	SRSR0		Section 4.5.3.12
204h	SRSR1		Section 4.5.3.13
280h	SECR0		Section 4.5.3.14
284h	SECR1		Section 4.5.3.15
300h	ESR0		Section 4.5.3.16
304h	ESR1		Section 4.5.3.17
380h	ECR0		Section 4.5.3.18
384h	ECR1		Section 4.5.3.19
400h	CMR0		Section 4.5.3.20
404h	CMR1		Section 4.5.3.21
408h	CMR2		Section 4.5.3.22

Table 4-115. PRU_ICSS_INTC Registers (continued)

Offset	Acronym	Register Name	Section
40Ch	CMR3		Section 4.5.3.23
410h	CMR4		Section 4.5.3.24
414h	CMR5		Section 4.5.3.25
418h	CMR6		Section 4.5.3.26
41Ch	CMR7		Section 4.5.3.27
420h	CMR8		Section 4.5.3.28
424h	CMR9		Section 4.5.3.29
428h	CMR10		Section 4.5.3.30
42Ch	CMR11		Section 4.5.3.31
430h	CMR12		Section 4.5.3.32
434h	CMR13		Section 4.5.3.33
438h	CMR14		Section 4.5.3.34
43Ch	CMR15		Section 4.5.3.35
800h	HMR0		Section 4.5.3.36
804h	HMR1		Section 4.5.3.37
808h	HMR2		Section 4.5.3.38
900h	HIPIR0		Section 4.5.3.39
904h	HIPIR1		Section 4.5.3.40
908h	HIPIR2		Section 4.5.3.41
90Ch	HIPIR3		Section 4.5.3.42
910h	HIPIR4		Section 4.5.3.43
914h	HIPIR5		Section 4.5.3.44
918h	HIPIR6		Section 4.5.3.45
91Ch	HIPIR7		Section 4.5.3.46
920h	HIPIR8		Section 4.5.3.47
924h	HIPIR9		Section 4.5.3.48
D00h	SIPR0		Section 4.5.3.49
D04h	SIPR1		Section 4.5.3.50
D80h	SITR0		Section 4.5.3.51
D84h	SITR1		Section 4.5.3.52
1100h	HINLR0		Section 4.5.3.53
1104h	HINLR1		Section 4.5.3.54
1108h	HINLR2		Section 4.5.3.55
110Ch	HINLR3		Section 4.5.3.56
1110h	HINLR4		Section 4.5.3.57
1114h	HINLR5		Section 4.5.3.58
1118h	HINLR6		Section 4.5.3.59
111Ch	HINLR7		Section 4.5.3.60
1120h	HINLR8		Section 4.5.3.61
1124h	HINLR9		Section 4.5.3.62
1500h	HIER		Section 4.5.3.63

4.5.3.1 REVID Register (offset = 0h) [reset = 4E82A900h]

REVID is shown in Figure 4-129 and described in Table 4-116.

Revision ID Register

Figure 4-129. REVID Register

31	30	29	28	27	26	25	24
REV_SCHEME		RESERVED		REV_MODULE			
R-1h		R-0h		R-E82h			
23	22	21	20	19	18	17	16
REV_MODULE							
R-E82h							
15	14	13	12	11	10	9	8
REV_RTL					REV_MAJOR		
R-15h					R-1h		
7	6	5	4	3	2	1	0
REV_CUSTOM		REV_MINOR					
R-0h		R-0h					

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-116. REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	REV_SCHEME	R	1h	SCHEME
29-28	RESERVED	R	0h	
27-16	REV_MODULE	R	E82h	MODULE ID
15-11	REV_RTL	R	15h	RTL REVISIONS
10-8	REV_MAJOR	R	1h	MAJOR REVISION
7-6	REV_CUSTOM	R	0h	CUSTOM REVISION
5-0	REV_MINOR	R	0h	MINOR REVISION

4.5.3.2 CR Register (offset = 4h) [reset = 0h]

CR is shown in Figure 4-130 and described in Table 4-117.

The Control Register holds global control parameters and can forces a soft reset on the module.

Figure 4-130. CR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-117. CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	
4	RESERVED	R	0h	Reserved.
3-2	NEST_MODE	R/W	0h	The nesting mode. 0 = no nesting 1 = automatic individual nesting (per host interrupt) 2 = automatic global nesting (over all host interrupts) 3 = manual nesting
1	RESERVED	R/W	0h	Reserved.
0	RESERVED	R/W	0h	

4.5.3.3 GER Register (offset = 10h) [reset = 0h]

GER is shown in Figure 4-131 and described in Table 4-118.

The Global Host Interrupt Enable Register enables all the host interrupts. Individual host interrupts are still enabled or disabled from their individual enables and are not overridden by the global enable.

Figure 4-131. GER Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-118. GER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	EN_HINT_ANY	R/W	0h	The current global enable value when read. Writes set the global enable.

4.5.3.4 GNLR Register (offset = 1Ch) [reset = 100h]

GNLR is shown in Figure 4-132 and described in Table 4-119.

The Global Nesting Level Register allows the checking and setting of the global nesting level across all host interrupts when automatic global nesting mode is set. The nesting level is the channel (and all of lower priority) that are nested out because of a current interrupt. This register is only available when nesting is configured.

Figure 4-132. GNLR Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							GLB_NEST_LEVEL
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
GLB_NEST_LEVEL							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-119. GNLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Always read as 0. Writes of 1 override the automatic nesting and set the nesting_level to the written data.
30-9	RESERVED	R/W	0h	
8-0	GLB_NEST_LEVEL	R/W	100h	The current global nesting level (highest channel that is nested). Writes set the nesting level. In auto nesting mode this value is updated internally unless the auto_override bit is set.

4.5.3.5 SISR Register (offset = 20h) [reset = 0h]

SISR is shown in Figure 4-133 and described in Table 4-120.

The System Event Status Indexed Set Register allows setting the status of an event. The event to set is the index value written (0-63). This sets the Raw Status Register bit of the given index.

Figure 4-133. SISR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-120. SISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	W	0h	
9-0	STS_SET_IDX	W	0h	Writes set the status of the event given in the index value (0-63). Reads return 0.

4.5.3.6 SICR Register (offset = 24h) [reset = 0h]

SICR is shown in Figure 4-134 and described in Table 4-121.

The System Event Status Indexed Clear Register allows clearing the status of an event. The event to clear is the index value written (0-63). This clears the Raw Status Register bit of the given index.

Figure 4-134. SICR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-121. SICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	W	0h	
9-0	STS_CLR_IDX	W	0h	Writes clear the status of the event given in the index value (0-63). Reads return 0.

4.5.3.7 EISR Register (offset = 28h) [reset = 0h]

EISR is shown in Figure 4-135 and described in Table 4-122.

The System Event Enable Indexed Set Register allows enabling an event. The event to enable is the index value written (0-63). This sets the Enable Register bit of the given index.

Figure 4-135. EISR Register

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
RESERVED																						EN_SET_IDX									
W-0h																						W-0h									

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-122. EISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	W	0h	
9-0	EN_SET_IDX	W	0h	Writes set the enable of the event given in the index value (0-63). Reads return 0.

4.5.3.8 EICR Register (offset = 2Ch) [reset = 0h]

EICR is shown in Figure 4-136 and described in Table 4-123.

The System Event Enable Indexed Clear Register allows disabling an event. The event to disable is the index value written (0-63). This clears the Enable Register bit of the given index.

Figure 4-136. EICR Register

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
RESERVED																						EN_CLR_IDX									
W-0h																						W-0h									

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-123. EICR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	W	0h	
9-0	EN_CLR_IDX	W	0h	Writes clear the enable of the event given in the index value (0-63). Reads return 0.

4.5.3.9 HIEISR Register (offset = 34h) [reset = 0h]

HIEISR is shown in Figure 4-137 and described in Table 4-124.

The Host Interrupt Enable Indexed Set Register allows enabling a host interrupt output. The host interrupt to enable is the index value written (0-9). This enables the host interrupt output or triggers the output again if already enabled.

Figure 4-137. HIEISR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-124. HIEISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	HINT_EN_SET_IDX	R/W	0h	Writes set the enable of the host interrupt given in the index value (0-9). Reads return 0.

4.5.3.10 HIDISR Register (offset = 38h) [reset = 0h]

HIDISR is shown in Figure 4-138 and described in Table 4-125.

The Host Interrupt Enable Indexed Clear Register allows disabling a host interrupt output. The host interrupt to disable is the index value written (0-9). This disables the host interrupt output.

Figure 4-138. HIDISR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-125. HIDISR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R/W	0h	
3-0	HINT_EN_CLR_IDX	R/W	0h	Writes clear the enable of the host interrupt given in the index value (0-9). Reads return 0.

4.5.3.11 GPIR Register (offset = 80h) [reset = 80000000h]

GPIR is shown in Figure 4-139 and described in Table 4-126.

The Global Prioritized Index Register shows the event number of the highest priority event pending across all the host interrupts.

Figure 4-139. GPIR Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-126. GPIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GLB_NONE	R	1h	No Interrupt is pending. Can be used by host to test for a negative value to see if no interrupts are pending.
30-10	RESERVED	R	0h	
9-0	GLB_PRI_INTR	R	0h	The currently highest priority event index (0-63) pending across all the host interrupts.

4.5.3.12 SRSR0 Register (offset = 200h) [reset = 0h]

SRSR0 is shown in Figure 4-140 and described in Table 4-127.

The System Event Status Raw/Set Register0 show the pending enabled status of the system events 0 to 31. Software can write to the Status Set Registers to set a system event without a hardware trigger. There is one bit per system event

Figure 4-140. SRSR0 Register

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
RAW_STS_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-127. SRSR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RAW_STS_31_0	R/W	0h	System event raw status and setting of the system events 0 to 31. Reads return the raw status. Write a 1 in a bit position to set the status of the system event. Writing a 0 has no effect.

4.5.3.13 SRSR1 Register (offset = 204h) [reset = 0h]

SRSR1 is shown in Figure 4-141 and described in Table 4-128.

The System Event Status Raw/Set Register1 show the pending enabled status of the system events 32 to 63. Software can write to the Status Set Registers to set a system event without a hardware trigger. There is one bit per system event

Figure 4-141. SRSR1 Register

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
RAW_STS_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-128. SRSR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RAW_STS_63_32	R/W	0h	System event raw status and setting of the system events 32 to 63. Reads return the raw status. Write a 1 in a bit position to set the status of the system event. Writing a 0 has no effect.

4.5.3.14 SECR0 Register (offset = 280h) [reset = 0h]

SECR0 is shown in Figure 4-142 and described in Table 4-129.

The System Event Status Enabled/Clear Register0 show the pending enabled status of the system events 0 to 31. Software can write to the Status Clear Registers to clear a system event after it has been serviced. If a system event status is not cleared then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. There is one bit per system event.

Figure 4-142. SECR0 Register

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
ENA_STS_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-129. SECR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENA_STS_31_0	R/W	0h	System event enabled status and clearing of the system events 0 to 31. Reads return the enabled status (before enabling with the Enable Registers). Write a 1 in a bit position to clear the status of the system event. Writing a 0 has no effect.

4.5.3.15 SECR1 Register (offset = 284h) [reset = 0h]

SECR1 is shown in Figure 4-143 and described in Table 4-130.

The System Event Status Enabled/Clear Register1 show the pending enabled status of the system events 32 to 63. Software can write to the Status Clear Registers to clear a system event after it has been serviced. If a system event status is not cleared then another host interrupt may not be triggered or another host interrupt may be triggered incorrectly. There is one bit per system event.

Figure 4-143. SECR1 Register

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
ENA_STS_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-130. SECR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ENA_STS_63_32	R/W	0h	System event enabled status and clearing of the system events 32 to 63. Reads return the enabled status (before enabling with the Enable Registers). Write a 1 in a bit position to clear the status of the system event. Writing a 0 has no effect.

4.5.3.16 ESR0 Register (offset = 300h) [reset = 0h]

ESR0 is shown in Figure 4-144 and described in Table 4-131.

The System Event Enable Set Register0 enables system events 0 to 31 to trigger outputs. System events that are not enabled do not interrupt the host. There is a bit per system event.

Figure 4-144. ESR0 Register

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
EN_SET_31_0																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-131. ESR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EN_SET_31_0		0h	System event enables system events 0 to 31. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to set that enable. Writing a 0 has no effect.

4.5.3.17 ESR1 Register (offset = 304h) [reset = 0h]

ESR1 is shown in Figure 4-145 and described in Table 4-132.

The System Event Enable Set Register1 enables system events 32 to 63 to trigger outputs. System events that are not enabled do not interrupt the host. There is a bit per system event.

Figure 4-145. ESR1 Register

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
EN_SET_63_32																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-132. ESR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EN_SET_63_32		0h	System event enables system events 32 to 63. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to set that enable. Writing a 0 has no effect.

4.5.3.18 ECR0 Register (offset = 380h) [reset = 0h]

ECR0 is shown in Figure 4-146 and described in Table 4-133.

The System Event Enable Clear Register0 disables system events 0 to 31 to map to channels. System events that are not enabled do not interrupt the host. There is a bit per system event.

Figure 4-146. ECR0 Register

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
EN_CLR_31_0																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-133. ECR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EN_CLR_31_0		0h	System event enables system events 0 to 31. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to clear that enable. Writing a 0 has no effect.

4.5.3.19 ECR1 Register (offset = 384h) [reset = 0h]

ECR1 is shown in Figure 4-147 and described in Table 4-134.

The System Event Enable Clear Register1 disables system events 32 to 63 to map to channels. System events that are not enabled do not interrupt the host. There is a bit per system event.

Figure 4-147. ECR1 Register

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
EN_CLR_63_32																															
0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-134. ECR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	EN_CLR_63_32		0h	System event enables system events 32 to 63. Read returns the enable value (0 = disabled, 1 = enabled). Write a 1 in a bit position to clear that enable. Writing a 0 has no effect.

4.5.3.20 CMR0 Register (offset = 400h) [reset = 0h]

CMR0 is shown in Figure 4-148 and described in Table 4-135.

The Channel Map Register0 specify the channel for the system events 0 to 3. There is one register per 4 system events. Note each CH_MAP_x bitfield corresponds to a system event. Channel numbers (0-9) should be written to these bitfields.

Figure 4-148. CMR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_3				RESERVED				CH_MAP_2			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_1				RESERVED				CH_MAP_0			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-135. CMR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_3	R/W	0h	Sets the channel for the system event 3
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_2	R/W	0h	Sets the channel for the system event 2
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_1	R/W	0h	Sets the channel for the system event 1
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_0	R/W	0h	Sets the channel for the system event 0

4.5.3.21 CMR1 Register (offset = 404h) [reset = 0h]

CMR1 is shown in Figure 4-149 and described in Table 4-136.

The Channel Map Register1 specify the channel for the system events 4 to 7. There is one register per 4 system events.

Figure 4-149. CMR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_7				RESERVED				CH_MAP_6			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_5				RESERVED				CH_MAP_4			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-136. CMR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_7	R/W	0h	Sets the channel for the system event 7
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_6	R/W	0h	Sets the channel for the system event 6
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_5	R/W	0h	Sets the channel for the system event 5
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_4	R/W	0h	Sets the channel for the system event 4

4.5.3.22 CMR2 Register (offset = 408h) [reset = 0h]

CMR2 is shown in Figure 4-150 and described in Table 4-137.

The Channel Map Register2 specify the channel for the system events 8 to 11. There is one register per 4 system events.

Figure 4-150. CMR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_11				RESERVED				CH_MAP_10			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_9				RESERVED				CH_MAP_8			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-137. CMR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_11	R/W	0h	Sets the channel for the system event 11
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_10	R/W	0h	Sets the channel for the system event 10
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_9	R/W	0h	Sets the channel for the system event 9
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_8	R/W	0h	Sets the channel for the system event 8

4.5.3.23 CMR3 Register (offset = 40Ch) [reset = 0h]

CMR3 is shown in Figure 4-151 and described in Table 4-138.

The Channel Map Register3 specify the channel for the system events 12 to 15. There is one register per 4 system events.

Figure 4-151. CMR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_15				RESERVED				CH_MAP_14			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_13				RESERVED				CH_MAP_12			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-138. CMR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_15	R/W	0h	Sets the channel for the system event 15
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_14	R/W	0h	Sets the channel for the system event 14
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_13	R/W	0h	Sets the channel for the system event 13
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_12	R/W	0h	Sets the channel for the system event 12

4.5.3.24 CMR4 Register (offset = 410h) [reset = 0h]

CMR4 is shown in Figure 4-152 and described in Table 4-139.

The Channel Map Register4 specify the channel for the system events 16 to 19. There is one register per 4 system events.

Figure 4-152. CMR4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_19				RESERVED				CH_MAP_18			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_17				RESERVED				CH_MAP_16			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-139. CMR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_19	R/W	0h	Sets the channel for the system event 19
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_18	R/W	0h	Sets the channel for the system event 18
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_17	R/W	0h	Sets the channel for the system event 17
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_16	R/W	0h	Sets the channel for the system event 16

4.5.3.25 CMR5 Register (offset = 414h) [reset = 0h]

CMR5 is shown in Figure 4-153 and described in Table 4-140.

The Channel Map Register5 specify the channel for the system events 20 to 23. There is one register per 4 system events.

Figure 4-153. CMR5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_23				RESERVED				CH_MAP_22			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_21				RESERVED				CH_MAP_20			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-140. CMR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_23	R/W	0h	Sets the channel for the system event 23
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_22	R/W	0h	Sets the channel for the system event 22
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_21	R/W	0h	Sets the channel for the system event 21
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_20	R/W	0h	Sets the channel for the system event 20

4.5.3.26 CMR6 Register (offset = 418h) [reset = 0h]

CMR6 is shown in Figure 4-154 and described in Table 4-141.

The Channel Map Register6 specify the channel for the system events 24 to 27. There is one register per 4 system events.

Figure 4-154. CMR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_27				RESERVED				CH_MAP_26			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_25				RESERVED				CH_MAP_24			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-141. CMR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_27	R/W	0h	Sets the channel for the system event 27
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_26	R/W	0h	Sets the channel for the system event 26
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_25	R/W	0h	Sets the channel for the system event 25
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_24	R/W	0h	Sets the channel for the system event 24

4.5.3.27 CMR7 Register (offset = 41Ch) [reset = 0h]

CMR7 is shown in Figure 4-155 and described in Table 4-142.

The Channel Map Register7 specify the channel for the system events 28 to 31. There is one register per 4 system events.

Figure 4-155. CMR7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_31				RESERVED				CH_MAP_30			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_29				RESERVED				CH_MAP_28			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-142. CMR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_31	R/W	0h	Sets the channel for the system event 31
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_30	R/W	0h	Sets the channel for the system event 30
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_29	R/W	0h	Sets the channel for the system event 29
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_28	R/W	0h	Sets the channel for the system event 28

4.5.3.28 CMR8 Register (offset = 420h) [reset = 0h]

CMR8 is shown in Figure 4-156 and described in Table 4-143.

The Channel Map Register8 specify the channel for the system events 32 to 35. There is one register per 4 system events.

Figure 4-156. CMR8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_35				RESERVED				CH_MAP_34			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_33				RESERVED				CH_MAP_32			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-143. CMR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_35	R/W	0h	Sets the channel for the system event 35
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_34	R/W	0h	Sets the channel for the system event 34
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_33	R/W	0h	Sets the channel for the system event 33
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_32	R/W	0h	Sets the channel for the system event 32

4.5.3.29 CMR9 Register (offset = 424h) [reset = 0h]

CMR9 is shown in Figure 4-157 and described in Table 4-144.

The Channel Map Register9 specify the channel for the system events 36 to 39. There is one register per 4 system events.

Figure 4-157. CMR9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_39				RESERVED				CH_MAP_38			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_37				RESERVED				CH_MAP_36			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-144. CMR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_39	R/W	0h	Sets the channel for the system event 39
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_38	R/W	0h	Sets the channel for the system event 38
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_37	R/W	0h	Sets the channel for the system event 37
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_36	R/W	0h	Sets the channel for the system event 36

4.5.3.30 CMR10 Register (offset = 428h) [reset = 0h]

CMR10 is shown in Figure 4-158 and described in Table 4-145.

The Channel Map Register10 specify the channel for the system events 40 to 43. There is one register per 4 system events.

Figure 4-158. CMR10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_43				RESERVED				CH_MAP_42			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_41				RESERVED				CH_MAP_40			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-145. CMR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_43	R/W	0h	Sets the channel for the system event 43
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_42	R/W	0h	Sets the channel for the system event 42
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_41	R/W	0h	Sets the channel for the system event 41
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_40	R/W	0h	Sets the channel for the system event 40

4.5.3.31 CMR11 Register (offset = 42Ch) [reset = 0h]

CMR11 is shown in Figure 4-159 and described in Table 4-146.

The Channel Map Register11 specify the channel for the system events 44 to 47. There is one register per 4 system events.

Figure 4-159. CMR11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_47				RESERVED				CH_MAP_46			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_45				RESERVED				CH_MAP_44			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-146. CMR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_47	R/W	0h	Sets the channel for the system event 47
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_46	R/W	0h	Sets the channel for the system event 46
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_45	R/W	0h	Sets the channel for the system event 45
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_44	R/W	0h	Sets the channel for the system event 44

4.5.3.32 CMR12 Register (offset = 430h) [reset = 0h]

CMR12 is shown in Figure 4-160 and described in Table 4-147.

The Channel Map Register12 specify the channel for the system events 48 to 51. There is one register per 4 system events.

Figure 4-160. CMR12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_51				RESERVED				CH_MAP_50			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_49				RESERVED				CH_MAP_48			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-147. CMR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_51	R/W	0h	Sets the channel for the system event 51
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_50	R/W	0h	Sets the channel for the system event 50
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_49	R/W	0h	Sets the channel for the system event 49
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_48	R/W	0h	Sets the channel for the system event 48

4.5.3.33 CMR13 Register (offset = 434h) [reset = 0h]

CMR13 is shown in Figure 4-161 and described in Table 4-148.

The Channel Map Register13 specify the channel for the system events 52 to 55. There is one register per 4 system events.

Figure 4-161. CMR13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_55				RESERVED				CH_MAP_54			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_53				RESERVED				CH_MAP_52			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-148. CMR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_55	R/W	0h	Sets the channel for the system event 55
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_54	R/W	0h	Sets the channel for the system event 54
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_53	R/W	0h	Sets the channel for the system event 53
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_52	R/W	0h	Sets the channel for the system event 52

4.5.3.34 CMR14 Register (offset = 438h) [reset = 0h]

CMR14 is shown in Figure 4-162 and described in Table 4-149.

The Channel Map Register14 specify the channel for the system events 56 to 59. There is one register per 4 system events.

Figure 4-162. CMR14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_59				RESERVED				CH_MAP_58			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_57				RESERVED				CH_MAP_56			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-149. CMR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_59	R/W	0h	Sets the channel for the system event 59
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_58	R/W	0h	Sets the channel for the system event 58
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_57	R/W	0h	Sets the channel for the system event 57
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_56	R/W	0h	Sets the channel for the system event 56

4.5.3.35 CMR15 Register (offset = 43Ch) [reset = 0h]

CMR15 is shown in Figure 4-163 and described in Table 4-150.

The Channel Map Register15 specify the channel for the system events 60 to 63. There is one register per 4 system events.

Figure 4-163. CMR15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				CH_MAP_63				RESERVED				CH_MAP_62			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				CH_MAP_61				RESERVED				CH_MAP_60			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-150. CMR15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	CH_MAP_63	R/W	0h	Sets the channel for the system event 63
23-20	RESERVED	R/W	0h	
19-16	CH_MAP_62	R/W	0h	Sets the channel for the system event 62
15-12	RESERVED	R/W	0h	
11-8	CH_MAP_61	R/W	0h	Sets the channel for the system event 61
7-4	RESERVED	R/W	0h	
3-0	CH_MAP_60	R/W	0h	Sets the channel for the system event 60

4.5.3.36 HMR0 Register (offset = 800h) [reset = 0h]

HMR0 is shown in Figure 4-164 and described in Table 4-151.

The Host Interrupt Map Register0 define the host interrupt for channels 0 to 3. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Figure 4-164. HMR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HINT_MAP_3				RESERVED				HINT_MAP_2			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_1				RESERVED				HINT_MAP_0			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-151. HMR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	HINT_MAP_3	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 3
23-20	RESERVED	R/W	0h	
19-16	HINT_MAP_2	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 2
15-12	RESERVED	R/W	0h	
11-8	HINT_MAP_1	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 1
7-4	RESERVED	R/W	0h	
3-0	HINT_MAP_0	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 0

4.5.3.37 HMR1 Register (offset = 804h) [reset = 0h]

HMR1 is shown in Figure 4-165 and described in Table 4-152.

The Host Interrupt Map Register1 define the host interrupt for channels 4 to 7. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Figure 4-165. HMR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				HINT_MAP_7				RESERVED				HINT_MAP_6			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_5				RESERVED				HINT_MAP_4			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-152. HMR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	
27-24	HINT_MAP_7	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 7
23-20	RESERVED	R/W	0h	
19-16	HINT_MAP_6	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 6
15-12	RESERVED	R/W	0h	
11-8	HINT_MAP_5	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 5
7-4	RESERVED	R/W	0h	
3-0	HINT_MAP_4	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 4

4.5.3.38 HMR2 Register (offset = 808h) [reset = 0h]

HMR2 is shown in Figure 4-166 and described in Table 4-153.

The Host Interrupt Map Register2 define the host interrupt for channels 8 to 9. There is one register per 4 channels. Channels with forced host interrupt mappings will have their fields read-only.

Figure 4-166. HMR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				HINT_MAP_9				RESERVED				HINT_MAP_8			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-153. HMR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R/W	0h	
11-8	HINT_MAP_9	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 9
7-4	RESERVED	R/W	0h	
3-0	HINT_MAP_8	R/W	0h	HOST INTERRUPT MAP FOR CHANNEL 8

4.5.3.39 HIPIR0 Register (offset = 900h) [reset = 80000000h]

HIPIR0 is shown in Figure 4-167 and described in Table 4-154.

The Host Interrupt Prioritized Index Register0 shows the highest priority current pending interrupt for the host interrupt 0. There is one register per host interrupt.

Figure 4-167. HIPIR0 Register

31	30	29	28	27	26	25	24
NONE_HINT_0	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_0	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_0							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-154. HIPIR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_0	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_0	R	0h	HOST INT 0 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.40 HIPIR1 Register (offset = 904h) [reset = 80000000h]

HIPIR1 is shown in Figure 4-168 and described in Table 4-155.

The Host Interrupt Prioritized Index Register1 shows the highest priority current pending interrupt for the host interrupt 1. There is one register per host interrupt.

Figure 4-168. HIPIR1 Register

31	30	29	28	27	26	25	24
NONE_HINT_1	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_1	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_1							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-155. HIPIR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_1	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_1	R	0h	HOST INT 1 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.41 HIPIR2 Register (offset = 908h) [reset = 80000000h]

HIPIR2 is shown in Figure 4-169 and described in Table 4-156.

The Host Interrupt Prioritized Index Register2 shows the highest priority current pending interrupt for the host interrupt 2. There is one register per host interrupt.

Figure 4-169. HIPIR2 Register

31	30	29	28	27	26	25	24
NONE_HINT_2	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_2	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_2							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-156. HIPIR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_2	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_2	R	0h	HOST INT 2 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.42 HIPIR3 Register (offset = 90Ch) [reset = 80000000h]

HIPIR3 is shown in Figure 4-170 and described in Table 4-157.

The Host Interrupt Prioritized Index Register3 shows the highest priority current pending interrupt for the host interrupt 3. There is one register per host interrupt.

Figure 4-170. HIPIR3 Register

31	30	29	28	27	26	25	24
NONE_HINT_3	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_3	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_3							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-157. HIPIR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_3	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_3	R	0h	HOST INT 3 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.43 HIPIR4 Register (offset = 910h) [reset = 80000000h]

HIPIR4 is shown in Figure 4-171 and described in Table 4-158.

The Host Interrupt Prioritized Index Register4 shows the highest priority current pending interrupt for the host interrupt 4. There is one register per host interrupt.

Figure 4-171. HIPIR4 Register

31	30	29	28	27	26	25	24
NONE_HINT_4	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_4	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_4							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-158. HIPIR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_4	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_4	R	0h	HOST INT 4 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.44 HIPIR5 Register (offset = 914h) [reset = 80000000h]

HIPIR5 is shown in Figure 4-172 and described in Table 4-159.

The Host Interrupt Prioritized Index Register5 shows the highest priority current pending interrupt for the host interrupt 5. There is one register per host interrupt.

Figure 4-172. HIPIR5 Register

31	30	29	28	27	26	25	24
NONE_HINT_5	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_5	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_5							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-159. HIPIR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_5	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_5	R	0h	HOST INT 5 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.45 HIPIR6 Register (offset = 918h) [reset = 80000000h]

HIPIR6 is shown in Figure 4-173 and described in Table 4-160.

The Host Interrupt Prioritized Index Register6 shows the highest priority current pending interrupt for the host interrupt 6. There is one register per host interrupt.

Figure 4-173. HIPIR6 Register

31	30	29	28	27	26	25	24
NONE_HINT_6	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_6	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_6							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-160. HIPIR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_6	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_6	R	0h	HOST INT 6 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.46 HIPIR7 Register (offset = 91Ch) [reset = 80000000h]

HIPIR7 is shown in Figure 4-174 and described in Table 4-161.

The Host Interrupt Prioritized Index Register7 shows the highest priority current pending interrupt for the host interrupt 7. There is one register per host interrupt.

Figure 4-174. HIPIR7 Register

31	30	29	28	27	26	25	24
NONE_HINT_7	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_7	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_7							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-161. HIPIR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_7	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_7	R	0h	HOST INT 7 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.47 HIPIR8 Register (offset = 920h) [reset = 80000000h]

HIPIR8 is shown in Figure 4-175 and described in Table 4-162.

The Host Interrupt Prioritized Index Register8 shows the highest priority current pending interrupt for the host interrupt 8. There is one register per host interrupt.

Figure 4-175. HIPIR8 Register

31	30	29	28	27	26	25	24
NONE_HINT_8	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_8	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_8							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-162. HIPIR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_8	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_8	R	0h	HOST INT 8 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.48 HIPIR9 Register (offset = 924h) [reset = 80000000h]

HIPIR9 is shown in Figure 4-176 and described in Table 4-163.

The Host Interrupt Prioritized Index Register9 shows the highest priority current pending interrupt for the host interrupt 9. There is one register per host interrupt.

Figure 4-176. HIPIR9 Register

31	30	29	28	27	26	25	24
NONE_HINT_9	RESERVED						
R-1h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						PRI_HINT_9	
R-0h						R-0h	
7	6	5	4	3	2	1	0
PRI_HINT_9							
R-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-163. HIPIR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	NONE_HINT_9	R	1h	No pending interrupt.
30-10	RESERVED	R	0h	
9-0	PRI_HINT_9	R	0h	HOST INT 9 PRIORITIZED INTERRUPT. Interrupt number of the highest priority pending interrupt for this host interrupt.

4.5.3.49 SIPR0 Register (offset = D00h) [reset = 1h]

SIPR0 is shown in Figure 4-177 and described in Table 4-164.

The System Event Polarity Register0 define the polarity of the system events 0 to 31. There is a polarity for each system event. The polarity of all system events is active high; always write 1 to the bits of this register.

Figure 4-177. SIPR0 Register

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
POLARITY_31_0																															
R/W-1h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-164. SIPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	POLARITY_31_0	R/W	1h	Interrupt polarity of the system events 0 to 31. 0 = active low. 1 = active high.

4.5.3.50 SIPR1 Register (offset = D04h) [reset = 1h]

SIPR1 is shown in Figure 4-178 and described in Table 4-165.

The System Event Polarity Register1 define the polarity of the system events 32 to 63. There is a polarity for each system event. The polarity of all system events is active high; always write 1 to the bits of this register.

Figure 4-178. SIPR1 Register

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
POLARITY_63_32																															
R/W-1h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-165. SIPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	POLARITY_63_32	R/W	1h	Interrupt polarity of the system events 32 to 63. 0 = active low. 1 = active high.

4.5.3.51 SITR0 Register (offset = D80h) [reset = 0h]

SITR0 is shown in Figure 4-179 and described in Table 4-166.

The System Event Type Register0 define the type of the system events 0 to 31. There is a type for each system event. The type of all system events is pulse; always write 0 to the bits of this register.

Figure 4-179. SITR0 Register

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
TYPE_31_0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-166. SITR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TYPE_31_0	R/W	0h	Interrupt type of the system events 0 to 31. 0 = level or pulse interrupt. 1 = edge interrupt (required edge detect).

4.5.3.52 SITR1 Register (offset = D84h) [reset = 0h]

SITR1 is shown in Figure 4-180 and described in Table 4-167.

The System Event Type Register1 define the type of the system events 32 to 63. There is a type for each system event. The type of all system events is pulse; always write 0 to the bits of this register.

Figure 4-180. SITR1 Register

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
TYPE_63_32																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-167. SITR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TYPE_63_32	R/W	0h	Interrupt type of the system events 32 to 63. 0 = level or pulse interrupt. 1 = edge interrupt (required edge detect).

4.5.3.53 HINLR0 Register (offset = 1100h) [reset = 100h]

HINLR0 is shown in Figure 4-181 and described in Table 4-168.

The Host Interrupt Nesting Level Register0 display and control the nesting level for host interrupt 0. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-181. HINLR0 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_0
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_0							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-168. HINLR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_0	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.54 HINLR1 Register (offset = 1104h) [reset = 100h]

HINLR1 is shown in Figure 4-182 and described in Table 4-169.

The Host Interrupt Nesting Level Register1 display and control the nesting level for host interrupt 1. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-182. HINLR1 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_1
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_1							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-169. HINLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_1	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.55 HINLR2 Register (offset = 1108h) [reset = 100h]

HINLR2 is shown in Figure 4-183 and described in Table 4-170.

The Host Interrupt Nesting Level Register2 display and control the nesting level for host interrupt 2. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-183. HINLR2 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_2
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_2							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-170. HINLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_2	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.56 HINLR3 Register (offset = 110Ch) [reset = 100h]

HINLR3 is shown in Figure 4-184 and described in Table 4-171.

The Host Interrupt Nesting Level Register3 display and control the nesting level for host interrupt 3. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-184. HINLR3 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_3
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_3							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-171. HINLR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_3	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.57 HINLR4 Register (offset = 1110h) [reset = 100h]

HINLR4 is shown in Figure 4-185 and described in Table 4-172.

The Host Interrupt Nesting Level Register4 display and control the nesting level for host interrupt 4. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-185. HINLR4 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_4
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_4							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-172. HINLR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_4	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.58 HINLR5 Register (offset = 1114h) [reset = 100h]

HINLR5 is shown in Figure 4-186 and described in Table 4-173.

The Host Interrupt Nesting Level Register5 display and control the nesting level for host interrupt 5. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-186. HINLR5 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_5
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_5							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-173. HINLR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_5	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.59 HINLR6 Register (offset = 1118h) [reset = 100h]

HINLR6 is shown in Figure 4-187 and described in Table 4-174.

The Host Interrupt Nesting Level Register6 display and control the nesting level for host interrupt 6. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-187. HINLR6 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_6
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_6							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-174. HINLR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_6	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.60 HINLR7 Register (offset = 111Ch) [reset = 100h]

HINLR7 is shown in Figure 4-188 and described in Table 4-175.

The Host Interrupt Nesting Level Register7 display and control the nesting level for host interrupt 7. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-188. HINLR7 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_7
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_7							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-175. HINLR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_7	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.61 HINLR8 Register (offset = 1120h) [reset = 100h]

HINLR8 is shown in Figure 4-189 and described in Table 4-176.

The Host Interrupt Nesting Level Register8 display and control the nesting level for host interrupt 8. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-189. HINLR8 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_8
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_8							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-176. HINLR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_8	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.62 HINLR9 Register (offset = 1124h) [reset = 100h]

HINLR9 is shown in Figure 4-190 and described in Table 4-177.

The Host Interrupt Nesting Level Register9 display and control the nesting level for host interrupt 9. The nesting level controls which channel and lower priority channels are nested. There is one register per host interrupt.

Figure 4-190. HINLR9 Register

31	30	29	28	27	26	25	24
AUTO_OVERRIDE	RESERVED						
W-0h	R/W-0h						
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							NEST_HINT_9
R/W-0h							R/W-100h
7	6	5	4	3	2	1	0
NEST_HINT_9							
R/W-100h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-177. HINLR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AUTO_OVERRIDE	W	0h	Reads return 0. Writes of a 1 override the auto updating of the nesting_level and use the write data.
30-9	RESERVED	R/W	0h	
8-0	NEST_HINT_9	R/W	100h	Reads return the current nesting level for the host interrupt. Writes set the nesting level for the host interrupt. In auto mode the value is updated internally unless the auto_override is set and then the write data is used.

4.5.3.63 HIER Register (offset = 1500h) [reset = 0h]

HIER is shown in Figure 4-191 and described in Table 4-178.

The Host Interrupt Enable Registers enable or disable individual host interrupts. These work separately from the global enables. There is one bit per host interrupt. These bits are updated when writing to the Host Interrupt Enable Index Set and Host Interrupt Enable Index Clear registers.

Figure 4-191. HIER Register

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
RESERVED																						EN_HINT									
R/W-0h																						R/W-0h									

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-178. HIER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R/W	0h	
9-0	EN_HINT	R/W	0h	The enable of the host interrupts (one per bit). 0 = disabled 1 = enabled

4.5.4 PRU_ICSS_IEP Registers

Table 4-179 lists the memory-mapped registers for the PRU_ICSS_IEP. All register offset addresses not listed in Table 4-179 should be considered as reserved locations and the register contents should not be modified.

Table 4-179. PRU_ICSS_IEP Registers

Offset	Acronym	Register Name	Section
0h	IEP_TMR_GLB_CFG		Section 4.5.4.1
4h	IEP_TMR_GLB_STS		Section 4.5.4.2
8h	IEP_TMR_COMPEN		Section 4.5.4.3
Ch	IEP_TMR_CNT		Section 4.5.4.4
10h	IEP_TMR_CAP_CFG		Section 4.5.4.5
14h	IEP_TMR_CAP_STS		Section 4.5.4.6
18h	IEP_TMR_CAPR0		Section 4.5.4.7
1Ch	IEP_TMR_CAPR1		Section 4.5.4.8
20h	IEP_TMR_CAPR2		Section 4.5.4.9
24h	IEP_TMR_CAPR3		Section 4.5.4.10
28h	IEP_TMR_CAPR4		Section 4.5.4.11
2Ch	IEP_TMR_CAPR5		Section 4.5.4.12
30h	IEP_TMR_CAPR6		Section 4.5.4.13
34h	IEP_TMR_CAPF6		Section 4.5.4.14
38h	IEP_TMR_CAPR7		Section 4.5.4.15
3Ch	IEP_TMR_CAPF7		Section 4.5.4.16
40h	IEP_TMR_CMP_CFG		Section 4.5.4.17
44h	IEP_TMR_CMP_STS		Section 4.5.4.18
48h	IEP_TMR_CMP0		Section 4.5.4.19
4Ch	IEP_TMR_CMP1		Section 4.5.4.20
50h	IEP_TMR_CMP2		Section 4.5.4.21
54h	IEP_TMR_CMP3		Section 4.5.4.22
58h	IEP_TMR_CMP4		Section 4.5.4.23
5Ch	IEP_TMR_CMP5		Section 4.5.4.24

Table 4-179. PRU_ICSS_IEP Registers (continued)

Offset	Acronym	Register Name	Section
60h	IEP_TMR_CMP6		Section 4.5.4.25
64h	IEP_TMR_CMP7		Section 4.5.4.26
80h	IEP_TMR_RXIPG0		Section 4.5.4.27
84h	IEP_TMR_RXIPG1		Section 4.5.4.28
100h	IEP_SYNC_CTRL	SYNC GENERATION CONTROL	Section 4.5.4.29
104h	IEP_SYNC_FIRST_STAT	SYNC GENERATION FIRST EVENT STATUS	Section 4.5.4.30
108h	IEP_SYNC0_STAT	SYNC0 STATUS	Section 4.5.4.31
10Ch	IEP_SYNC1_STAT	SYNC1 STATUS	Section 4.5.4.32
110h	IEP_SYNC_PWIDTH	SYNC PULSE WIDTH CONFIGURE	Section 4.5.4.33
114h	IEP_SYNC0_PERIOD	SYNC PERIOD CONFIGURE	Section 4.5.4.34
118h	IEP_SYNC1_DELAY	SYNC CTRL	Section 4.5.4.35
11Ch	IEP_SYNC_START	SYNC START CONFIGURE	Section 4.5.4.36
200h	IEP_WD_PREDIV		Section 4.5.4.37
204h	IEP_PDI_WD_TIM		Section 4.5.4.38
208h	IEP_PD_WD_TIM		Section 4.5.4.39
20Ch	IEP_WD_STATUS		Section 4.5.4.40
210h	IEP_WD_EXP_CNT		Section 4.5.4.41
214h	IEP_WD_CTRL		Section 4.5.4.42
300h	IEP_DIGIO_CTRL		Section 4.5.4.43
304h	IEP_DIGIO_STATUS		Section 4.5.4.44
308h	IEP_DIGIO_DATA_IN		Section 4.5.4.45
30Ch	IEP_DIGIO_DATA_IN_RAW		Section 4.5.4.46
310h	IEP_DIGIO_DATA_OUT		Section 4.5.4.47
314h	IEP_DIGIO_DATA_OUT_EN		Section 4.5.4.48
318h	IEP_DIGIO_EXP		Section 4.5.4.49

4.5.4.1 IEP_TMR_GLB_CFG Register (offset = 0h) [reset = 550h]

IEP_TMR_GLB_CFG is shown in Figure 4-192 and described in Table 4-180.

GLOBAL CONFIGURE

Figure 4-192. IEP_TMR_GLB_CFG Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-180. IEP_TMR_GLB_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	
19-8	CMP_INC	R/W	5h	Defines the increment value when compensation is active
7-4	DEFAULT_INC	R/W	5h	Defines the default increment value
3-1	RESERVED	R	0h	
0	CNT_ENABLE	R/W	0h	Counter enable 0: Disables the counter. The counter maintains the current count. 1: Enables the counter.

4.5.4.2 IEP_TMR_GLB_STS Register (offset = 4h) [reset = 0h]

IEP_TMR_GLB_STS is shown in Figure 4-193 and described in Table 4-181.

GLOBAL STATUS

Figure 4-193. IEP_TMR_GLB_STS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CNT_OVF
R-0h							R/W1toClr-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-181. IEP_TMR_GLB_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	CNT_OVF	R/W1toClr	0h	Counter overflow status 0: No overflow 1: Overflow occurred

4.5.4.3 IEP_TMR_COMPEN Register (offset = 8h) [reset = 0h]

IEP_TMR_COMPEN is shown in Figure 4-194 and described in Table 4-182.

COMPENSATION

Figure 4-194. IEP_TMR_COMPEN Register

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
RESERVED								COMPEN_CNT																							
R-0h								R/W-0h																							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-182. IEP_TMR_COMPEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	
23-0	COMPEN_CNT	R/W	0h	Compensation counter. Read returns the current compen_cnt value. 0: Compensation is disabled and counter will increment by DEFAULT_INC. n: Compensation is enabled until COMPEN_CNT decrements to 0. The COMPEN_CNT value decrements on every iep_clk/ocp_clk cycle. When COMPEN_CNT is greater than 0, then count value increments by CMP_INC.

4.5.4.4 IEP_TMR_CNT Register (offset = Ch) [reset = 0h]

IEP_TMR_CNT is shown in Figure 4-195 and described in Table 4-183.

COUNT is a free running counter with a sticky over flag status bit. The counter overflow flag is set when the counter switches or rolls over from 0xffff_ffff -> 0x0000_0000 and continues to count up. The software must read and clear the counter overflow flag, and increment the MSB in software variable.

Figure 4-195. IEP_TMR_CNT Register

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
COUNT																															
R/W1toClr-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-183. IEP_TMR_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W1toClr	0h	32-bit count value. Increments by (DEFAULT_INC or CMP_INC) on every positive edge of iep_clk (200MHz) or ocp_clk.

4.5.4.5 IEP_TMR_CAP_CFG Register (offset = 10h) [reset = 1FC00h]

IEP_TMR_CAP_CFG is shown in Figure 4-196 and described in Table 4-184.

CAPTURE CONFIGURE

Figure 4-196. IEP_TMR_CAP_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED						CAP_ASYNC_EN	
R/W-0h						R/W-7Fh	
15	14	13	12	11	10	9	8
CAP_ASYNC_EN						CAP7F_1ST_EVENT_EN	CAP7R_1ST_EVENT_EN
R/W-7Fh						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CAP6F_1ST_EVENT_EN	CAP6R_1ST_EVENT_EN	CAP_1ST_EVENT_EN					
R/W-0h	R/W-0h	R/W-0h					

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; -*n* = value after reset

Table 4-184. IEP_TMR_CAP_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R/W	0h	
17-10	CAP_ASYNC_EN	R/W	7Fh	Synchronization of the capture inputs to the iep_clk/ocp_clk enable. Note if input capture signal is asynchronous to iep_clk, enabling synchronization will cause the capture contents to be invalid. CAP_ASYNC_EN[n] maps to CAPR[n]. 0: Disable synchronization 1: Enable synchronization
9	CAP7F_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAPF7 0: Continues mode. The capture status is not set when events occur. 1: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
8	CAP7R_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAPR7 0: Continues mode. The capture status is not set when events occur. 1: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
7	CAP6F_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAPF6 0: Continues mode. The capture status is not set when events occur. 1: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.

Table 4-184. IEP_TMR_CAP_CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	CAP6R_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for CAPR6 0: Continues mode. The capture status is not set when events occur. 1: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.
5-0	CAP_1ST_EVENT_EN	R/W	0h	Capture 1st Event Enable for registers. CAP_1ST_EVENT_EN[n] maps to CAPR[n]. 0: Continues mode. The capture status is not set when events occur. 1: First Event mode. The capture status is set when the first event occurs and must be cleared before new data will fill buffer. Time value is captured when first event occurs and held until time is read.

4.5.4.6 IEP_TMR_CAP_STS Register (offset = 14h) [reset = 0h]

Register mask: 0h

IEP_TMR_CAP_STS is shown in Figure 4-197 and described in Table 4-185.

CAPTURE STATUS CONFIGURE. Note: Capture will always occur as long as it is enabled, if enabled the user cannot tell if 2 events occurred.

Figure 4-197. IEP_TMR_CAP_STS Register

31	30	29	28	27	26	25	24
RESERVED							
R-X							
23	22	21	20	19	18	17	16
CAP_RAW							
R-X							
15	14	13	12	11	10	9	8
RESERVED					CAP_VALID	CAPF7_VALID	CAPR7_VALID
R-X					R-0h	RtoClr-0h	RtoClr-0h
7	6	5	4	3	2	1	0
CAPF6_VALID	CAPR6_VALID	CAPR_VALID					
RtoClr-0h	RtoClr-0h	RtoClr-0h					

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-185. IEP_TMR_CAP_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	X	
23-16	CAP_RAW	R	X	Raw/Current status bit for each of the capture registers, where CAP_RAW[n] maps to CAPR[n]. 0: Current state is low 1: Current state is high
15-11	RESERVED	R	X	
10	CAP_VALID	R	0h	Valid status for capture function. Reflects the ORed result from CAP_STATUS [9:0]. 0: No Hit for any capture event, i.e., there are all 0 in CAP_STATUS [9:0]. 1: Hit for 1 or more captures events is pending, i.e., there has at least one value equal to 1 in CAP_STATUS [9:0].
9	CAPF7_VALID	RtoClr	0h	Valid status for CAPF7 (fall). 0: No Hit, no capture event occurred 1: Hit, capture event occurred
8	CAPR7_VALID	RtoClr	0h	Valid status for CAPR7 (rise). 0: No Hit, no capture event occurred 1: Hit, capture event occurred
7	CAPF6_VALID	RtoClr	0h	Valid status for CAPF6 (fall). 0: No Hit, no capture event occurred 1: Hit, capture event occurred
6	CAPR6_VALID	RtoClr	0h	Valid status for CAPR6 (rise). 0: No Hit, no capture event occurred 1: Hit, capture event occurred
5-0	CAPR_VALID	RtoClr	0h	Valid status bit for each compare register, where CAPR_VALID[n] maps to CAPR[n] (rise). 0: No Hit, no capture event occurred 1: Hit, capture event occurred

4.5.4.7 IEP_TMR_CAPR0 Register (offset = 18h) [reset = 0h]

IEP_TMR_CAPR0 is shown in Figure 4-198 and described in Table 4-186.

CAPTURE RISE0

Figure 4-198. IEP_TMR_CAPR0 Register

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
CAPR0																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-186. IEP_TMR_CAPR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR0	R	0h	Value captured for CAPR0 event

4.5.4.8 IEP_TMR_CAPR1 Register (offset = 1Ch) [reset = 0h]

IEP_TMR_CAPR1 is shown in Figure 4-199 and described in Table 4-187.

CAPTURE RISE1

Figure 4-199. IEP_TMR_CAPR1 Register

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
CAPR1																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-187. IEP_TMR_CAPR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR1	R	0h	Value captured for CAPR1 event

4.5.4.9 IEP_TMR_CAPR2 Register (offset = 20h) [reset = 0h]

IEP_TMR_CAPR2 is shown in Figure 4-200 and described in Table 4-188.

CAPTURE RISE2

Figure 4-200. IEP_TMR_CAPR2 Register

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
CAPR2																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-188. IEP_TMR_CAPR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR2	R	0h	Value captured for CAPR2 event

4.5.4.10 IEP_TMR_CAPR3 Register (offset = 24h) [reset = 0h]

IEP_TMR_CAPR3 is shown in Figure 4-201 and described in Table 4-189.

CAPTURE RISE3

Figure 4-201. IEP_TMR_CAPR3 Register

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
CAPR3																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-189. IEP_TMR_CAPR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR3	R	0h	Value captured for CAPR3 event

4.5.4.11 IEP_TMR_CAPR4 Register (offset = 28h) [reset = 0h]

IEP_TMR_CAPR4 is shown in Figure 4-202 and described in Table 4-190.

CAPTURE RISE4

Figure 4-202. IEP_TMR_CAPR4 Register

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
CAPR4																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-190. IEP_TMR_CAPR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR4	R	0h	Value captured for CAPR4 event

4.5.4.12 IEP_TMR_CAPR5 Register (offset = 2Ch) [reset = 0h]

IEP_TMR_CAPR5 is shown in Figure 4-203 and described in Table 4-191.

CAPTURE RISE5

Figure 4-203. IEP_TMR_CAPR5 Register

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
CAPR5																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-191. IEP_TMR_CAPR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR5	R	0h	Value captured for CAPR5 event

4.5.4.13 IEP_TMR_CAPR6 Register (offset = 30h) [reset = 0h]

IEP_TMR_CAPR6 is shown in Figure 4-204 and described in Table 4-192.

CAPTURE RISE6

Figure 4-204. IEP_TMR_CAPR6 Register

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
CAPR6																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-192. IEP_TMR_CAPR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR6	R	0h	Value captured for CAPR6 event

4.5.4.14 IEP_TMR_CAPF6 Register (offset = 34h) [reset = 0h]

IEP_TMR_CAPF6 is shown in Figure 4-205 and described in Table 4-193.

CAPTURE FALL6

Figure 4-205. IEP_TMR_CAPF6 Register

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
CAPF6																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-193. IEP_TMR_CAPF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF6	R	0h	Value captured for CAPF6 event

4.5.4.15 IEP_TMR_CAPR7 Register (offset = 38h) [reset = 0h]

IEP_TMR_CAPR7 is shown in Figure 4-206 and described in Table 4-194.

CAPTURE RISE7

Figure 4-206. IEP_TMR_CAPR7 Register

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
CAPR7																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-194. IEP_TMR_CAPR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPR7	R	0h	Value captured for CAPR7 event

4.5.4.16 IEP_TMR_CAPF7 Register (offset = 3Ch) [reset = 0h]

IEP_TMR_CAPF7 is shown in Figure 4-207 and described in Table 4-195.

CAPTURE FALL7

Figure 4-207. IEP_TMR_CAPF7 Register

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
CAPF7																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-195. IEP_TMR_CAPF7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAPF7	R	0h	Value captured for CAPF7 event

4.5.4.17 IEP_TMR_CMP_CFG Register (offset = 40h) [reset = 0h]

IEP_TMR_CMP_CFG is shown in Figure 4-208 and described in Table 4-196.

COMPARE CONFIGURE

Figure 4-208. IEP_TMR_CMP_CFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							CMP_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
CMP_EN							CMP0_RST_CNT_EN
R/W-0h							R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-196. IEP_TMR_CMP_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8-1	CMP_EN	R/W	0h	Compare registers enable, where CMP_EN[0] maps to CMP[0]. 0: Disables event 1: Enables event
0	CMP0_RST_CNT_EN	R/W	0h	Counter reset enable. 0: Disable 1: Enable the reset of the counter if a CMP0 event occurs

4.5.4.18 IEP_TMR_CMP_STS Register (offset = 44h) [reset = 0h]

IEP_TMR_CMP_STS is shown in Figure 4-209 and described in Table 4-197.

COMPARE STATUS

Figure 4-209. IEP_TMR_CMP_STS Register

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
RESERVED																								CMP_HIT							
R-0h																								R/W1toClr-0h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-197. IEP_TMR_CMP_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-0	CMP_HIT	R/W1toClr	0h	Status bit for each of the compare registers, where CMP_HIT[n] maps to CMP[n]. "Match" indicates the current counter is greater than or equal to the compare value. Note it is the firmware's responsibility to handle the IEP overflow. 0: Match has not occurred. 1: Match occurred. The associated hardware event signal will assert and remain high until the status is cleared.

4.5.4.19 IEP_TMR_CMP0 Register (offset = 48h) [reset = 0h]

IEP_TMR_CMP0 is shown in Figure 4-210 and described in Table 4-198.

COMPARE0

Figure 4-210. IEP_TMR_CMP0 Register

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
CMP0																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-198. IEP_TMR_CMP0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP0	R/W	0h	Compare 0 value

4.5.4.20 IEP_TMR_CMP1 Register (offset = 4Ch) [reset = 0h]

IEP_TMR_CMP1 is shown in Figure 4-211 and described in Table 4-199.

COMPARE1

Figure 4-211. IEP_TMR_CMP1 Register

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
CMP1																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-199. IEP_TMR_CMP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP1	R/W	0h	Compare 1 value

4.5.4.21 IEP_TMR_CMP2 Register (offset = 50h) [reset = 0h]

IEP_TMR_CMP2 is shown in Figure 4-212 and described in Table 4-200.

COMPARE2

Figure 4-212. IEP_TMR_CMP2 Register

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
CMP2																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-200. IEP_TMR_CMP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP2	R/W	0h	Compare 2 value

4.5.4.22 IEP_TMR_CMP3 Register (offset = 54h) [reset = 0h]

IEP_TMR_CMP3 is shown in Figure 4-213 and described in Table 4-201.

COMPARE3

Figure 4-213. IEP_TMR_CMP3 Register

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
CMP3																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-201. IEP_TMR_CMP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP3	R/W	0h	Compare 3 value

4.5.4.23 IEP_TMR_CMP4 Register (offset = 58h) [reset = 0h]

IEP_TMR_CMP4 is shown in Figure 4-214 and described in Table 4-202.

COMPARE4

Figure 4-214. IEP_TMR_CMP4 Register

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
CMP4																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-202. IEP_TMR_CMP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP4	R/W	0h	Compare 4 value

4.5.4.24 IEP_TMR_CMP5 Register (offset = 5Ch) [reset = 0h]

IEP_TMR_CMP5 is shown in Figure 4-215 and described in Table 4-203.

COMPARE5

Figure 4-215. IEP_TMR_CMP5 Register

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
CMP5																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-203. IEP_TMR_CMP5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP5	R/W	0h	Compare 5 value

4.5.4.25 IEP_TMR_CMP6 Register (offset = 60h) [reset = 0h]

IEP_TMR_CMP6 is shown in Figure 4-216 and described in Table 4-204.

COMPARE6

Figure 4-216. IEP_TMR_CMP6 Register

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
CMP6																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-204. IEP_TMR_CMP6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP6	R/W	0h	Compare 6 value

4.5.4.26 IEP_TMR_CMP7 Register (offset = 64h) [reset = 0h]

IEP_TMR_CMP7 is shown in Figure 4-217 and described in Table 4-205.

COMPARE7

Figure 4-217. IEP_TMR_CMP7 Register

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
CMP7																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-205. IEP_TMR_CMP7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMP7	R/W	0h	Compare 7 value

4.5.4.27 IEP_TMR_RXIPG0 Register (offset = 80h) [reset = FFFF0000h]

IEP_TMR_RXIPG0 is shown in Figure 4-218 and described in Table 4-206.

RX InterPackage Gap (IPG) 0

Figure 4-218. IEP_TMR_RXIPG0 Register

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
RX_MIN_IPG																RX_IPG															
R/WtoReset-FFFFh																R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-206. IEP_TMR_RXIPG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MIN_IPG	R/WtoReset	FFFFh	Defines the minimum number of iep_clk/ocp_clk cycles that is RX_DV is sampled low. It stores the smallest RX_IPG duration. It can be read at any time and gets updated after RX_IPG is updated, if RX_MIN_IPG is greater than RX_IPG.
15-0	RX_IPG	R	0h	Records the current number of iep_clk/ocp_clk cycles RX_DV is sampled low. Value is updated after RX_DV transitions from low to high. It will saturate at 0xffff.

4.5.4.28 IEP_TMR_RXIPG1 Register (offset = 84h) [reset = FFFF0000h]

IEP_TMR_RXIPG1 is shown in Figure 4-219 and described in Table 4-207.

RX InterPackage Gap (IPG) 1

Figure 4-219. IEP_TMR_RXIPG1 Register

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
RX_MIN_IPG																RX_IPG															
R/WtoReset-FFFFh																R-0h															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-207. IEP_TMR_RXIPG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MIN_IPG	R/WtoReset	FFFFh	Defines the minimum number of iep_clk/ocp_clk cycles that is RX_DV is sampled low. It stores the smallest RX_IPG duration. It can be read at any time and gets updated after RX_IPG is updated, if RX_MIN_IPG is greater than RX_IPG.
15-0	RX_IPG	R	0h	Records the current number of iep_clk/ocp_clk cycles RX_DV is sampled low. Value is updated after RX_DV transitions from low to high. It will saturate at 0xffff.

4.5.4.29 IEP_SYNC_CTRL Register (Offset = 100h) [reset = 0h]

IEP_SYNC_CTRL is shown in Figure 4-220 and described in Table 4-208.

SYNC GENERATION CONTROL

Figure 4-220. IEP_SYNC_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							SYNC1_IND_EN
R-0h							R/W-0h
7	6	5	4	3	2	1	0
SYNC1_CYCLIC_EN	SYNC1_ACK_EN	SYNC0_CYCLIC_EN	SYNC0_ACK_EN	RESERVED	SYNC1_EN	SYNC0_EN	SYNC_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 4-208. IEP_SYNC_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	
8	SYNC1_IND_EN	R/W	0h	SYNC1 independent mode enable. Independent mode means the SYNC1 signal can be different from SYNC0. 0: Dependent mode 1: Independent mode
7	SYNC1_CYCLIC_EN	R/W	0h	SYNC1 single shot or cyclic/auto generation mode enable 0: Disable, single shot mode 1: Enable, cyclic generation mode
6	SYNC1_ACK_EN	R/W	0h	SYNC1 acknowledgement mode enable 0: Disable, SYNC1 will go low after pulse width is met. 1: Enable, SYNC1 will remain asserted until receiving software acknowledges by reading SYNC1_STATUS which clears on read.
5	SYNC0_CYCLIC_EN	R/W	0h	SYNC0 single shot or cyclic/auto generation mode enable 0: Disable, single shot mode 1: Enable, cyclic generation mode
4	SYNC0_ACK_EN	R/W	0h	SYNC0 acknowledgement mode enable 0: Disable, SYNC0 will go low after pulse width is met. 1: Enable, SYNC0 will remain asserted until receiving software acknowledges by reading SYNC1_STATUS which clears on read.
3	RESERVED	R	0h	
2	SYNC1_EN	R/W	0h	SYNC1 generation enable 0: Disable 1: Enable
1	SYNC0_EN	R/W	0h	SYNC0 generation enable 0: Disable 1: Enable

Table 4-208. IEP_SYNC_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SYNC_EN	R/W	0h	SYNC generation enable 0: Disable the generation and clocking of SYNC0 and SYNC1 logic 1: Enables SYNC0 and SYNC1 generation

4.5.4.30 IEP_SYNC_FIRST_STAT Register (Offset = 104h) [reset = 0h]

IEP_SYNC_FIRST_STAT is shown in Figure 4-221 and described in Table 4-209.

SYNC GENERATION FIRST EVENT STATUS

Figure 4-221. IEP_SYNC_FIRST_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FIRST_SYNC1	FIRST_SYNC0
R-0h						R-0h	R-0h

Table 4-209. IEP_SYNC_FIRST_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	FIRST_SYNC1	R	0h	SYNC1 First Event status. This bit is cleared when SYNC1_EN = 0. 0: Not occurred 1: Occurred
0	FIRST_SYNC0	R	0h	SYNC0 First Event status. This bit is cleared when SYNC0_EN = 0. 0: Not occurred 1: Occurred

4.5.4.31 IEP_SYNC0_STAT Register (Offset = 108h) [reset = 0h]

IEP_SYNC0_STAT is shown in Figure 4-222 and described in Table 4-210.

SYNC0 STATUS

Figure 4-222. IEP_SYNC0_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SYNC0_PEND
R-0h							R/W1C-0h

Table 4-210. IEP_SYNC0_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	SYNC0_PEND	R/W1C	0h	SYNC0 pending state. 0: Not pending 1: Pending or the SYNC0_PEND has occurred when SYNC0_ACK_EN = 0 (Disable).

4.5.4.32 IEP_SYNC1_STAT Register (Offset = 10Ch) [reset = 0h]

IEP_SYNC1_STAT is shown in Figure 4-223 and described in Table 4-211.

SYNC1 STATUS

Figure 4-223. IEP_SYNC1_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							SYNC1_PEND
R-0h							R/W1C-0h

Table 4-211. IEP_SYNC1_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	
0	SYNC1_PEND	R/W1C	0h	SYNC1 pending state. 0: Not pending 1: Pending or the SYNC1_PEND has occurred when SYNC1_ACK_EN = 0 (Disable).

4.5.4.33 IEP_SYNC_PWIDTH Register (Offset = 110h) [reset = 0h]

IEP_SYNC_PWIDTH is shown in Figure 4-224 and described in Table 4-212.

SYNC PULSE WIDTH CONFIGURE

Figure 4-224. IEP_SYNC_PWIDTH Register

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
SYNC_HPW																															
R/W-0h																															

Table 4-212. IEP_SYNC_PWIDTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC_HPW	R/W	0h	Defines the number of clock cycles SYNC0/1 will be high. Note if SYNC0/1 is disabled during pulse width time (that is, SYNC_CTRL[SYNC0_EN | SYNC1_EN | SYNC_EN] = 0), the ongoing pulse will be terminated. 0: 1 clock cycle. 1: 2 clock cycles. N: N+1 clock cycles.

4.5.4.34 IEP_SYNC0_PERIOD Register (Offset = 114h) [reset = 1h]

IEP_SYNC0_PERIOD is shown in Figure 4-225 and described in Table 4-213.

SYNC PERIOD CONFIGURE

Figure 4-225. IEP_SYNC0_PERIOD Register

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
SYNC0_PERIOD																															
R/W-1h																															

Table 4-213. IEP_SYNC0_PERIOD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC0_PERIOD	R/W	1h	Defines the period between the rising edges of SYNC0. 0: reserved. 1: 2 clock cycles period. N: N+1 clock cycles period.

4.5.4.35 IEP_SYNC1_DELAY Register (Offset = 118h) [reset = 0h]

IEP_SYNC1_DELAY is shown in Figure 4-226 and described in Table 4-214.

SYNC CTRL

Figure 4-226. IEP_SYNC1_DELAY Register

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
SYNC1_DELAY																															
R/W-0h																															

Table 4-214. IEP_SYNC1_DELAY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC1_DELAY	R/W	0h	When SYNC1_IND_EN = 0, defines number of clock cycles from the start of SYNC0 to the start of SYNC1. Note this is the delay before the start of SYNC1. 0: No delay. 1: 1 clock cycle delay. N: N clock cycles delay. When SYNC1_IND_EN = 1, defines the period between the rising edges of SYNC1. 0: reserved. 1: 2 clock cycles period. N: N+1 clock cycles period.

4.5.4.36 IEP_SYNC_START Register (Offset = 11Ch) [reset = 0h]

IEP_SYNC_START is shown in Figure 4-227 and described in Table 4-215.

SYNC START CONFIGURE

Figure 4-227. IEP_SYNC_START Register

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
SYNC_START																															
R/W-0h																															

Table 4-215. IEP_SYNC_START Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SYNC_START	R/W	0h	Defines the start time after the activation event. 0: 1 clock cycle delay. 1: 2 clock cycles delay. N: N+1 clock cycles delay.

4.5.4.37 IEP_WD_PREDIV Register (offset = 200h) [reset = 4E20h]

IEP_WD_PREDIV is shown in Figure 4-228 and described in Table 4-216.

WATCHDOG PRE-DIVIDER

Figure 4-228. IEP_WD_PREDIV Register

31	30	29	28	27	26	25	24
Reserved							
R-0h							
23	22	21	20	19	18	17	16
Reserved							
R-0h							
15	14	13	12	11	10	9	8
PRE_DIV							
R/WtoClr-4E20h							
7	6	5	4	3	2	1	0
PRE_DIV							
R/WtoClr-4E20h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-216. IEP_WD_PREDIV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	Reserved	R	0h	
15-0	PRE_DIV	R/WtoClr	4E20h	Defines the number of iep_clk cycles per WD clock event. Note that the WD clock is a free-running clock. The value 0x4e20 (or 20000) generates a rate of 100 us if iep_clk is 200 MHz. seconds/(WD event) = (clock cycles per WD event)/(clock cycles per second) = 20000/(200 x [10]^6) = 100 us

4.5.4.38 IEP_PDI_WD_TIM Register (offset = 204h) [reset = 3E8h]

IEP_PDI_WD_TIM is shown in Figure 4-229 and described in Table 4-217.

PDI WATCHDOG TIMER CONFIGURE

Figure 4-229. IEP_PDI_WD_TIM Register

31	30	29	28	27	26	25	24
Reserved							
R-0h							
23	22	21	20	19	18	17	16
Reserved							
R-0h							
15	14	13	12	11	10	9	8
PDI_WD_TIME							
R/WtoReset-3E8h							
7	6	5	4	3	2	1	0
PDI_WD_TIME							
R/WtoReset-3E8h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-217. IEP_PDI_WD_TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	Reserved	R	0h	
15-0	PDI_WD_TIME	R/WtoReset	3E8h	Defines the number of WD ticks (or increments) for PDI WD, that is, the number of WD increments. If PRE_DIV is set to 100 us, then the value 0x03e8 (or 1000) provides a rate of 100ms. Read returns the current count. Counter is reset by software write to register or when Digital Data In capture occurs. WD is disabled if WD time is set to 0x0. Note when an expiration event occurs, the expiration counter (PDI_EXP_CNT) increments and status (PDI_WD_STAT) clears.

4.5.4.39 IEP_PD_WD_TIM Register (offset = 208h) [reset = 3E8h]

IEP_PD_WD_TIM is shown in Figure 4-230 and described in Table 4-218.

PD WATCHDOG TIMER CONFIGURE

Figure 4-230. IEP_PD_WD_TIM Register

31	30	29	28	27	26	25	24
Reserved							
R-0h							
23	22	21	20	19	18	17	16
Reserved							
R-0h							
15	14	13	12	11	10	9	8
PD_WD_TIME							
R/WtoReset-3E8h							
7	6	5	4	3	2	1	0
PD_WD_TIME							
R/WtoReset-3E8h							

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-218. IEP_PD_WD_TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	Reserved	R	0h	
15-0	PD_WD_TIME	R/WtoReset	3E8h	Defines the number of WD ticks (or increments) for PDI WD, that is, the number of WD increments. If PRE_DIV is set to 100 us, then 0x03e8 (or 1000) provides a rate of 100ms. Read returns the current count. Counter is reset by software write to register or every write access to Sync Managers with WD trigger enable bit set. WD is disabled if WD time is set to 0x0. Expiration actions: Increment expiration counter, clear status. Digital Data out forced to zero if pr1_edio_oe_ext = 1 and DIGIO_EXT.SW_DATA_OUT_UPDATE = 0.

4.5.4.40 IEP_WD_STATUS Register (offset = 20Ch) [reset = 10001h]

IEP_WD_STATUS is shown in Figure 4-231 and described in Table 4-219.

WATCHDOG STATUS

Figure 4-231. IEP_WD_STATUS Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-219. IEP_WD_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	Reserved	R	0h	
16	PDI_WD_STAT	R	1h	WD PDI status. 0: Expired (PDI_WD_EXP event generated) 1: Active or disabled
15-1	Reserved	R	0h	
0	PD_WD_STAT	R	1h	WD PD status (triggered by Sync Mangers status). Note reading this register clears application event request. 0: Expired (PD_WD_EXP event generated) 1: Active or disabled

4.5.4.41 IEP_WD_EXP_CNT Register (offset = 210h) [reset = 0h]

IEP_WD_EXP_CNT is shown in Figure 4-232 and described in Table 4-220.

WATCHDOG TIMER EXPIRATION COUNTER

Figure 4-232. IEP_WD_EXP_CNT Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-220. IEP_WD_EXP_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	Reserved	R	0h	
15-8	PD_EXP_CNT	R/WtoClr	0h	WD PD expiration counter. Counter increments on every PD time out and stops at 0xff.
7-0	PDI_EXP_CNT	R/WtoClr	0h	WD PDI expiration counter. Counter increments on every PDI time out and stops at 0xff.

4.5.4.42 IEP_WD_CTRL Register (offset = 214h) [reset = 0h]

IEP_WD_CTRL is shown in Figure 4-233 and described in Table 4-221.

WATCHDOG CONTROL

Figure 4-233. IEP_WD_CTRL Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-221. IEP_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	Reserved	R	0h	
16	PDI_WD_EN	R/W	0h	Enable WD PDI 0: Disable 1: Enable
15-1	Reserved	R	0h	
0	PD_WD_EN	R/W	0h	Enable WD PD 0: Disable 1: Enable

4.5.4.43 IEP_DIGIO_CTRL Register (offset = 300h) [reset = 4h]

IEP_DIGIO_CTRL is shown in Figure 4-234 and described in Table 4-222.

DIGITAL INPUT OUTPUT CONTROL

Figure 4-234. IEP_DIGIO_CTRL Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-222. IEP_DIGIO_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-6	OUT_MODE	R/W	0h	Defines event that triggers data out to be updated. Note if OUTVALID_MODE is set, then data out is forced to zero if a WD PD expiration occurs (PD_WD_EXP) from the WD block and pr1_edio_oe_ext = 1. 0: PRU0/1_RX_EOF 1: Reserved 2: DC SYNC0 event 3: DC SYNC1 event
5-4	IN_MODE	R/W	0h	Defines event that triggers data in to be sampled. 0: PRU0/1_RX_SOF 1: Rising edge of pr1_edio_latch_in 2: DC rising edge of SYNC0 event 3: DC rising edge of SYNC1 event
3	WD_MODE	R/W	0h	Defines watchdog behavior. 0: Outputs are reset immediately after watchdog expires 1: Outputs are reset with next output event that follows watchdog expiration
2	BIDI_MODE	R	1h	Defines the digital input/output direction. 0: Unidirectional mode: digital input/output direction of pins configured individually 1: Bidirectional mode: all I/O pins are bidirectional and direction configuration is Ignored
1	OUTVALID_MODE	R/W	0h	Defines the outvalid mode behavior. 0: Output event signaling 1: Output data is updated if watchdog is triggered. Output data is forced to zero if PD_WD_EXP from the WD block and pr1_edio_oe_ext = 1
0	OUTVALID_POL	R	0h	Defines outvalid polarity. 0: Active high 1: Active low

4.5.4.44 IEP_DIGIO_STATUS Register (offset = 304h) [reset = 0h]

IEP_DIGIO_STATUS is shown in Figure 4-235 and described in Table 4-223.

DIGITAL INPUT OUTPUT STATUS

Figure 4-235. IEP_DIGIO_STATUS Register

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
DIGIO_STAT																															
R-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-223. IEP_DIGIO_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DIGIO_STAT	R	0h	Reserved

4.5.4.45 IEP_DIGIO_DATA_IN Register (offset = 308h) [reset = 0h]

Register mask: 0h

IEP_DIGIO_DATA_IN is shown in Figure 4-236 and described in Table 4-224.

DIGITAL DATA INPUT

Figure 4-236. IEP_DIGIO_DATA_IN Register

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
DATA_IN																															
R-X																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-224. IEP_DIGIO_DATA_IN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_IN	R	X	Data input. Digital inputs can be configured to be sampled in four ways. 1: Digital inputs are sampled at the start of each frame. The SOF signal can be used externally to update the input data, because the SOF is signaled before input data is sampled. 2: The sample time can be controlled externally by using the pr1_edio_latch_in signal. 3: Digital inputs are sampled at SYNC0 events. 4: Digital inputs are sampled at SYNC1 events. These can be configured by in_mode.

4.5.4.46 IEP_DIGIO_DATA_IN_RAW Register (offset = 30Ch) [reset = 0h]

Register mask: 0h

IEP_DIGIO_DATA_IN_RAW is shown in Figure 4-237 and described in Table 4-225.

DIGITAL DATA INPUT DIRECT SAMPLE

Figure 4-237. IEP_DIGIO_DATA_IN_RAW Register

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
DATA_IN_RAW																															
R-X																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-225. IEP_DIGIO_DATA_IN_RAW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_IN_RAW	R	X	Raw data input. Direct sample of pr1_edio_data_in[31:0].

4.5.4.47 IEP_DIGIO_DATA_OUT Register (offset = 310h) [reset = 0h]

IEP_DIGIO_DATA_OUT is shown in Figure 4-238 and described in Table 4-226.

DIGITAL DATA OUTPUT

Figure 4-238. IEP_DIGIO_DATA_OUT Register

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
DATA_OUT																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-226. IEP_DIGIO_DATA_OUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_OUT	R/W	0h	Data output. Digital outputs can be configured to be updated in four ways. 1: Digital outputs are updated at the end of each frame (EOF mode). 2: Digital outputs are updated with SYNC0 events 3: Digital outputs are updated SYNC1events. 4: Digital outputs are updated at the end of a frame which triggered the Process Data Watchdog. Digital Outputs are only updated if the frame was correct (WD_TRIG mode). These can be configured by out_mode.

4.5.4.48 IEP_DIGIO_DATA_OUT_EN Register (offset = 314h) [reset = 0h]

IEP_DIGIO_DATA_OUT_EN is shown in Figure 4-239 and described in Table 4-227.

DIGITAL DATA OUT ENABLE

Figure 4-239. IEP_DIGIO_DATA_OUT_EN Register

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
DATA_OUT_EN																															
R/W-0h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-227. IEP_DIGIO_DATA_OUT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA_OUT_EN	R/W	0h	Enables tri-state control for pr1_edio_data_out[31:0].

4.5.4.49 IEP_DIGIO_EXP Register (offset = 318h) [reset = 20h]

IEP_DIGIO_EXP is shown in Figure 4-240 and described in Table 4-228.

DIGIO EXPANSION REGISTER

Figure 4-240. IEP_DIGIO_EXP Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-228. IEP_DIGIO_EXP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	
13	EOF_SEL	R/W	0h	Defines the source of RX_EOF is used for pr1_edio_data_in capture. 0: PRU0_RX_EOF 1: PRU1_RX_EOF
12	SOF_SEL	R/W	0h	Defines the source of RX_SOF is used for pr1_edio_data_in capture. 0: PRU0_RX_SOF 1: PRU1_RX_SOF
11-8	SOF_DLY	R/W	0h	Defines the number of iep_clk cycles delay before pr1_edio_data_in captures.
7-4	OUTVALID_DLY	R/W	2h	Define the number of iep_clk cycles delay on assertion of pr1_edio_outvalid. Minimum is 2 clock cycles Maximum is 16 clock cycles
3	RESERVED	R	0h	
2	SW_OUTVALID	R/W	0h	pr1_edio_outvalid = SW_OUTVALID, only if OUTVALID_OVR_EN is set.
1	OUTVALID_OVR_EN	R/W	0h	Software override enable 0: Disable override 1: Enable override
0	SW_DATA_OUT_UPDATE	R/W	0h	Defines the value of pr1_edio_data_out when OUTVALID_OVR_EN = 1. Read 1: Start bit event occurred Read 0: Start bit event has not occurred Write 1: pr1_edio_data_out by software data out. Write 0: No Effect

4.5.5 PRU_ICSS_UART Registers

The system programmer has access to and control over any of the UART registers that are listed in Table 4-229. These registers, which control UART operations, receive data, and transmit data, are available at 32-bit addresses in the device memory map.

- RBR, THR, and DLL share one address. When the DLAB bit in LCR is 0, reading from the address gives the content of RBR, and writing to the address modifies THR. When DLAB = 1, all accesses at the address read or modify DLL. DLL can also be accessed with address offset 20h.
- IER and DLH share one address. When DLAB = 0, all accesses read or modify IER. When DLAB = 1,

all accesses read or modify DLH. DLH can also be accessed with address offset 24h.

- IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing modifies FCR.

Table 4-229. PRU_ICSS_UART Registers

Offset	Acronym	Register Description	Section
0h	RBR	Receiver Buffer Register (read only)	Section 4.5.5.1
0h	THR	Transmitter Holding Register (write only)	Section 4.5.5.2
4h	IER	Interrupt Enable Register	Section 4.5.5.3
8h	IIR	Interrupt Identification Register (read only)	Section 4.5.5.4
8h	FCR	FIFO Control Register (write only)	Section 4.5.5.5
Ch	LCR	Line Control Register	Section 4.5.5.6
10h	MCR	Modem Control Register	Section 4.5.5.7
14h	LSR	Line Status Register	Section 4.5.5.8
18h	MSR	Modem Status Register	Section 4.5.5.9
1Ch	SCR	Scratch Pad Register	Section 4.5.5.10
20h	DLL	Divisor LSB Latch	Section 4.5.5.11
24h	DLH	Divisor MSB Latch	Section 4.5.5.11
28h	REVID1	Revision Identification Register 1	Section 4.5.5.12
2Ch	REVID2	Revision Identification Register 2	Section 4.5.5.12
30h	PWREMU_MGMT	Power and Emulation Management Register	Section 4.5.5.13
34h	MDR	Mode Definition Register	Section 4.5.5.14

4.5.5.1 Receiver Buffer Register (RBR)

The receiver buffer register (RBR) is shown in Figure 4-241 and described in Table 4-230.

The UART receiver section consists of a receiver shift register (RSR) and a receiver buffer register (RBR). When the UART is in the FIFO mode, RBR is a 16-byte FIFO. Timing is supplied by the 16x receiver clock or 13x receiver clock by programming OSM_SEL bit field of MDR register. Receiver section control is a function of the line control register (LCR).

RSR receives serial data from the UARTn_RXD pin. Then RSR concatenates the data and moves it into RBR (or the receiver FIFO). In the non-FIFO mode, when a character is placed in RBR and the receiver data-ready interrupt is enabled (DR = 1 in IER), an interrupt is generated. This interrupt is cleared when the character is read from RBR. In the FIFO mode, the interrupt is generated when the FIFO is filled to the trigger level selected in the FIFO control register (FCR), and it is cleared when the FIFO contents drop below the trigger level.

Access considerations:

RBR, THR, and DLL share one address. To read RBR, write 0 to the DLAB bit in LCR, and read from the shared address. When DLAB = 0, writing to the shared address modifies THR. When DLAB = 1, all accesses at the shared address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that RBR and THR are always selected at the shared address.

Figure 4-241. Receiver Buffer Register (RBR)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		DATA	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-230. Receiver Buffer Register (RBR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Received data

4.5.5.2 Transmitter Holding Register (THR)

The transmitter holding register (THR) is shown in Figure 4-242 and described in Table 4-231.

The UART transmitter section consists of a transmitter hold register (THR) and a transmitter shift register (TSR). When the UART is in the FIFO mode, THR is a 16-byte FIFO. Transmitter section control is a function of the line control register (LCR).

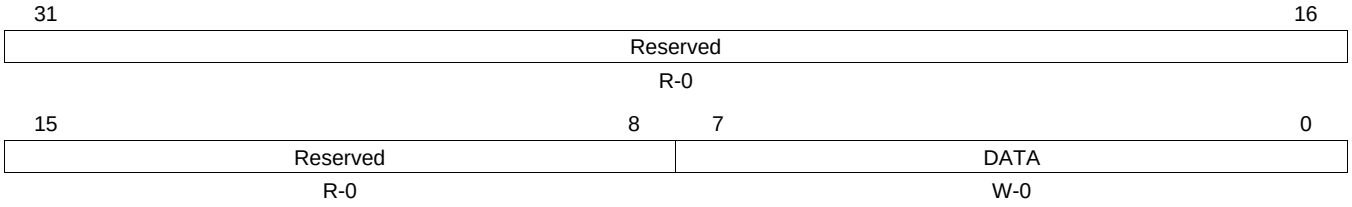
THR receives data from the internal data bus and when TSR is idle, the UART moves the data from THR to TSR. The UART serializes the data in TSR and transmits the data on the TX pin. In the non-FIFO mode, if THR is empty and the THR empty (THRE) interrupt is enabled (ETBEI = 1 in IER), an interrupt is generated. This interrupt is cleared when a character is loaded into THR or the interrupt identification register (IIR) is read. In the FIFO mode, the interrupt is generated when the transmitter FIFO is empty, and it is cleared when at least one byte is loaded into the FIFO or IIR is read.

Access considerations:

RBR, THR, and DLL share one address. To load THR, write 0 to the DLAB bit of LCR, and write to the shared address. When DLAB = 0, reading from the shared address gives the content of RBR. When DLAB = 1, all accesses at the address read or modify DLL.

DLL also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that RBR and THR are always selected at the shared address.

Figure 4-242. Transmitter Holding Register (THR)



LEGEND: R = Read only; W = Write only; -*n* = value after reset

Table 4-231. Transmitter Holding Register (THR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DATA	0-FFh	Data to transmit

4.5.5.3 Interrupt Enable Register (IER)

The interrupt enable register (IER) is used to individually enable or disable each type of interrupt request that can be generated by the UART. Each interrupt request that is enabled in IER is forwarded to the CPU. IER is shown in Figure 4-243 and described in Table 4-232.

Access considerations:

IER and DLH share one address. To read or modify IER, write 0 to the DLAB bit in LCR. When DLAB = 1, all accesses at the shared address read or modify DLH.

DLH also has a dedicated address. If you use the dedicated address, you can keep DLAB = 0, so that IER is always selected at the shared address.

Figure 4-243. Interrupt Enable Register (IER)

31	16
Reserved	
R-0	

15	4	3	2	1	0
Reserved		Rsvd	ELSI	ETBEI	ERBI
R-0		R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-232. Interrupt Enable Register (IER) Field Descriptions

Bit	Field	Value	Description
31-4	Reserved	0	Reserved
3	EDSSI	0	Enable Modem Status Interrupt
2	ELSI		Receiver line status interrupt enable.
		0	Receiver line status interrupt is disabled.
		1	Receiver line status interrupt is enabled.
1	ETBEI		Transmitter holding register empty interrupt enable.
		0	Transmitter holding register empty interrupt is disabled.
		1	Transmitter holding register empty interrupt is enabled.
0	ERBI		Receiver data available interrupt and character timeout indication interrupt enable.
		0	Receiver data available interrupt and character timeout indication interrupt is disabled.
		1	Receiver data available interrupt and character timeout indication interrupt is enabled.

4.5.5.4 Interrupt Identification Register (IIR)

The interrupt identification register (IIR) is a read-only register at the same address as the FIFO control register (FCR), which is a write-only register. When an interrupt is generated and enabled in the interrupt enable register (IER), IIR indicates that an interrupt is pending in the IPEND bit and encodes the type of interrupt in the INTID bits. Reading IIR clears any THR empty (THRE) interrupts that are pending.

IIR is shown in Figure 4-244 and described in Figure 4-244.

The UART has an on-chip interrupt generation and prioritization capability that permits flexible communication with the CPU. The UART provides three priority levels of interrupts:

- Priority 1 - Receiver line status (highest priority)
- Priority 2 - Receiver data ready or receiver timeout
- Priority 3 - Transmitter holding register empty

The FIFOEN bit in IIR can be checked to determine whether the UART is in the FIFO mode or the non-FIFO mode.

Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit in LCR, reading from the address gives the content of IIR, and writing to the address modifies FCR.

Figure 4-244. Interrupt Identification Register (IIR)

31	16
Reserved	
R-0	

15	8	7	6	5	4	3	1	0
Reserved		FIFOEN		Reserved		INTID		IPEND
R-0		R-0		R-0		R-0		R-1

LEGEND: R = Read only; -*n* = value after reset

Table 4-233. Interrupt Identification Register (IIR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	FIFOEN	0-3h	FIFOs enabled.
		0	Non-FIFO mode
		1h-2h	Reserved
		3h	FIFOs are enabled. FIFOEN bit in the FIFO control register (FCR) is set to 1.
5-4	Reserved	0	Reserved
3-1	INTID	0-7h	Interrupt type. See Table 4-234.
		0	Reserved
		1h	Transmitter holding register empty (priority 3)
		2h	Receiver data available (priority 2)
		3h	Receiver line status (priority 1, highest)
		4h-5h	Reserved
		6h	Character timeout indication (priority 2)
		7h	Reserved
0	IPEND		Interrupt pending. When any UART interrupt is generated and is enabled in IER, IPEND is forced to 0. IPEND remains 0 until all pending interrupts are cleared or until a hardware reset occurs. If no interrupts are enabled, IPEND is never forced to 0.
		0	Interrupts pending.
		1	No interrupts pending.

Table 4-234. Interrupt Identification and Interrupt Clearing Information

Priority Level	IIR Bits 3	2	1	0	Interrupt Type	Interrupt Source	Event That Clears Interrupt
None	0	0	0	1	None	None	None
1	0	1	1	0	Receiver line status	Overrun error, parity error, framing error, or break is detected.	For an overrun error, reading the line status register (LSR) clears the interrupt. For a parity error, framing error, or break, the interrupt is cleared only after all the erroneous data have been read.
2	0	1	0	0	Receiver data-ready	Non-FIFO mode: Receiver data is ready.	Non-FIFO mode: The receiver buffer register (RBR) is read.
						FIFO mode: Trigger level reached. If four character times (see Table 4-27) pass with no access of the FIFO, the interrupt is asserted again.	FIFO mode: The FIFO drops below the trigger level.(1)
2	1	1	0	0	Receiver time-out	FIFO mode only: No characters have been removed from or input to the receiver FIFO during the last four character times (see Table 4-27), and there is at least one character in the receiver FIFO during this time.	One of the following events: • A character is read from the receiver FIFO.(1) • A new character arrives in the receiver FIFO. • The URRST bit in the power and emulation management register (PWREMU_MGMT) is loaded with 0.
3	0	0	1	0	Transmitter holding register empty	Non-FIFO mode: Transmitter holding register (THR) is empty. FIFO mode: Transmitter FIFO is empty.	A character is written to the transmitter holding register (THR) or the interrupt identification register (IIR) is read.

(1) In the FIFO mode, the receiver data-ready interrupt or receiver time-out interrupt is cleared by the CPU or by the DMA controller, whichever reads from the receiver FIFO first.

4.5.5.5 FIFO Control Register (FCR)

The FIFO control register (FCR) is a write-only register at the same address as the interrupt identification register (IIR), which is a read-only register. Use FCR to enable and clear the FIFOs and to select the receiver FIFO trigger level FCR is shown in Figure 4-245 and described in Table 4-235. The FIFOEN bit must be set to 1 before other FCR bits are written to or the FCR bits are not programmed.

Access consideration:

IIR and FCR share one address. Regardless of the value of the DLAB bit, reading from the address gives the content of IIR, and writing to the address modifies FCR.

CAUTION

For proper communication between the UART and the EDMA controller, the DMAMODE1 bit must be set to 1. Always write a 1 to the DMAMODE1 bit, and after a hardware reset, change the DMAMODE1 bit from 0 to 1.

Figure 4-245. FIFO Control Register (FCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved							
R-0							

7	6	5	4	3	2	1	0
RXFIFTL		Reserved		DMAMODE1[(1)]	TXCLR	RXCLR	FIFOEN
W-0		R-0		W-0	W1C-0	W1C-0	W-0

LEGEND: R = Read only; W = Write only; W1C = Write 1 to clear (writing 0 has no effect); -*n* = value after reset

(1) Always write 1 to the DMAMODE1 bit. After a hardware reset, change the DMAMODE1 bit from 0 to 1. DMAMODE = 1 is required for proper communication between the UART and the DMA controller.

Table 4-235. FIFO Control Register (FCR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-6	RXFIFTL	0-3h	Receiver FIFO trigger level. RXFIFTL sets the trigger level for the receiver FIFO. When the trigger level is reached, a receiver data-ready interrupt is generated (if the interrupt request is enabled). Once the FIFO drops below the trigger level, the interrupt is cleared.
		0	1 byte
		1h	4 bytes
		2h	8 bytes
		3h	14 bytes
5-4	Reserved	0	Reserved
3	DMAMODE1		DMA MODE1 enable if FIFOs are enabled. Always write 1 to DMAMODE1. After a hardware reset, change DMAMODE1 from 0 to 1. DMAMOD1 = 1 is a requirement for proper communication between the UART and the EDMA controller.
		0	DMA MODE1 is disabled.
		1	DMA MODE1 is enabled.
2	TXCLR		Transmitter FIFO clear. Write a 1 to TXCLR to clear the bit.
		0	No effect.
		1	Clears transmitter FIFO and resets the transmitter FIFO counter. The shift register is not cleared.
1	RXCLR		Receiver FIFO clear. Write a 1 to RXCLR to clear the bit.
		0	No effect.
		1	Clears receiver FIFO and resets the receiver FIFO counter. The shift register is not cleared.
0	FIFOEN		Transmitter and receiver FIFOs mode enable. FIFOEN must be set before other FCR bits are written to or the FCR bits are not programmed. Clearing this bit clears the FIFO counters.
		0	Non-FIFO mode. The transmitter and receiver FIFOs are disabled, and the FIFO pointers are cleared.
		1	FIFO mode. The transmitter and receiver FIFOs are enabled.

4.5.5.6 Line Control Register (LCR)

The line control register (LCR) is shown in Figure 4-246 and described in Table 4-236.

The system programmer controls the format of the asynchronous data communication exchange by using LCR. In addition, the programmer can retrieve, inspect, and modify the content of LCR; this eliminates the need for separate storage of the line characteristics in system memory.

Figure 4-246. Line Control Register (LCR)

31								16
Reserved								
R-0								
15	8	7	6	5	4	3	2	1 0
Reserved		DLAB	BC	SP	EPS	PEN	STB	WLS
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-236. Line Control Register (LCR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	DLAB		Divisor latch access bit. The divisor latch registers (DLL and DLH) can be accessed at dedicated addresses or at addresses shared by RBR, THR, and IER. Using the shared addresses requires toggling DLAB to change which registers are selected. If you use the dedicated addresses, you can keep DLAB = 0.
		0	Allows access to the receiver buffer register (RBR), the transmitter holding register (THR), and the interrupt enable register (IER) selected. At the address shared by RBR, THR, and DLL, the CPU can read from RBR and write to THR. At the address shared by IER and DLH, the CPU can read from and write to IER.
		1	Allows access to the divisor latches of the baud generator during a read or write operation (DLL and DLH). At the address shared by RBR, THR, and DLL, the CPU can read from and write to DLL. At the address shared by IER and DLH, the CPU can read from and write to DLH.
6	BC		Break control.
		0	Break condition is disabled.
		1	Break condition is transmitted to the receiving UART. A break condition is a condition where the UARTn_TXD signal is forced to the spacing (cleared) state.
5	SP		Stick parity. The SP bit works in conjunction with the EPS and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 4-237.
		0	Stick parity is disabled.
		1	Stick parity is enabled. • When odd parity is selected (EPS = 0), the PARITY bit is transmitted and checked as set. • When even parity is selected (EPS = 1), the PARITY bit is transmitted and checked as cleared.
4	EPS		Even parity select. Selects the parity when parity is enabled (PEN = 1). The EPS bit works in conjunction with the SP and PEN bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 4-237.
		0	Odd parity is selected (an odd number of logic 1s is transmitted or checked in the data and PARITY bits).
		1	Even parity is selected (an even number of logic 1s is transmitted or checked in the data and PARITY bits).
3	PEN		Parity enable. The PEN bit works in conjunction with the SP and EPS bits. The relationship between the SP, EPS, and PEN bits is summarized in Table 4-237.
		0	No PARITY bit is transmitted or checked.
		1	Parity bit is generated in transmitted data and is checked in received data between the last data word bit and the first STOP bit.

Table 4-236. Line Control Register (LCR) Field Descriptions (continued)

Bit	Field	Value	Description
2	STB		Number of STOP bits generated. STB specifies 1, 1.5, or 2 STOP bits in each transmitted character. When STB = 1, the WLS bit determines the number of STOP bits. The receiver clocks only the first STOP bit, regardless of the number of STOP bits selected. The number of STOP bits generated is summarized in Table 4-238.
		0	1 STOP bit is generated.
		1	WLS bit determines the number of STOP bits: • When WLS = 0, 1.5 STOP bits are generated. • When WLS = 1h, 2h, or 3h, 2 STOP bits are generated.
1-0	WLS	0-3h	Word length select. Number of bits in each transmitted or received serial character. When STB = 1, the WLS bit determines the number of STOP bits.
		0	5 bits
		1h	6 bits
		2h	7 bits
		3h	8 bits

Table 4-237. Relationship Between ST, EPS, and PEN Bits in LCR

ST Bit	EPS Bit	PEN Bit	Parity Option
x	x	0	Parity disabled: No PARITY bit is transmitted or checked
0	0	1	Odd parity selected: Odd number of logic 1s
0	1	1	Even parity selected: Even number of logic 1s
1	0	1	Stick parity selected with PARITY bit transmitted and checked as set
1	1	1	Stick parity selected with PARITY bit transmitted and checked as cleared

Table 4-238. Number of STOP Bits Generated

STB Bit	WLS Bits	Word Length Selected with WLS Bits	Number of STOP Bits Generated	Baud Clock (BCLK) Cycles
0	x	Any word length	1	16
1	0h	5 bits	1.5	24
1	1h	6 bits	2	32
1	2h	7 bits	2	32
1	3h	8 bits	2	32

4.5.5.7 Modem Control Register (MCR)

The modem control register (MCR) is shown in Figure 4-247 and described in Table 4-239. The modem control register provides the ability to enable/disable the autoflow functions, and enable/disable the loopback function for diagnostic purposes.

Figure 4-247. Modem Control Register (MCR)

31	16
Reserved	
R-0	

15	6	5	4	3	2	1	0
Reserved		AFE[1]	LOOP	OUT2	OUT1	RTS[1]	Rsvd
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

(1) All UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.

Table 4-239. Modem Control Register (MCR) Field Descriptions

Bit	Field	Value	Description
31-6	Reserved	0	Reserved
5	AFE		Autoflow control enable. Autoflow control allows the $\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ signals to provide handshaking between UARTs during data transfer. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.
		0	Autoflow control is disabled.
		1	Autoflow control is enabled: • When RTS = 0, $\overline{\text{UARTn\_CTS}}$ is only enabled. • When RTS = 1, $\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ are enabled.
4	LOOP		Loop back mode enable. LOOP is used for the diagnostic testing using the loop back feature.
		0	Loop back mode is disabled.
		1	Loop back mode is enabled. When LOOP is set, the following occur: • The UARTn_TXD signal is set high. • The UARTn_RXD pin is disconnected • The output of the transmitter shift register (TSR) is lopped back in to the receiver shift register (RSR) input.
3	OUT2	0	OUT2 Control Bit
2	OUT1	0	OUT1 Control Bit
1	RTS		RTS control. When AFE = 1, the RTS bit determines the autoflow control enabled. Note that all UARTs do not support this feature, see your device-specific data manual for supported features. If this feature is not available, this bit is reserved and should be cleared to 0.
		0	$\overline{\text{UARTn\_RTS}}$ is disabled, $\overline{\text{UARTn\_CTS}}$ is only enabled.
		1	$\overline{\text{UARTn\_RTS}}$ and $\overline{\text{UARTn\_CTS}}$ are enabled.
0	Reserved	0	Reserved

4.5.5.8 Line Status Register (LSR)

The line status register (LSR) is shown in Figure 4-248 and described in Table 4-240. LSR provides information to the CPU concerning the status of data transfers. LSR is intended for read operations only; do not write to this register. Bits 1 through 4 record the error conditions that produce a receiver line status interrupt.

Figure 4-248. Line Status Register (LSR)

31								16
Reserved								
R-0								
15	8	7	6	5	4	3	2	1	0
Reserved		RXFIFOE	TEMT	THRE	BI	FE	PE	OE	DR
R-0		R-0	R-1	R-1	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 4-240. Line Status Register (LSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	RXFIFOE		Receiver FIFO error.
			In non-FIFO mode:
		0	There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	There is a parity error, framing error, or break indicator in the receiver buffer register (RBR).
			In FIFO mode:
		0	There has been no error, or RXFIFOE was cleared because the CPU read the erroneous character from the receiver FIFO and there are no more errors in the receiver FIFO.
		1	At least one parity error, framing error, or break indicator in the receiver FIFO.
6	TEMT		Transmitter empty (TEMT) indicator.
			In non-FIFO mode:
		0	Either the transmitter holding register (THR) or the transmitter shift register (TSR) contains a data character.
		1	Both the transmitter holding register (THR) and the transmitter shift register (TSR) are empty.
			In FIFO mode:
		0	Either the transmitter FIFO or the transmitter shift register (TSR) contains a data character.
		1	Both the transmitter FIFO and the transmitter shift register (TSR) are empty.
5	THRE		Transmitter holding register empty (THRE) indicator. If the THRE bit is set and the corresponding interrupt enable bit is set (ETBEI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	Transmitter holding register (THR) is not empty. THR has been loaded by the CPU.
		1	Transmitter holding register (THR) is empty (ready to accept a new character). The content of THR has been transferred to the transmitter shift register (TSR).
			In FIFO mode:
		0	Transmitter FIFO is not empty. At least one character has been written to the transmitter FIFO. You can write to the transmitter FIFO if it is not full.
		1	Transmitter FIFO is empty. The last character in the FIFO has been transferred to the transmitter shift register (TSR).

Table 4-240. Line Status Register (LSR) Field Descriptions (continued)

Bit	Field	Value	Description
4	BI		Break indicator. The BI bit is set whenever the receive data input (UARTn_RXD) was held low for longer than a full-word transmission time. A full-word transmission time is defined as the total time to transmit the START, data, PARITY, and STOP bits. If the BI bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver buffer register (RBR).
		1	A break has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No break has been detected, or the BI bit was cleared because the CPU read the erroneous character from the receiver FIFO and the next character to be read from the FIFO has no break indicator.
		1	A break has been detected with the character at the top of the receiver FIFO.
3	FE		Framing error (FE) indicator. A framing error occurs when the received character does not have a valid STOP bit. In response to a framing error, the UART sets the FE bit and waits until the signal on the RX pin goes high. Once the RX signal goes high, the receiver is ready to detect a new START bit and receive new data. If the FE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A framing error has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No framing error has been detected, or the FE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no framing error.
		1	A framing error has been detected with the character at the top of the receiver FIFO.
2	PE		Parity error (PE) indicator. A parity error occurs when the parity of the received character does not match the parity selected with the EPS bit in the line control register (LCR). If the PE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver buffer register (RBR).
		1	A parity error has been detected with the character in the receiver buffer register (RBR).
			In FIFO mode:
		0	No parity error has been detected, or the PE bit was cleared because the CPU read the erroneous data from the receiver FIFO and the next character to be read from the FIFO has no parity error.
		1	A parity error has been detected with the character at the top of the receiver FIFO.
1	OE		Overrun error (OE) indicator. An overrun error in the non-FIFO mode is different from an overrun error in the FIFO mode. If the OE bit is set and the corresponding interrupt enable bit is set (ELSI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. Before the character in the receiver buffer register (RBR) could be read, it was overwritten by the next character arriving in RBR.
			In FIFO mode:
		0	No overrun error has been detected, or the OE bit was cleared because the CPU read the content of the line status register (LSR).
		1	Overrun error has been detected. If data continues to fill the FIFO beyond the trigger level, an overrun error occurs only after the FIFO is full and the next character has been completely received in the shift register. An overrun error is indicated to the CPU as soon as it happens. The new character overwrites the character in the shift register, but it is not transferred to the FIFO.

Table 4-240. Line Status Register (LSR) Field Descriptions (continued)

Bit	Field	Value	Description
0	DR		Data-ready (DR) indicator for the receiver. If the DR bit is set and the corresponding interrupt enable bit is set (ERBI = 1 in IER), an interrupt request is generated.
			In non-FIFO mode:
		0	Data is not ready, or the DR bit was cleared because the character was read from the receiver buffer register (RBR).
		1	Data is ready. A complete incoming character has been received and transferred into the receiver buffer register (RBR).
			In FIFO mode:
		0	Data is not ready, or the DR bit was cleared because all of the characters in the receiver FIFO have been read.
		1	Data is ready. There is at least one unread character in the receiver FIFO. If the FIFO is empty, the DR bit is set as soon as a complete incoming character has been received and transferred into the FIFO. The DR bit remains set until the FIFO is empty again.

4.5.5.9 Modem Status Register (MSR)

The Modem status register (MSR) is shown in Figure 4-249 and described in Table 4-241. MSR provides information to the CPU concerning the status of modem control signals. MSR is intended for read operations only; do not write to this register.

Figure 4-249. Modem Status Register (MSR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CD	RI	DSR	CTS	DCD	TERI	DDSR	DCTS
R-0								R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0

LEGEND: R = Read only; -*n* = value after reset

Table 4-241. Modem Status Register (MSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7	CD	0	Complement of the Carrier Detect input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 3 (OUT2).
6	RI	0	Complement of the Ring Indicator input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 2 (OUT1).
5	DSR	0	Complement of the Data Set Ready input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 0 (DTR).
4	CTS	0	Complement of the Clear To Send input. When the UART is in the diagnostic test mode (loopback mode MCR[4] = 1), this bit is equal to the MCR bit 1 (RTS).
3	DCD	0	Change in DCD indicator bit. DCD indicates that the DCD input has changed state since the last time it was read by the CPU. When DCD is set and the modem status interrupt is enabled, a modem status interrupt is generated.
2	TERI	0	Trailing edge of RI (TERI) indicator bit. TERI indicates that the RI input has changed from a low to a high. When TERI is set and the modem status interrupt is enabled, a modem status interrupt is generated.
1	DDSR	0	Change in DSR indicator bit. DDSR indicates that the DSR input has changed state since the last time it was read by the CPU. When DDSR is set and the modem status interrupt is enabled, a modem status interrupt is generated.
0	DCTS	0	Change in CTS indicator bit. DCTS indicates that the CTS input has changed state since the last time it was read by the CPU. When DCTS is set (autoflow control is not enabled and the modem status interrupt is enabled), a modem status interrupt is generated. When autoflow control is enabled, no interrupt is generated.

4.5.5.10 Scratch Pad Register (SCR)

The Scratch Pad register (SCR) is shown in Figure 4-250 and described in Table 4-242. SCR is intended for programmer's use as a scratch pad. It temporarily holds the programmer's data without affecting UART operation.

Figure 4-250. Scratch Pad Register (SCR)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		SCR	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-242. Scratch Pad Register (MSR) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	SCR	0	These bits are intended for the programmer's use as a scratch pad in the sense that it temporarily holds the programmer's data without affecting any other UART operation.

4.5.5.11 Divisor Latches (DLL and DLH)

Two 8-bit register fields (DLL and DLH), called divisor latches, store the 16-bit divisor for generation of the baud clock in the baud generator. The latches are in DLH and DLL. DLH holds the most-significant bits of the divisor, and DLL holds the least-significant bits of the divisor. These divisor latches must be loaded during initialization of the UART in order to ensure desired operation of the baud generator. Writing to the divisor latches results in two wait states being inserted during the write access while the baud generator is loaded with the new value.

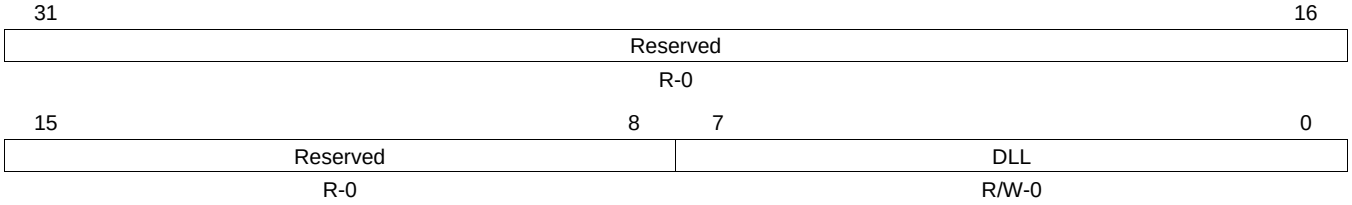
Access considerations:

- RBR, THR, and DLL share one address. When DLAB = 1 in LCR, all accesses at the shared address are accesses to DLL. When DLAB = 0, reading from the shared address gives the content of RBR, and writing to the shared address modifies THR.
- IER and DLH share one address. When DLAB = 1 in LCR, accesses to the shared address read or modify to DLH. When DLAB = 0, all accesses at the shared address read or modify IER.

DLL and DLH also have dedicated addresses. If you use the dedicated addresses, you can keep the DLAB bit cleared, so that RBR, THR, and IER are always selected at the shared addresses.

The divisor LSB latch (DLL) is shown in Figure 4-251 and described in Table 4-243. The divisor MSB latch (DLH) is shown in Figure 4-252 and described in Table 4-244.

Figure 4-251. Divisor LSB Latch (DLL)

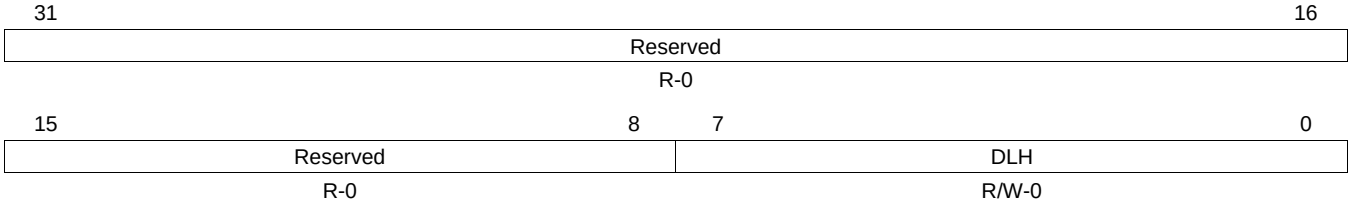


LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-243. Divisor LSB Latch (DLL) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLL	0-Fh	The 8 least-significant bits (LSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

Figure 4-252. Divisor MSB Latch (DLH)



LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-244. Divisor MSB Latch (DLH) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	DLH	0-Fh	The 8 most-significant bits (MSBs) of the 16-bit divisor for generation of the baud clock in the baud rate generator.

4.5.5.12 Revision Identification Registers (REVID1 and REVID2)

The revision identification registers (REVID1 and REVID2) contain peripheral identification data for the peripheral. REVID1 is shown in Figure 4-253 and described in Table 4-245. REVID2 is shown in Figure 4-254 and described in Table 4-246.

Figure 4-253. Revision Identification Register 1 (REVID1)

31	0
REVID1	
R-1102 0002h	

LEGEND: R = Read only; -*n* = value after reset

Table 4-245. Revision Identification Register 1 (REVID1) Field Descriptions

Bit	Field	Value	Description
31-0	REVID1	1102 0002h	Peripheral Identification Number

Figure 4-254. Revision Identification Register 2 (REVID2)

31	16
Reserved	
R-0	

15	8	7	0
Reserved		REVID2	
R-0		R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 4-246. Revision Identification Register 2 (REVID2) Field Descriptions

Bit	Field	Value	Description
31-8	Reserved	0	Reserved
7-0	REVID2	0	Peripheral Identification Number

4.5.5.13 Power and Emulation Management Register (PWREMU_MGMT)

The power and emulation management register (PWREMU_MGMT) is shown in Figure 4-255 and described in Table 4-247.

Figure 4-255. Power and Emulation Management Register (PWREMU_MGMT)

31	16
Reserved	
R-0	

15	14	13	12	1	0
Rsvd	UTRST	URRST	Reserved		FREE
R/W-0	R/W-0	R/W-0	R-1		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-247. Power and Emulation Management Register (PWREMU_MGMT) Field Descriptions

Bit	Field	Value	Description
31-16	Reserved	0	Reserved
15	Reserved	0	Reserved. This bit must always be written with a 0.
14	UTRST		UART transmitter reset. Resets and enables the transmitter.
		0	Transmitter is disabled and in reset state.
		1	Transmitter is enabled.
13	URRST		UART receiver reset. Resets and enables the receiver.
		0	Receiver is disabled and in reset state.
		1	Receiver is enabled.
12-1	Reserved	1	Reserved
0	FREE		Free-running enable mode bit. This bit determines the emulation mode functionality of the UART. When halted, the UART can handle register read/write requests, but does not generate any transmission/reception, interrupts or events.
		0	If a transmission is not in progress, the UART halts immediately. If a transmission is in progress, the UART halts after completion of the one-word transmission.
		1	Free-running mode is enabled; UART continues to run normally.

4.5.5.14 Mode Definition Register (MDR)

The Mode Definition register (MDR) determines the over-sampling mode for the UART. MDR is shown in Figure 4-256 and described in Table 4-248.

Figure 4-256. Mode Definition Register (MDR)

31	16
Reserved	
R-0	

15	1	0
Reserved		OSM_SEL
R-0		R/W-0

LEGEND: R/W = Read/Write; R = Read only; -*n* = value after reset

Table 4-248. Mode Definition Register (MDR) Field Descriptions

Bit	Field	Value	Description
31-1	Reserved	0	Reserved
0	OSM_SEL		Over-Sampling Mode Select.
		0	16× over-sampling.
		1	13× over-sampling.

4.5.6 PRU_ICSS_ECAP Registers

The PRU ECAP module within the PRU-ICSS is identical to the ECAP module in the AM335x PWMSS.

For additional details about the ECAP registers, see Section 15.3.

4.5.7 PRU_ICSS_MII_RT Registers

Table 4-249 lists the memory-mapped registers for the PRU_ICSS_MII_RT. All register offset addresses not listed in Table 4-249 should be considered as reserved locations and the register contents should not be modified.

Table 4-249. PRU_ICSS_MII_RT Registers

Offset	Acronym	Register Name	Section
0h	RXCFG0		Section 4.5.7.1
4h	RXCFG1		Section 4.5.7.2
10h	TXCFG0		Section 4.5.7.3
14h	TXCFG1		Section 4.5.7.4
20h	TXCRC0		Section 4.5.7.5
24h	TXCRC1		Section 4.5.7.6
30h	TXIPG0		Section 4.5.7.7
34h	TXIPG1		Section 4.5.7.8
38h	PRS0		Section 4.5.7.9
3Ch	PRS1		Section 4.5.7.10
40h	RXFRMS0		Section 4.5.7.11
44h	RXFRMS1		Section 4.5.7.12
48h	RXPCNT0		Section 4.5.7.13
4Ch	RXPCNT1		Section 4.5.7.14
50h	RXERR0		Section 4.5.7.15
54h	RXERR1		Section 4.5.7.16

4.5.7.1 RXCFG0 Register (Offset = 0h) [reset = 0h]

RXCFG0 is shown in Figure 4-257 and described in Table 4-250.

Return to Summary Table.

RX CONFIG0

Figure 4-257. RXCFG0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RX_AUTOFWD_PRE	RX_BYTE_SWAP	RX_L2_ENABLE	RX_MUX_SEL	RX_CUT_PREAMBLE	RESERVED	RX_ENABLE
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h

Table 4-250. RXCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	
6	RX_AUTOFWD_PRE	R/W	0h	Enables auto-forward of received preamble. When enabled, this will forward the preamble nibbles including the SFD to the TX L1 FIFO that is attached to the PRU. First data byte seen by PRU R31 and/or RX L2 is destination address (DA). Odd number of preamble nibbles is supported in this mode. For example, 0x55D. Note that new RX should only occur after the current TX completes. 0x 0: Disable 0x 1: Enable, it must disable RX_CUT_PREAMBLE and TX_AUTO_PREAMBLE
5	RX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for RX R31 and RX L2. Note that if TX_AUTO_SEQUENCE enabled, this bit cannot get enable since TX_BYTE_SWAP on swaps the PRU output. This bit must be selected/updated when the port is disabled or there is no traffic. 0x 0: R31 [15:8]/RXL2 [15:8] = Byte1{Nibble3,Nibble2} R31[7:0]/RXL2 [7:0] = Byte0{Nibble1,Nibble0} 0x 1: R31 [15:8]/RXL2 [15:8] = Byte0{Nibble1,Nibble0} R31[7:0]/RXL2 [7:0] = Byte1{Nibble3,Nibble2} Nibble0 is the first nibble received.
4	RX_L2_ENABLE	R/W	0h	Enables RX L2 buffer. 0x 0: Disable (RX L2 can function as generic scratch pad) 0x 1: Enable

Table 4-250. RXCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RX_MUX_SEL	R/W	0h	Selects receive data source. Typically, the setting for this will not be identical for the two MII receive configuration registers. 0x 0: MII RX Data from Port 0 (default for RXCFG0) 0x 1: MII RX Data from Port 1 (default for RXCFG1)
2	RX_CUT_PREAMBLE	R/W	0h	Removes received preamble. 0x 0: All data from Ethernet PHY are passed on to PRU register. This assumes Ethernet PHY which does not shorten the preamble. 0x 1: MII interface suppresses preamble and sync frame delimiter. First data byte seen by PRU register is destination address.
1	RESERVED	R	0h	
0	RX_ENABLE	R/W	0h	Enables the receive traffic currently selected by RX_MUX_SELECT. 0x 0: Disable 0x 1: Enable

4.5.7.2 RXCFG1 Register (Offset = 4h) [reset = 8h]

RXCFG1 is shown in Figure 4-258 and described in Table 4-251.

Return to Summary Table.

RX CONFIG1

Figure 4-258. RXCFG1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RX_AUTOFWD_PRE	RX_BYTE_SWAP	RX_L2_ENABLE	RX_MUX_SEL	RX_CUT_PREAMBLE	RESERVED	RX_ENABLE
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R-0h	R/W-0h

Table 4-251. RXCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	
6	RX_AUTOFWD_PRE	R/W	0h	Enables auto-forward of received preamble. When enabled, this will forward the preamble nibbles including the SFD to the TX L1 FIFO that is attached to the PRU. First data byte seen by PRU R31 and/or RX L2 is destination address (DA). Odd number of preamble nibbles is supported in this mode. For example, 0x55D. Note that new RX should only occur after the current TX completes. 0x 0: Disable 0x 1: Enable, it must disable RX_CUT_PREAMBLE and TX_AUTO_PREAMBLE
5	RX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for RX R31 and RX L2. Note that if TX_AUTO_SEQUENCE enabled, this bit cannot get enable since TX_BYTE_SWAP on swaps the PRU output. This bit must be selected/updated when the port is disabled or there is no traffic. 0x 0: R31 [15:8]/RXL2 [15:8] = Byte1{Nibble3,Nibble2} R31[7:0]/RXL2 [7:0] = Byte0{Nibble1,Nibble0} 0x 1: R31 [15:8]/RXL2 [15:8] = Byte0{Nibble1,Nibble0} R31[7:0]/RXL2 [7:0] = Byte1{Nibble3,Nibble2} Nibble0 is the first nibble received.
4	RX_L2_ENABLE	R/W	0h	Enables RX L2 buffer. 0x 0: Disable (RX L2 can function as generic scratch pad) 0x 1: Enable

Table 4-251. RXCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	RX_MUX_SEL	R/W	1h	Selects receive data source. Typically, the setting for this will not be identical for the two MII receive configuration registers. 0x 0: MII RX Data from Port 0 (default for RXCFG0) 0x 1: MII RX Data from Port 1 (default for RXCFG1)
2	RX_CUT_PREAMBLE	R/W	0h	Removes received preamble. 0x 0: All data from Ethernet PHY are passed on to PRU register. This assumes Ethernet PHY which does not shorten the preamble. 0x 1: MII interface suppresses preamble and sync frame delimiter. First data byte seen by PRU register is destination address.
1	RESERVED	R	0h	
0	RX_ENABLE	R/W	0h	Enables the receive traffic currently selected by RX_MUX_SELECT. 0x 0: Disable 0x 1: Enable

4.5.7.3 TXCFG0 Register (Offset = 10h) [reset = 00020010h]

TXCFG0 is shown in Figure 4-259 and described in Table 4-252.

Return to Summary Table.

TX CONFIG0

Figure 4-259. TXCFG0 Register

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

Table 4-252. TXCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	
30-28	TX_CLK_DELAY	R/W	0h	To ensure the MII_RT IO timing values published in the device data manual, the ocp_clk must be configured for 200 MHz and TX_CLK_DELAY must be set to 6h.
27-26	RESERVED	R	0h	
25-16	TX_START_DELAY	R/W	2h	Defines the minimum time interval (delay) between receiving the RXDV for the current frame and the start of the transmit interface sending data to the MII interface. Delay value is in units of MII_RT clock cycles, which uses the ocp_clk (default is 200MHz, or 5ns). Default TX_START_DELAY value is 320ns, which is optimized for minimum latency at 16 bit processing. Counter is started with RX_DV signal going active. Transmit interface stops sending data when no more data is written into transmit interface by PRU along with TX_EOF marker bit set. If the TX FIFO has data when the delay expires, then TX will start sending data. But if the TX FIFO is empty, it will not start until the TX FIFO is not empty. It is possible to overflow the TX FIFO with the max delay setting when auto-forwarding is enabled since the time delay is larger than the amount of data it needs to store. As long as TX L1 FIFO overflows, software will need to issue a TX_RESET to reset the TX FIFO. The total delay is 64-byte times (size of TX FIFO), but you need to allow delays for synchronization. Do to this fact, the maximum delay should be 80ns less when auto forwarding is enabled. Therefore, 0x3F0 is the maximum in this configuration.
15-10	RESERVED	R	0h	

Table 4-252. TXCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	TX_AUTO_SEQUENCE	R/W	0h	Enables transmit auto-sequence. Note the transmit data source is determined by TX_MUX_SEL setting. 0x 0: Disable 0x 1: Enable, transmit state machine based on events on receiver path that is connected to the respective transmitter. Also, the masking logic is disabled and only the MII data is used.
8	TX_MUX_SEL	R/W	0h	Selects transmit data source. The default/reset setting for TX Port 0 is 1. This setting permits MII TX Port 0 to receive data from PRU1 and the MII TX Port 1 which is connected to PRU0 by default. 0x 0: Data from PRU0 (default for TXCFG1) 0x 1: Data from PRU1 (default for TXCFG0)
7-4	RESERVED	R	1h	
3	TX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for TX R30. This bit must be selected/updated when the port is disabled or there is no traffic. 0x 0: R30 [15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30 [31:24] = TX_MASK [15:8] R30 [23:16] = TX_MASK [7:0] 0x 1: R30 [15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30 [31:24] = TX_MASK [7:0] R30 [23:16] = TX_MASK [15:8] Nibble0 is the first nibble received.
2	TX_EN_MODE	R/W	0h	Enables transmit self clear on TX_EOF event. Note that iep.cmp[3] must be set before transmission will start for TX0, and iep_cmp[4] for TX1. This is a new dependency, in addition to TX L1 FIFO not empty and TX_START_DELAY expiration, to start transmission. 0x 0: Disable 0x 1: Enable, TX_ENABLE will be clear for a TX_EOF event by itself.
1	TX_AUTO_PREAMBLE	R/W	0h	Transmit data auto-preamble. 0x 0: PRU will provide full preamble 0x 1: TX FIFO will insert pre-amble automatically. Note that the TX FIFO does not get preloaded with the preamble until the first write occurs. This can cause the latency to be larger than the minimum latency.
0	TX_ENABLE	R/W	0h	Enables transmit traffic on TX PORT. If TX_EN_MODE is set, then TX_ENABLE will self clear during a TX_EOF event. Note Software can use this to pre-fill the TX FIFO and then start the TX frame during non-ECS operations. 0x 0: TX PORT is disabled/stopped immediately 0x 1: TX PORT is enabled and the frame will start once the IPG counter expired and TX Start Delay counter has expired

4.5.7.4 TXCFG1 Register (Offset = 14h) [reset = 00400000h]

TXCFG1 is shown in Figure 4-260 and described in Table 4-253.

Return to Summary Table.

TX CONFIG1

Figure 4-260. TXCFG1 Register

31	30	29	28	27	26	25	24
RESERVED	TX_CLK_DELAY			RESERVED		TX_START_DELAY	
R-0h	R/W-0h			R-0h		R/W-40h	
23	22	21	20	19	18	17	16
TX_START_DELAY							
R/W-40h							
15	14	13	12	11	10	9	8
RESERVED						TX_AUTO_SEQUENCE	TX_MUX_SEL
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				TX_BYTE_SWAP	TX_EN_MODE	TX_AUTO_PREAMBLE	TX_ENABLE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 4-253. TXCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	
30-28	TX_CLK_DELAY	R/W	0h	To ensure the MII_RT IO timing values published in the device data manual, the ocp_clk must be configured for 200 MHz and TX_CLK_DELAY must be set to 6h.
27-26	RESERVED	R	0h	
25-16	TX_START_DELAY	R/W	40h	Defines the minimum time interval (delay) between receiving the RXDV for the current frame and the start of the transmit interface sending data to the MII interface. Delay value is in units of MII_RT clock cycles, which uses the ocp_clk (default is 200MHz, or 5ns). Default TX_START_DELAY value is 320ns, which is optimized for minimum latency at 16 bit processing. Counter is started with RX_DV signal going active. Transmit interface stops sending data when no more data is written into transmit interface by PRU along with TX_EOF marker bit set. If the TX FIFO has data when the delay expires, then TX will start sending data. But if the TX FIFO is empty, it will not start until the TX FIFO is not empty. It is possible to overflow the TX FIFO with the max delay setting when auto-forwarding is enabled since the time delay is larger than the amount of data it needs to store. As long as TX L1 FIFO overflows, software will need to issue a TX_RESET to reset the TX FIFO. The total delay is 64-byte times (size of TX FIFO), but you need to allow delays for synchronization. Do to this fact, the maximum delay should be 80ns less when auto forwarding is enabled. Therefore, 0x3F0 is the maximum in this configuration.
15-10	RESERVED	R	0h	

Table 4-253. TXCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	TX_AUTO_SEQUENCE	R/W	0h	Enables transmit auto-sequence. Note the transmit data source is determined by TX_MUX_SEL setting. 0x 0: Disable 0x 1: Enable, transmit state machine based on events on receiver path that is connected to the respective transmitter. Also, the masking logic is disabled and only the MII data is used.
8	TX_MUX_SEL	R/W	0h	Selects transmit data source. The default/reset setting for TX Port 0 is 1. This setting permits MII TX Port 0 to receive data from PRU1 and the MII TX Port 1 which is connected to PRU0 by default. 0x 0: Data from PRU0 (default for TXCFG1) 0x 1: Data from PRU1 (default for TXCFG0)
7-4	RESERVED	R	0h	
3	TX_BYTE_SWAP	R/W	0h	Defines the order of Byte0/1 placement for TX R30. This bit must be selected/updated when the port is disabled or there is no traffic. 0x 0: R30 [15:8] = Byte1{Nibble3,Nibble2} R30[7:0] = Byte0{Nibble1,Nibble0} R30 [31:24] = TX_MASK [15:8] R30 [23:16] = TX_MASK [7:0] 0x 1: R30 [15:8] = Byte0{Nibble1,Nibble0} R30[7:0] = Byte1{Nibble3,Nibble2} R30 [31:24] = TX_MASK [7:0] R30 [23:16] = TX_MASK [15:8] Nibble0 is the first nibble received.
2	TX_EN_MODE	R/W	0h	Enables transmit self clear on TX_EOF event. Note that iep.cmp[3] must be set before transmission will start for TX0, and iep_cmp[4] for TX1. This is a new dependency, in addition to TX L1 FIFO not empty and TX_START_DELAY expiration, to start transmission. 0x 0: Disable 0x 1: Enable, TX_ENABLE will be clear for a TX_EOF event by itself.
1	TX_AUTO_PREAMBLE	R/W	0h	Transmit data auto-preamble. 0x 0: PRU will provide full preamble 0x 1: TX FIFO will insert pre-amble automatically. Note that the TX FIFO does not get preloaded with the preamble until the first write occurs. This can cause the latency to be larger than the minimum latency.
0	TX_ENABLE	R/W	0h	Enables transmit traffic on TX PORT. If TX_EN_MODE is set, then TX_ENABLE will self clear during a TX_EOF event. Note Software can use this to pre-fill the TX FIFO and then start the TX frame during non-ECS operations. 0x 0: TX PORT is disabled/stopped immediately 0x 1: TX PORT is enabled and the frame will start once the IPG counter expired and TX Start Delay counter has expired

4.5.7.5 TXCRC0 Register (Offset = 20h) [reset = 0h]

TXCRC0 is shown in Figure 4-261 and described in Table 4-254.

Return to Summary Table.

TX CYCLIC REDUNDANCY CHECK0

Figure 4-261. TXCRC0 Register

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
TX_CRC32																															
R-0h																															

Table 4-254. TXCRC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX_CRC32	R	0h	FCS (CRC32) data can be read by PRU for diagnostics. It is only valid after 6 clocks after a TX_CRC_HIGH command is given.

4.5.7.6 TXCRC1 Register (Offset = 24h) [reset = 0h]

TXCRC1 is shown in Figure 4-262 and described in Table 4-255.

Return to Summary Table.

TX CYCLIC REDUNDANCY CHECK1

Figure 4-262. TXCRC1 Register

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
TX_CRC32																															
R-0h																															

Table 4-255. TXCRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TX_CRC32	R	0h	FCS (CRC32) data can be read by PRU for diagnostics. It is only valid after 6 clocks after a TX_CRC_HIGH command is given.

4.5.7.7 TXIPG0 Register (Offset = 30h) [reset = 28h]

TXIPG0 is shown in Figure 4-263 and described in Table 4-256.

Return to Summary Table.

TX INTERPACKET GAP0

Figure 4-263. TXIPG0 Register

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
RESERVED																						TX_IPG									
R-0h																						R/W-28h									

Table 4-256. TXIPG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9-0	TX_IPG	R/W	28h	Defines the minimum of transmit Inter Packet Gap (IPG) which is the number of ocp_clk cycles between the de-assertion of TX_EN and the assertion of TX_EN. The start of the TX will get delayed when the incoming packet IPG is less than defined minimum value. In general, software should program in increments of 8, 40ns to insure the extra delays takes effect.

4.5.7.8 TXIPG1 Register (Offset = 34h) [reset = 28h]

TXIPG1 is shown in Figure 4-264 and described in Table 4-257.

Return to Summary Table.

TX INTERPACKET GAP1

Figure 4-264. TXIPG1 Register

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
RESERVED																						TX_IPG									
R-0h																						R/W-28h									

Table 4-257. TXIPG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	
9-0	TX_IPG	R/W	28h	Defines the minimum of transmit Inter Packet Gap (IPG) which is the number of ocp_clk cycles between the de-assertion of TX_EN and the assertion of TX_EN. The start of the TX will get delayed when the incoming packet IPG is less than defined minimum value. In general, software should program in increments of 8, 40ns to insure the extra delays takes effect.

4.5.7.9 PRS0 Register (Offset = 38h) [reset = 0h]

PRS0 is shown in Figure 4-265 and described in Table 4-258.

Return to Summary Table.

PORT RAW STATUS0

Figure 4-265. PRS0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						pr1_mii0_crs	pr1_mii0_col
R-0h						R-0h	R-0h

Table 4-258. PRS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	pr1_mii0_crs	R	0h	Current state of pr1_mii0_crs
0	pr1_mii0_col	R	0h	Current state of pr1_mii0_col

4.5.7.10 PRS1 Register (Offset = 3Ch) [reset = 0h]

PRS1 is shown in Figure 4-266 and described in Table 4-259.

Return to Summary Table.

PORT RAW STATUS1

Figure 4-266. PRS1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						pr1_mii1_crs	pr1_mii1_col
R-0h						R-0h	R-0h

Table 4-259. PRS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	
1	pr1_mii1_crs	R	0h	Current state of pr1_mii1_crs
0	pr1_mii1_col	R	0h	Current state of pr1_mii1_col

4.5.7.11 RXFRMS0 Register (Offset = 40h) [reset = 9E1h]

RXFRMS0 is shown in Figure 4-267 and described in Table 4-260.

Return to Summary Table.

RX FRAME SIZE0

Figure 4-267. RXFRMS0 Register

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
RX_MAX_FRM_CNT																RX_MIN_FRM_CNT															
R/W-0h																R/W-9E1h															

Table 4-260. RXFRMS0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MAX_FRM_CNT	R/W	5F1h	Defines the maximum received frame count. If the total byte count of the received frame is more than defined value, RX_MAX_FRM_ERR will get set. 0x 0: 1 byte after SFD and including CRC. N: N+1 bytes after SFD and including CRC Note if the incoming frame is truncated at the marker, RX_CRC and RX_NIBBLE_ODD will not get asserted.
15-0	RX_MIN_FRM_CNT	R/W	3Fh	Defines the minimum received frame count. If the total byte count of received frame is less than defined value, RX_MIN_FRM_ERR will get set. 0x 0: 1 byte after SFD and including CRC. N: N+1 bytes after SFD and including CRC

4.5.7.12 RXFRMS1 Register (Offset = 44h) [reset = 05F1003Fh]

RXFRMS1 is shown in Figure 4-268 and described in Table 4-261.

Return to Summary Table.

RX FRAME SIZE1

Figure 4-268. RXFRMS1 Register

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
RX_MAX_FRM_CNT																RX_MIN_FRM_CNT															
R/W-5F1h																R/W-3Fh															

Table 4-261. RXFRMS1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RX_MAX_FRM_CNT	R/W	5F1h	Defines the maximum received frame count. If the total byte count of the received frame is more than defined value, RX_MAX_FRM_ERR will get set. 0x 0: 1 byte after SFD and including CRC. N: N+1 bytes after SFD and including CRC Note if the incoming frame is truncated at the marker, RX_CRC and RX_NIBBLE_ODD will not get asserted.
15-0	RX_MIN_FRM_CNT	R/W	3Fh	Defines the minimum received frame count. If the total byte count of received frame is less than defined value, RX_MIN_FRM_ERR will get set. 0x 0: 1 byte after SFD and including CRC. N: N+1 bytes after SFD and including CRC

4.5.7.13 RXPCNT0 Register (Offset = 48h) [reset = E1h]

RXPCNT0 is shown in Figure 4-269 and described in Table 4-262.

Return to Summary Table.

RX PREAMABLE COUNT0

Figure 4-269. RXPCNT0 Register

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

Table 4-262. RXPCNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-4	RX_MAX_PRE_CNT	R/W	Eh	Defines the maximum number of nibbles until the start of frame delimiter (SFD) event occurred (i.e. matches 0x5D). RX_MAX_PRE_COUNT_ERR will be set if the preamble counts more than the value of RX_MAX_PRE_CNT. If the SFD does not occur within 16 nibbles, the error will assert and the incoming frame will be truncated. 0x 0: Disabled 0x 1: Reserved 0x 2: 4th nibble needs to have built 0x5D 0xe: 16th nibble needs to have built 0x5D Note the 16th nibble is transmitted.
3-0	RX_MIN_PRE_CNT	R/W	1h	Defines the minimum number of nibbles until the start of frame delimiter (SFD) event occurred, which is matched the value 0x5D. RX_MIN_PRE_COUNT_ERR will be set if the preamble counts less than the value of RX_MIN_PRE_CNT. 0x 0: Disabled 0x 1: 1 0x5 before 0x5D 0x 2: 2 0x5 before 0x5D N: N 0x5 before 0x5D Note it does not need to be 0x5.

4.5.7.14 RXPCNT1 Register (Offset = 4Ch) [reset = E1h]

RXPCNT1 is shown in Figure 4-270 and described in Table 4-263.

Return to Summary Table.

RX PREAMABLE COUNT1

Figure 4-270. RXPCNT1 Register

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

Table 4-263. RXPCNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	
7-4	RX_MAX_PRE_CNT	R/W	Eh	Defines the maximum number of nibbles until the start of frame delimiter (SFD) event occurred (i.e. matches 0x5D). RX_MAX_PRE_COUNT_ERR will be set if the preamble counts more than the value of RX_MAX_PRE_CNT. If the SFD does not occur within 16 nibbles, the error will assert and the incoming frame will be truncated. 0x 0: Disabled 0x 1: Reserved 0x 2: 4th nibble needs to have built 0x5D 0xe: 16th nibble needs to have built 0x5D Note the 16th nibble is transmitted.
3-0	RX_MIN_PRE_CNT	R/W	1h	Defines the minimum number of nibbles until the start of frame delimiter (SFD) event occurred, which is matched the value 0x5D. RX_MIN_PRE_COUNT_ERR will be set if the preamble counts less than the value of RX_MIN_PRE_CNT. 0x 0: Disabled 0x 1: 1 0x5 before 0x5D 0x 2: 2 0x5 before 0x5D N: N 0x5 before 0x5D Note it does not need to be 0x5.

4.5.7.15 RXERR0 Register (Offset = 50h) [reset = 0h]

RXERR0 is shown in Figure 4-271 and described in Table 4-264.

Return to Summary Table.

RX ERROR0

Figure 4-271. RXERR0 Register

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

Table 4-264. RXERR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	RX_MAX_FRM_CNT_ERR	R	0h	Error status of received frame is more than the value of RX_MAX_FRM_CNT. 0x 0: No error occurred 0x 1: Error occurred
2	RX_MIN_FRM_CNT_ERR	R	0h	Error status of received frame is less than the value of RX_MIN_FRM_CNT. 0x 0: No error occurred 0x 1: Error occurred
1	RX_MAX_PRE_CNT_ERR	R	0h	Error status of received preamble nibble is more than the value of RX_MAX_PRE_CNT. 0x 0: No error occurred 0x 1: Error occurred
0	RX_MIN_PRE_CNT_ERR	R	0h	Error status of received preamble nibble is less than the value of RX_MIN_PRE_CNT. 0x 0: No error occurred 0x 1: Error occurred

4.5.7.16 RXERR1 Register (Offset = 54h) [reset = 0h]

RXERR1 is shown in Figure 4-272 and described in Table 4-265.

Return to Summary Table.

RX ERROR1

Figure 4-272. RXERR1 Register

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

Table 4-265. RXERR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	
3	RX_MAX_FRM_CNT_ERR	R	0h	Error status of received frame is more than the value of RX_MAX_FRM_CNT. 0x 0: No error occurred 0x 1: Error occurred
2	RX_MIN_FRM_CNT_ERR	R	0h	Error status of received frame is less than the value of RX_MIN_FRM_CNT. 0x 0: No error occurred 0x 1: Error occurred
1	RX_MAX_PRE_CNT_ERR	R	0h	Error status of received preamble nibble is more than the value of RX_MAX_PRE_CNT. 0x 0: No error occurred 0x 1: Error occurred
0	RX_MIN_PRE_CNT_ERR	R	0h	Error status of received preamble nibble is less than the value of RX_MIN_PRE_CNT. 0x 0: No error occurred 0x 1: Error occurred

4.5.8 PRU_ICSS_MDIO Registers

For additional details about the MDIO registers, see Section 14.5.10, *MDIO Registers*.

4.5.9 PRU_ICSS_CFG Registers

Table 4-266 lists the memory-mapped registers for the PRU_ICSS_CFG. All register offset addresses not listed in Table 4-266 should be considered as reserved locations and the register contents should not be modified.

Table 4-266. PRU_ICSS_CFG Registers

Offset	Acronym	Register Name	Section
0h	REVID		Section 4.5.9.1
4h	SYSCFG		Section 4.5.9.2
8h	GPCFG0		Section 4.5.9.3
Ch	GPCFG1		Section 4.5.9.4
10h	CGR		Section 4.5.9.5
14h	ISRP		Section 4.5.9.6
18h	ISP		Section 4.5.9.7
1Ch	IESP		Section 4.5.9.8
20h	IECP		Section 4.5.9.9
28h	PMAO		Section 4.5.9.10
2Ch	MII_RT		Section 4.5.9.11
30h	IEPCLK		Section 4.5.9.12
34h	SPP		Section 4.5.9.13
40h	PIN_MX		Section 4.5.9.14

4.5.9.1 REVID Register (offset = 0h) [reset = 47000000h]

REVID is shown in Figure 4-273 and described in Table 4-267.

The Revision Register contains the ID and revision information.

Figure 4-273. REVID Register

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
REVID																															
R-47000000h																															

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-267. REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVID	R	47000000h	Revision ID

4.5.9.2 SYSCFG Register (offset = 4h) [reset = 1Ah]

SYSCFG is shown in Figure 4-274 and described in Table 4-268.

The System Configuration Register defines the power IDLE and STANDBY modes.

Figure 4-274. SYSCFG Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-268. SYSCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	
5	SUB_MWAIT	R	0h	Status bit for wait state. 0 = Ready for Transaction 1 = Wait until 0
4	STANDBY_INIT	R/W	1h	1 = Initiate standby sequence. 0 = Enable OCP master ports.
3-2	STANDBY_MODE	R/W	2h	0h = Force standby mode: Initiator unconditionally in standby (standby = 1). 1h = No standby mode: Initiator unconditionally out of standby (standby = 0). 2h = Smart standby mode: Standby requested by initiator depending on internal conditions. 3h = Reserved.
1-0	IDLE_MODE	R/W	2h	0h = Force-idle mode. 1h = No-idle mode. 2h = Smart-idle mode. 3h = Reserved.

4.5.9.3 GPCFG0 Register (offset = 8h) [reset = 0h]

GPCFG0 is shown in Figure 4-275 and described in Table 4-269.

The General Purpose Configuration 0 Register defines the GPI/O configuration for PRU0.

Figure 4-275. GPCFG0 Register

31	30	29	28	27	26	25	24
RESERVED						PRU0_GPO_SH_SEL	PRU0_GPO_DIV1
R/W-0h						R-0h	R/W-0h
23	22	21	20	19	18	17	16
PRU0_GPO_DIV1				PRU0_GPO_DIV0			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU0_GPO_DIV0	PRU0_GPO_MODE	PRU0_GPI_SB	PRU0_GPI_DIV1				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRU0_GPI_DIV0					PRU0_GPI_CLK_MODE	PRU0_GPI_MODE	
R/W-0h					R/W-0h	R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-269. GPCFG0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	
25	PRU0_GPO_SH_SEL	R/W	0h	Defines which shadow register is currently getting used for GPO shifting. 0 = gpo_sh0 is selected 1 = gpo_sh1 is selected
24-20	PRU0_GPO_DIV1	R/W	0h	Divisor value (divide by PRU0_GPO_DIV1 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
19-15	PRU0_GPO_DIV0	R/W	0h	Divisor value (divide by PRU0_GPO_DIV0 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
14	PRU0_GPO_MODE	R/W	0h	PRU GPO (R30) modes: 0 = Direct output mode 1 = Serial output mode
13	PRU0_GPI_SB	R/W	0h	Start Bit event for 28-bit shift in mode. PRU0_GPI_SB (pru0_r31_status[29]) is set when first capture of a 1 on pru0_r31_status[0]. Read 1: Start Bit event occurred. Read 0: Start Bit event has not occurred. Write 1: Will clear PRU0_GPI_SB and clear the whole shift register. Write 0: No Effect.
12-8	PRU0_GPI_DIV1	R/W	0h	Divisor value (divide by PRU0_GPI_DIV1 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.

Table 4-269. GPCFG0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-3	PRU0_GPI_DIV0	R/W	0h	Divisor value (divide by PRU0_GPI_DIV0 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
2	PRU0_GPI_CLK_MODE	R/W	0h	Parallel 16-bit capture mode clock edge. 0 = Use the positive edge of pru0_r31_status[16] 1 = Use the negative edge of pru0_r31_status[16]
1-0	PRU0_GPI_MODE	R/W	0h	PRU GPI (R31) modes: 0h = Direct input mode. 1h = 16bit parallel capture mode. 2h = 28bit shift in mode. 3h = Mii_rt mode

4.5.9.4 GPCFG1 Register (offset = Ch) [reset = 0h]

GPCFG1 is shown in Figure 4-276 and described in Table 4-270.

The General Purpose Configuration 1 Register defines the GPI/O configuration for PRU1.

Figure 4-276. GPCFG1 Register

31	30	29	28	27	26	25	24
RESERVED						PRU1_GPO_SH_SEL	PRU1_GPO_DIV1
R/W-0h						R-0h	R/W-0h
23	22	21	20	19	18	17	16
PRU1_GPO_DIV1				PRU1_GPO_DIV0			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU1_GPO_DIV0	PRU1_GPO_MODE	PRU1_GPI_SB	PRU1_GPI_DIV1				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
7	6	5	4	3	2	1	0
PRU1_GPI_DIV0					PRU1_GPI_CLK_MODE	PRU1_GPI_MODE	
R/W-0h					R/W-0h	R/W-0h	

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-270. GPCFG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R/W	0h	
25	PRU1_GPO_SH_SEL	R/W	0h	Defines which shadow register is currently getting used for GPO shifting. 0 = gpo_sh0 is selected 1 = gpo_sh1 is selected
24-20	PRU1_GPO_DIV1	R/W	0h	Divisor value (divide by PRU1_GPO_DIV1 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
19-15	PRU1_GPO_DIV0	R/W	0h	Divisor value (divide by PRU1_GPO_DIV0 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
14	PRU1_GPO_MODE	R/W	0h	PRU GPO (R30) modes: 0 = Direct output mode 1 = Serial output mode
13	PRU1_GPI_SB	R/W	0h	28-bit shift in mode Start Bit event. PRU1_GPI_SB (pru1_r31_status[29]) is set when first capture of a 1 on pru1_r31_status[0]. Read 1: Start Bit event occurred. Read 0: Start Bit event has not occurred. Write 1: Will clear PRU1_GPI_SB and clear the whole shift register. Write 0: No Effect.
12-8	PRU1_GPI_DIV1	R/W	0h	Divisor value (divide by PRU1_GPI_DIV1 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.

Table 4-270. GPCFG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-3	PRU1_GPI_DIV0	R/W	0h	Divisor value (divide by PRU1_GPI_DIV0 + 1). 0h = div 1.0. 1h = div 1.5. 2h = div 2.0. .. 1eh = div 16.0. 1fh = reserved.
2	PRU1_GPI_CLK_MODE	R/W	0h	Parallel 16-bit capture mode clock edge. 0 = Use the positive edge of pru1_r31_status[16] 1 = Use the negative edge of pru1_r31_status[16]
1-0	PRU1_GPI_MODE	R/W	0h	PRU GPI (R31) modes: 0h = Direct input mode. 1h = 16bit parallel capture mode. 2h = 28bit shift in mode. 3h = Mii_rt mode

4.5.9.5 CGR Register (offset = 10h) [reset = 24924h]

CGR is shown in Figure 4-277 and described in Table 4-271.

The Clock Gating Register controls the state of Clock Management of the different modules. Software should not clear {module}_CLK_EN until {module}_CLK_STOP_ACK is 0x1.

Figure 4-277. CGR Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED						IEP_CLK_EN	IEP_CLK_STOP_ACK
R/W-0h						R/W-1h	R-0h
15	14	13	12	11	10	9	8
IEP_CLK_STOP_REQ	ECAP_CLK_EN	ECAP_CLK_STOP_ACK	ECAP_CLK_STOP_REQ	UART_CLK_EN	UART_CLK_STOP_ACK	UART_CLK_STOP_REQ	INTC_CLK_EN
R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h
7	6	5	4	3	2	1	0
INTC_CLK_STOP_ACK	INTC_CLK_STOP_REQ	PRU1_CLK_EN	PRU1_CLK_STOP_ACK	PRU1_CLK_STOP_REQ	PRU0_CLK_EN	PRU0_CLK_STOP_ACK	PRU0_CLK_STOP_REQ
R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h	R/W-1h	R-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-271. CGR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R/W	0h	
17	IEP_CLK_EN	R/W	1h	IEP clock enable. 0 = Disable Clock 1 = Enable Clock
16	IEP_CLK_STOP_ACK	R	0h	Acknowledgement that IEP clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
15	IEP_CLK_STOP_REQ	R/W	0h	IEP request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock
14	ECAP_CLK_EN	R/W	1h	ECAP clock enable. 0 = Disable Clock 1 = Enable Clock
13	ECAP_CLK_STOP_ACK	R	0h	Acknowledgement that ECAP clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
12	ECAP_CLK_STOP_REQ	R/W	0h	ECAP request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock
11	UART_CLK_EN	R/W	1h	UART clock enable. 0 = Disable Clock 1 = Enable Clock
10	UART_CLK_STOP_ACK	R	0h	Acknowledgement that UART clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
9	UART_CLK_STOP_REQ	R/W	0h	UART request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock
8	INTC_CLK_EN	R/W	1h	INTC clock enable. 0 = Disable Clock 1 = Enable Clock

Table 4-271. CGR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	INTC_CLK_STOP_ACK	R	0h	Acknowledgement that INTC clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
6	INTC_CLK_STOP_REQ	R/W	0h	INTC request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock
5	PRU1_CLK_EN	R/W	1h	PRU1 clock enable. 0 = Disable Clock 1 = Enable Clock
4	PRU1_CLK_STOP_ACK	R	0h	Acknowledgement that PRU1 clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
3	PRU1_CLK_STOP_REQ	R/W	0h	PRU1 request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock
2	PRU0_CLK_EN	R/W	1h	PRU0 clock enable. 0 = Disable Clock 1 = Enable Clock
1	PRU0_CLK_STOP_ACK	R	0h	Acknowledgement that PRU0 clock can be stopped. 0 = Not Ready to Gate Clock 1 = Ready to Gate Clock
0	PRU0_CLK_STOP_REQ	R/W	0h	PRU0 request to stop clock. 0 = do not request to stop Clock 1 = request to stop Clock

4.5.9.6 ISRP Register (offset = 14h) [reset = 0h]

ISRP is shown in Figure 4-278 and described in Table 4-272.

The IRQ Status Raw Parity register is a snapshot of the IRQ raw status for the PRU ICSS memory parity events. The raw status is set even if the event is not enabled.

Figure 4-278. ISRP Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				RAM_PE_RAW			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU1_DMEM_PE_RAW				PRU1_IMEM_PE_RAW			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
PRU0_DMEM_PE_RAW				PRU0_IMEM_PE_RAW			
R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-272. ISRP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	
19-16	RAM_PE_RAW	R/W	0h	RAM Parity Error RAW for Byte3, Byte2, Byte1, Byte0. Note RAM_PE_RAW[0] maps to Byte0. Write 0: No action. Read 0: No event pending. Read 1: Event pending. Write 1: Set event (debug).
15-12	PRU1_DMEM_PE_RAW	R/W	0h	PRU1 DMEM Parity Error RAW for Byte3, Byte2, Byte1, Byte0. Note PRU1_DMEM_PE_RAW[0] maps to Byte0. Write 0: No action. Read 0: No event pending. Read 1: Event pending. Write 1: Set event (debug).
11-8	PRU1_IMEM_PE_RAW	R/W	0h	PRU1 IMEM Parity Error RAW for Byte3, Byte2, Byte1, Byte0. Note PRU1_IMEM_PE_RAW[0] maps to Byte0. Write 0: No action. Read 0: No event pending. Read 1: Event pending. Write 1: Set event (debug).
7-4	PRU0_DMEM_PE_RAW	R/W	0h	PRU0 DMEM Parity Error RAW for Byte3, Byte2, Byte1, Byte0. Note PRU0_DMEM_PE_RAW[0] maps to Byte0. Write 0: No action. Read 0: No event pending. Read 1: Event pending. Write 1: Set event (debug) .
3-0	PRU0_IMEM_PE_RAW	R/W	0h	PRU0 IMEM Parity Error RAW for Byte3, Byte2, Byte1, Byte0. Note PRU0_IRAM_PE_RAW[0] maps to Byte0. Write 0: No action. Read 0: No event pending. Read 1: Event pending. Write 1: Set event (debug) .

4.5.9.7 ISP Register (offset = 18h) [reset = 0h]

ISP is shown in Figure 4-279 and described in Table 4-273.

The IRQ Status Parity Register is a snapshot of the IRQ status for the PRU ICSS memory parity events. The status is set only if the event is enabled. Write 1 to clear the status after the interrupt has been serviced.

Figure 4-279. ISP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												RAM_PE			
R-0h												0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRU1_DMEM_PE				PRU1_IMEM_PE				PRU0_DMEM_PE				PRU0_IMEM_PE			
0h				0h				0h				0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-273. ISP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	
19-16	RAM_PE		0h	RAM Parity Error for Byte3, Byte2, Byte1, Byte0. Note RAM_PE[0] maps to Byte0. Write 0: No action. Read 0: No (enabled) event pending. Read 1: Event pending. Write 1: Clear event.
15-12	PRU1_DMEM_PE		0h	PRU1 DMEM Parity Error for Byte3, Byte2, Byte1, Byte0. Note PRU1_DMEM_PE[0] maps to Byte0. Write 0: No action. Read 0: No (enabled) event pending. Read 1: Event pending. Write 1: Clear event.
11-8	PRU1_IMEM_PE		0h	PRU1 IMEM Parity Error for Byte3, Byte2, Byte1, Byte0. Note PRU1_IMEM_PE[0] maps to Byte0. Write 0: No action. Read 0: No (enabled) event pending. Read 1: Event pending. Write 1: Clear event.
7-4	PRU0_DMEM_PE		0h	PRU0 DMEM Parity Error for Byte3, Byte2, Byte1, Byte0. Note PRU0_DMEM_PE[0] maps to Byte0. Write 0: No action. Read 0: No(enabled) event pending. Read 1: Event pending. Write 1: Clear event.
3-0	PRU0_IMEM_PE		0h	PRU0 IMEM Parity Error for Byte3, Byte2, Byte1, Byte0. Note PRU0_IMEM_PE[0] maps to Byte0. Write 0: No action. Read 0: No (enabled) event pending. Read 1: Event pending. Write 1: Clear event.

4.5.9.8 IESP Register (offset = 1Ch) [reset = 0h]

IESP is shown in Figure 4-280 and described in Table 4-274.

The IRQ Enable Set Parity Register enables the IRQ PRU ICSS memory parity events.

Figure 4-280. IESP Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED				RAM_PE_SET			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
PRU1_DMEM_PE_SET				PRU1_IMEM_PE_SET			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
PRU0_DMEM_PE_SET				PRU0_IMEM_PE_SET			
R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-274. IESP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	
19-16	RAM_PE_SET	R/W	0h	RAM Parity Error Set Enable for Byte3, Byte2, Byte1, Byte0. Note RAM_PE_SET[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Enable interrupt.
15-12	PRU1_DMEM_PE_SET	R/W	0h	PRU1 DMEM Parity Error Set Enable for Byte3, Byte2, Byte1, Byte0. Note PRU1_DMEM_PE_SET[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Enable interrupt.
11-8	PRU1_IMEM_PE_SET	R/W	0h	PRU1 IMEM Parity Error Set Enable for Byte3, Byte2, Byte1, Byte0. Note PRU1_IMEM_PE_SET[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Enable interrupt.
7-4	PRU0_DMEM_PE_SET	R/W	0h	PRU0 DMEM Parity Error Set Enable for Byte3, Byte2, Byte1, Byte0. Note PRU0_DMEM_PE_SET[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Enable interrupt.
3-0	PRU0_IMEM_PE_SET	R/W	0h	PRU0 IMEM Parity Error Set Enable for Byte3, Byte2, Byte1, Byte0. Note PRU0_IMEM_PE_SET[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Enable interrupt.

4.5.9.9 IECP Register (offset = 20h) [reset = 0h]

IECP is shown in Figure 4-281 and described in Table 4-275.

The IRQ Enable Clear Parity Register disables the IRQ PRU ICSS memory parity events.

Figure 4-281. IECP Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
PRU1_DMEM_PE_CLR				PRU1_IMEM_PE_CLR			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
PRU0_DMEM_PE_CLR				PRU0_IMEM_PE_CLR			
R/W-0h				R/W-0h			

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-275. IECP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	
15-12	PRU1_DMEM_PE_CLR	R/W	0h	PRU1 DMEM Parity Error Clear Enable for Byte3, Byte2, Byte1, Byte0. Note PRU1_DMEM_PE_CLR[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Disable interrupt.
11-8	PRU1_IMEM_PE_CLR	R/W	0h	PRU1 IMEM Parity Error Clear Enable for Byte3, Byte2, Byte1, Byte0. Note PRU1_IMEM_PE_CLR[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Disable interrupt.
7-4	PRU0_DMEM_PE_CLR	R/W	0h	PRU0 DMEM Parity Error Clear Enable for Byte3, Byte2, Byte1, Byte0. Note PRU0_DMEM_PE_CLR[0] maps to Byte0. Write 0: No action. Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Disable interrupt.
3-0	PRU0_IMEM_PE_CLR	R/W	0h	PRU0 IMEM Parity Error Clear Enable for Byte3, Byte2, Byte1, Byte0. Note PRU0_IMEM_PE_CLR[0] maps to Byte0. Write 0: No action . Read 0: Interrupt disabled (masked). Read 1: Interrupt enabled. Write 1: Disable interrupt.

4.5.9.10 PMAO Register (offset = 28h) [reset = 0h]

PMAO is shown in Figure 4-282 and described in Table 4-276.

The PRU Master OCP Address Offset Register enables for the PRU OCP Master Port Address to have an offset of minus 0x0008_0000. This enables the PRU to access External Host address space starting at 0x0000_0000.

Figure 4-282. PMAO Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						PMAO_PRU1	PMAO_PRU0
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-276. PMAO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	PMAO_PRU1	R/W	0h	PRU1 OCP Master Port Address Offset Enable. 0 = Disable address offset. 1 = Enable address offset of -0x0008_0000.
0	PMAO_PRU0	R/W	0h	PRU0 OCP Master Port Address Offset Enable. 0 = Disable address offset. 1 = Enable address offset of -0x0008_0000.

4.5.9.11 MII_RT Register (Offset = 2Ch) [reset = 0h]

TXCRC1 is shown in Figure 4-283 and described in Table 4-277.

Return to Summary Table.

The MII_RT Event Enable Register enables MII_RT mode events to the INTC.

Figure 4-283. MII_RT Register

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

Table 4-277. MII_RT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	MII_RT_EVENT_EN	R/W	1h	Enables the MII_RT Events to the INTC. 0h = Disabled (use external events) 1h = Enabled (use MII_RT events)

4.5.9.12 IEPCLK Register (offset = 30h) [reset = 0h]

IEPCLK is shown in Figure 4-284 and described in Table 4-278.

The IEP Clock Source Register defines the source of the IEP clock.

Figure 4-284. IEPCLK Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-278. IEPCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R/W	0h	
0	OCP_EN	R/W	0h	Selects IEP clock source 0 = iep_clk is the source. 1 = ocp_clk is the source.

4.5.9.13 SPP Register (offset = 34h) [reset = 0h]

SPP is shown in Figure 4-285 and described in Table 4-279.

The Scratch Pad Priority and Configuration Register defines the access priority assigned to the PRU cores and configures the scratch pad XFR shift functionality.

Figure 4-285. SPP Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED						XFR_SHIFT_EN	PRU1_PAD_HP_EN
R/W-0h						R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-279. SPP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R/W	0h	
1	XFR_SHIFT_EN	R/W	0h	Enables XIN/XOUT shift functionality. When enabled, R0 [4:0] (internal to PRU) defines the 32-bit offset for XIN and XOUT operations with the scratch pad. 0 = Disabled. 1 = Enabled.
0	PRU1_PAD_HP_EN	R/W	0h	Defines which PRU wins write cycle arbitration to a common scratch pad bank. The PRU which has higher priority will always perform the write cycle with no wait states. The lower PRU will get stalled/wait states until higher PRU is not performing write cycles. If the lower priority PRU writes to the same byte has the higher priority PRU, then the lower priority PRU will over write the bytes. 0 = PRU0 has highest priority. 1 = PRU1 has highest priority.

4.5.9.14 PIN_MX Register (offset = 40h) [reset = 0h]

PIN_MX is shown in Figure 4-286 and described in Table 4-280.

The Pin Mux Select Register defines the state of the PRU ICSS internal pinmuxing.

Figure 4-286. PIN_MX Register

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

LEGEND: R/W = Read/Write; R = Read only; W1toCl = Write 1 to clear bit; *-n* = value after reset

Table 4-280. PIN_MX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R/W	0h	
7-0	PIN_MUX_SEL	R/W	0h	Defines the state of PIN_MUX_SEL [1:0] for internal pinmuxing.