Programmer's Reference Guide

BCM5725/BCM5762/BCM57767

NetXtreme®/NetLink® BCM5725/BCM5762/ BCM57767 Programmer's Guide

5725_5762_57767-PG101-R

5300 California Avenue • Irvine, CA 92617 • Phone: 949-926-5000 • Fax: 949-926-5203

June 27, 2013

Revision History

Revision	*Date*	*Change Description*
5725_5762_57767-PG101-R	06/27/13	**Updated:** • "DMA Read/Write Control Register (offset: 0x6C) — Function 0" on page 258 • "Device ID and Vendor ID Register (offset: 0x00) – Function 1" on page 291 • "PCI Classcode and Revision ID Register (offset: 0x8) – Function 1" on page 293 • "Subsystem ID/Vendor ID Register (offset: 0x2C) – Function 1" on page 298 • "EMAC Mode Register (offset: 0x400)" on page 321 • "EMAC Event Enable Register (offset: 0x408)" on page 323 • "Autopolling Status Register (offset: 0x458) " on page 329 • "Transmit MAC Mode Register (offset: 0x45C)" on page 330 • "Receive MAC Mode Register (offset: 0x468)" on page 332 • "Receive Rules Configuration Register (offset: 0x500)" on page 336 • "CPMU Control Register (offset: 0x3600)" on page 385 • "Link Aware Power Mode Clock Policy Register (offset: 0x3610)" on page 390 • "Link Idle Power Mode Clock Policy Register (offset: 0x3618)" on page 392 • "Clock Speed Override Policy Register (offset: 0x3624)" on page 394 • "Clock Status Register (offset: 0x3630)" on page 397 • "GPHY Control/Status Register (offset: 0x3638)" on page 398 • "RAM Control Register (offset: 0x363C)" on page 399 • CPMU Energy Detect Raw Debounce Control 2 Register (offset: 0x367c) updated to "CPMU TOP MISC. Control 1 Register (offset: 0x367C)" on page 415 • CPMU Device ID Status and Control Register (offset: 0x3690) updated to "CPMU LTR Control Register (offset: 0x3690)" on page 417 • "Miscellaneous Control Register (offset: 0x36AC)" on page 420 • "EEE Link Idle Control Register (offset: 0x36BC)" on page 424 • "EEE Link Idle Status Register (offset: 0x36C0)" on page 425 • "Card Reader Idle Control Register (offset: 0x36E0)" on page 427 • "Card Reader Clock Policy Register (offset: 0x36E4)" on page 430 • "Receive Coalescing Ticks Register (offset: 0x3C08)" on page 444 • "Send Coalescing Ticks Register (offset: 0x3C0C)" on page 445

Revision	*Date*	*Change Description*
		• "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446 • "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 446 • "Receive Max Coalesced BD Count During Interrupt (offset 0x3C18)" on page 447 • "Send Max Coalesced BD Count During Interrupt (offset 0x3C1C)" on page 447 • "MSI Mode Register (offset: 0x6000)" on page 519 • "Mode Control Register (offset: 0x6800)" on page 521 • "Power Management Debug Register – Debug Control (offset: 0x68A4)" on page 538 • "EAV REF CLOCK CONTROL Register (offset: 0x6908)" on page 543 • "NVM Auto-Sense Status Register (offset: 0x7038)" on page 562 • "SRAM Test Mode Register (offset: 0x7408)" on page 565 • "SRAM Test Mode Register (offset: 0x740c)" on page 565 • "02h: PHY_Identifier_MSB_Register" on page 579 • "03h: PHY_Identifier_LSB_Register" on page 579 **Added:** • BCM5727 to Table 1: "Product Features," on page 46 • PCIe Uart for SOL/Text Console redirection row to Table 1: "Product Features," on page 46 • BCM5725 B0 and BCM5727 A0 to Table 2: "Device Revision Levels," on page 49 • "Product ID and ASIC revision (offset: 0xF4) — Function 0" on page 276 • "APE Memory Indirect Address Register (offset: 0xF8)" on page 276 • "APE Memory Indirect Data Register (offset: 0xFC)" on page 276 • "Extended Magic Pack Registers" on page 352 – "Extended Magic Packet Length and Mask Register (offset: 0x6E0)" on page 352 – "Extended Magic Packet Pattern Register 0 (offset: 0x6E4)" on page 353 – "Extended Magic Packet Pattern Register 1 (offset: 0x6E8)" on page 353 • "APE Sleep State Clock Policy Register (offset: 0x3620)" on page 392 • "PLL Control1 Register (offset: 0x36F0)" on page 434 • "PLL Control2 Register (offset: 0x36F4)" on page 434 • "PLL Control3 Register (offset: 0x36F8)" on page 434 • "Clock Generator Control Register (offset: 0x36FC)" on page 435 • "Power Gating Mask Register (offset: 0x3700)" on page 436 • "Power Gating Mask Register for APE (offset: 0x3704)" on page 436

Revision	***Date***	***Change Description***
		• "Power Gating Control Register (offset: 0x3708)" on page 437 • "Power Good Delay Register (offset: 0x370c)" on page 437 • "Power Gating Status (offset: 0x3710)" on page 438 • "Power Gating Debug Register (offset: 0x3718)" on page 438 • "Memory Power Up/Down Wait Register (offset: 0x371C)" on page 439 • "Power Management Communication Channels (offset: 0x3720)" on page 439 • "Runtime D3 Control and Status Register (offset: 0x3724)" on page 440 • "Fast Boot Register (offset: 0x3728)" on page 441 • "Switching Regulator Control 4 Register (offset: 0x372C)" on page 441 • "Runtime D3 PCIE Configuration Register 0-6 (offset: 0x373C-0x3754)" on page 442 • "Switching Regulator Status Register (offset: 0x3758)" on page 442 • "BD RDMA Registers" on page 461 • "Non_LSO DMA Read Engine" on page 482 • "TPH ISO Control Register (offset: 0x68C0)" on page 541 • "TPH Control Register (offset: 0x68FC)" on page 542 • "TX TIME WATCHDOG LSB[0] REG (offset: 0x6918)" on page 544 • "TX TIME WATCHDOG MSB[0] REG (offset: 0x691C)" on page 545 • "TX TIME WATCHDOG LSB[1] REG (offset: 0x6920)" on page 545 • "TX TIME WATCHDOG MSB[1] REG (offset: 0x6924)" on page 546 • "EAV REF-COUNT SNAP-SHOT LSB[1] REG (offset: 0x6930)" on page 546 • "EAV REF-COUNT SNAP-SHOT MSB[1] REG (offset 0X6934)" on page 547 • "PCI Host Direct Register Access to PCIE Configuration Registers via BAR Access" on page 547 • "NVRAM State Machine Status 2 Register (offset: 0x703C)" on page 564 • "SRAM Test Mode Register (offset: 0x7410)" on page 565 • "SRAM Test Mode Register (offset: 0x7414)" on page 565 • "SRAM Test Mode Register (offset: 0x7418)" on page 566

Revision	*Date*	*Change Description*
		Removed: • Product ID and ASIC revision (offset: 0xFC) — Function 0 DBU TXDATA Register (offset: 0x3814) Not Used in BCM5725/ BCM5762/BCM57767 • Read DMA Miscellaneous Control1 Register- Debug Controls (offset: 0x4900) • Read DMA Miscellaneous Control2 Register (offset: 0x4904) • Read DMA Miscellaneous Control3 Register – Debug Control (offset: 0x4910) • ASF/Legacy SMBus Registers: – ASF Control Register (offset: 0x6C00) – SMBus Input Register (offset: 0x6C04) – SMBus Output Register (offset: 0x6C08) – ASF Watchdog Timer Register (offset: 0x6C0C) – ASF Heartbeat Timer Register (offset: 0x6C10) – Poll ASF Timer Register (offset: 0x6C14) – Poll Legacy Timer Register (offset: 0x6C18) – Retransmission Timer Register (offset: 0x6C1C) – Time Stamp Counter Register (offset: 0x6C20) – SM Bus Driver Select Register (offset: 0x6C24)
5725_5762_57767-PG100-R	10/05/12	Initial release

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

Table of Contents

List of Figures

List of Tables

About This Document

Purpose and Audience

This document covers the following devices:

- BCM5725
- BCM5762
- BCM57767

The document focuses on the registers, control blocks, and software interfaces necessary for host software programming. It is intended to complement the data sheet for the appropriate member of the NetXtreme®/NetLink® Ethernet controller family. The errata documentation (see "Revision Levels" on page 49) complements this document.

Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

Acronyms and abbreviations in this document are also defined in Appendix B: "Acronyms and Abbreviations," on page 682.

For a comprehensive list of acronyms and other terms used in Broadcom® documents, go to: http://www.broadcom.com/press/glossary.php.

Document Conventions

The following conventions may be used in this document:

Convention	*Description*
Bold	User input and actions: for example, type **exit**, click **OK,** press **Alt+C**
`Monospace`	Code: `#include <iostream>` HTML: `<td rowspan = 3>` Command line commands and parameters: `wl [-l] <command>`
< >	Placeholders for *required* elements: enter your <username> or `wl <command>`
[ ]	Indicates *optional* command-line parameters: `wl [-l]` Indicates bit and byte ranges (inclusive): [0:3] or [7:0]

References

The references in this section may be used in conjunction with this document.

Note: Broadcom provides customer access to technical documentation and software through its Customer Support Portal (CSP) and Downloads & Support site (see Technical Support).

For Broadcom documents, replace the "xx" in the document number with the largest number available in the repository to ensure that you have the most current version of the document.

Document (or Item) Name	***Number***	***Source***
Broadcom Items		
[1] NetXtreme® Ethernet Audio Video Application Note	57765-AN2xx-R	Broadcom CSP
[2] Host Programmer Interface Specification for the NetXtreme® Family of Highly Integrated Media Access Controllers[a]	57XX-PG1xx-R	Broadcom CSP
[3] Card Reader Programmer's Guide (BCM577XX Family) Application Note	57785-PG2xx-R	Broadcom CSP
[4] NetXtreme®/NetLink® Shared Memory Communication Application Note	NetXtreme-AN8xx-R	Broadcom CSP
[5] NetXtreme®/NetLink® NVRAM Configuration Options Application Note	NetXtreme-AN6xx-R	Broadcom CSP
[6] NetXtreme®/NetLink® NVRAM Access Application Note	NetXtreme-AN5xx-R	Broadcom CSP
[7] NetXtreme®/NetLink® Software Self-Boot NVRAM Application Note	NetXtreme-AN4xx-R	Broadcom CSP
[8] Self Boot Option Application Note	5754X_5787X-AN1xx-R	Broadcom CSP

[a] This BCM5XX *Programmers' Guide* contains information for the BCM5700, BCM5701, BCM5702, BCM5703, BCM5704, BCM5705, BCM5721, BCM5751, BCM5752, BCM5714, BCM5715, and BCM57XX devices

Refer to the following Broadcom documents for additional information on the Ethernet controllers:

- Data sheets documentation for the following devices: BCM5725/BCM5762/BCM57767
- Applicable BCM5725/BCM5762/BCM57767 errata documentation (each BCM5725/BCM5762/BCM57767 controller SKU has its own errata document).

Technical Support

Broadcom provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates through its customer support portal (https://support.broadcom.com). For a CSP account, contact your Sales or Engineering support representative.

In addition, Broadcom provides other product support through its Downloads & Support site (http://www.broadcom.com/support/).

Section 1: Introduction

Introduction

The NetXtreme and NetLink family of Media Access Controller (MAC) devices are highly-integrated, single-chip gigabit Ethernet LAN controller solutions for high-performance network applications. These devices integrate the following major functions to provide a single-chip solution for gigabit LAN-on-motherboard (LOM) and network interface card (NIC) applications.

- Triple-speed IEEE 802.3-compliant MAC functionality
- Triple-speed IEEE 802.3-compliant Ethernet PHY transceiver
- PCI Express (PCIe) bus interface
- On-chip packet buffer memory
- On-chip RISC processor for custom frame processing

Product Features

Table 1: Product Features

Feature	***BCM5725***	***BCM5727***	***BCM5762***	***BCM57767***
Data Management				
VLAN tag support (IEEE 802.1Q)	Yes	Yes	Yes	Yes
Layer 2 priority encoding (IEEE 802.1p)	Yes	Yes	Yes	Yes
Link aggregation (IEEE 802.3ad)	Yes	Yes	Yes	Yes
Full-duplex flow control (IEEE 802.3x)	Yes	Yes	Yes	Yes
Programmable rules checker for advanced packet filtering and classification	Yes	Yes	Yes	Yes
Frame/packet buffer memory	40 KB RX, 32 KB TX	40 KB RX, 32 KB TX	40 KB RX, 32 KB TX	40 KB RX, 32 KB TX
Jumbo frame support	Yes	Yes	Yes	Yes
TCP checksum offload on TX & RX for TCP over IPv4	Yes	Yes	Yes	Yes
UDP checksum offload on TX & RX for UDP over IPv4	Yes	Yes	Yes	Yes
IPv4 checksum offload on TX and RX	Yes	Yes	Yes	Yes

Table 1: Product Features (Cont.)

Feature	***BCM5725***	***BCM5727***	***BCM5762***	***BCM57767***
TCP checksum offload on TX for TCP over IPv6	Yes	Yes	Yes	Yes
UDP checksum offload on TX for UDP over IPv6	Yes	Yes	Yes	Yes
TCP checksum offload on RX for TCP over IPv6	Yes	Yes	Yes	Yes
UDP checksum offload on RX for UDP over IPv6	Yes	Yes	Yes	Yes
TCP segmentation offload (hardware-based) over IPv4/IPv6	Yes	Yes	Yes	Yes
Receive-side scaling (RSS)	Yes	Yes	Yes	Yes
MMRR	Yes	Yes	Yes	Yes
Multiple receive descriptor queues	Yes	Yes	Yes	Yes
Scatter/gather bus mastering architecture	Yes	Yes	Yes	Yes
Statistics for SNMP MIB II, Ethernet like MIB, Ethernet MIB (IEEE 802.3z, Clause 30)	Yes	Yes	Yes	Yes
ASF v2.0 support	Yes	No	Yes	No
iSCSI Boot	Yes	Yes	No	No
DASH	No	No	Yes	No
SMASH	Yes	No	No	No
KCS	Yes	No	No	No
PCIe Uart for SOL/Text Console redirection	Yes	No	Yes	No
NCSI	Yes	Yes	No	No
1588	Yes	Yes	No	No
Ethernet Audio Video (EAV) Support	Yes	Yes	Yes	Yes
Teaming	Yes	Yes	Yes	Yes
Host Bus Interfaces				
PCIe v2.0 x1 bus interface	Yes	No	Yes	Yes
LAN Interfaces				
10/100/1000BASE-T full-duplex/ half-duplex	Yes (see note)	Yes (see note)	Yes	Yes
Integrated 10/100/1000BASE-T transceiver	Yes (see note)	Yes (see note)	Yes	Yes
Internal MII/GMII Interface	Yes	Yes	Yes	Yes
Other Bus Interfaces				

Table 1: Product Features (Cont.)

Feature	***BCM5725***	***BCM5727***	***BCM5762***	***BCM57767***
SMBus 2.0 interface	Yes	Yes	Yes	No
Interface to Flash memory	Yes	Yes	Yes	Yes
Interface to Serial EEPROM	Yes	Yes	Yes	Yes
Flash Autoconfig Support	Yes	Yes	Yes	Yes
Self-Test				
Test modes (BIST, SCAN, etc.)	Yes	Yes	Yes	Yes
Factory-level JTAG support	No	No	No	No
Technology				
High-performance, low-overhead SW/HW interface	Yes	Yes	Yes	Yes
High-speed on-chip RISC processor	Yes	Yes	Yes	Yes
Wake-on-LAN (WOL)	Yes	Yes	Yes	Yes
Process voltage	1.0V	1.0V	1.0V	1.0V
CMOS linewidth	40 nm	40 nm	40 nm	40 nm
5V-tolerant PCI I/Os	Yes	Yes	Yes	Yes

Revision Levels

See Table 2 for the revision levels of the Ethernet controllers covered by this document. Host software can use the PCI Revision ID and Chip ID information in the PCI configuration registers to determine the revision level of the Ethernet controller on the board, and then load the appropriate workaround described in the errata sheets.

The Broadcom PCI vendor ID is 0x14E4. Table 2 shows the default values of PCI device IDs. These values may be modified by firmware in accordance with the manufacturing information supplied in NVRAM (see "NVRAM Configuration" on page 65 for more details).

Table 2: Device Revision Levels

Devices	*Device ID*[a]	*Revision Level*	*PCI Revision ID*[b]	*Chip ID*[c]	*Errata Sheet*[d]
BCM57767	0X1683	A0	0X00	0xF000xxxx	Yes
BCM5762	0x16B7	A0	0x00	0xF000xxxx	Yes
BCM5725	0X1643	A0	0x00	0xF000xxxx	Yes
BCM5725	0X1643	B0	0x00	0xF100xxxx	–
BCM5727	0X16f3	A0	0x00	0xF000xxxx	–
SD/MMC[e]	0x16BC	A0	0x00	0xF000xxxx	–
SD4.0	0x1640	A0	0X00	0xF000xxxx	–

[a] See "Device ID and Vendor ID Register (Offset: 0x00) — Function 0" on page 245.
[b] See "PCI Classcode and Revision ID Register (offset: 0x8) — Function 0" on page 248 The hardware default value of this register is 0x00. The boot code firmware programs this register with the value as given in the table.
[c] See "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256. The lower 16 bits are don't cares for determining chip id.
[d] See the appropriate errata documentation for the errata information and resolutions.
[e] Secure Digital card reader function in BCM57767

Programming the Ethernet Controllers

See Table 2 on page 49 for the revision levels of the Ethernet controllers. Host software can use the PCI Revision ID and Chip ID information in the PCI configuration registers to determine the revision level of the Ethernet controller on the board, and then load the appropriate workarounds described in the errata sheets.

Choice of host access mode determines the mailboxes:

- Host standard mode uses the high-priority mailboxes (see "High-Priority Mailbox Registers" on page 319).
- Indirect mode uses the low-priority mailboxes (see "Low Priority Mailboxes" on page 513).

The reference documents for Ethernet controller software development include this manual and the errata documentation (see "Revision Levels" on page 49) that provide the necessary information for writing a host-based device driver. The Broadcom Linux® driver (a.k.a. "tg3") is also a very good reference source for writing your own driver.

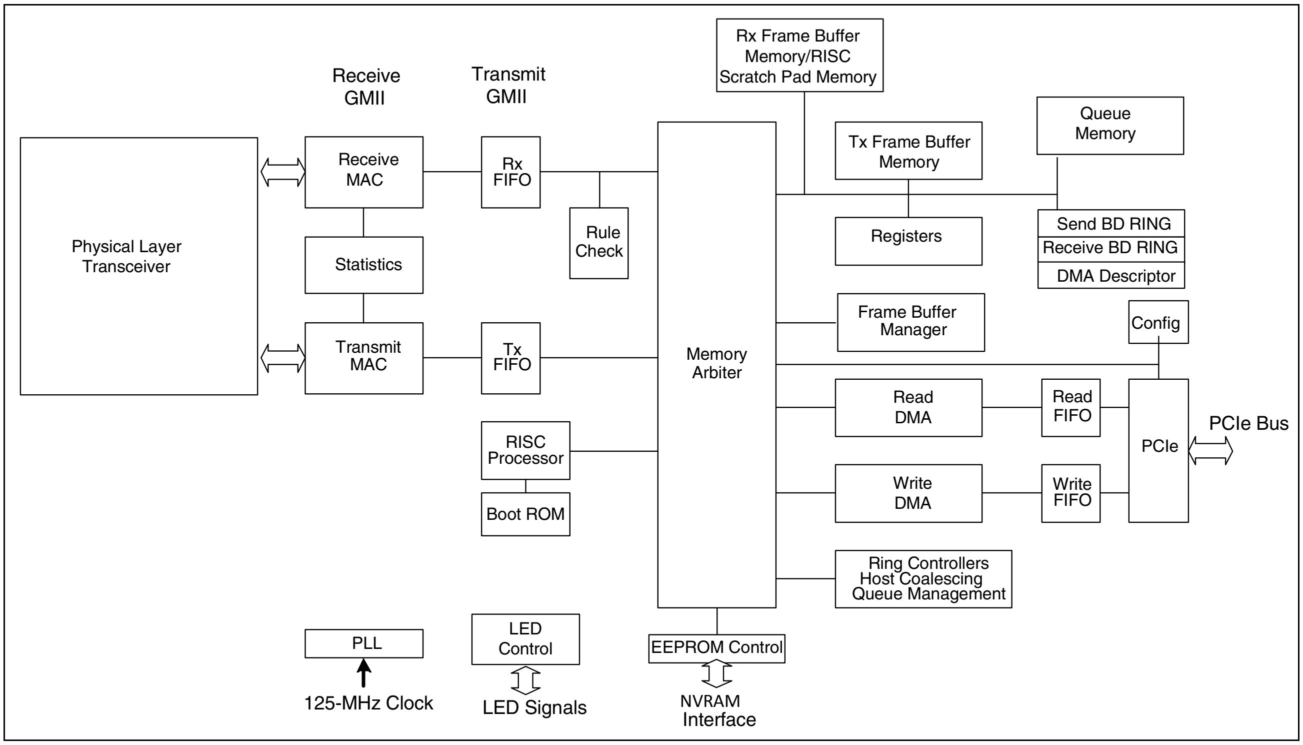
The programming model for the NetXtreme/NetLink Ethernet controllers does not depend on OS or processor instruction sets. Programmers using Motorola 68000, Intel x86, or DEC Alpha host instruction sets can leverage this document to aid in device driver development. Concepts provided in this document are also applicable to device drivers native to any operating system (i.e., DOS, UNIX, Microsoft, or Novell).

Section 2: Hardware Architecture

Theory of Operation

Figure 1 shows the major functional blocks and interfaces of the Ethernet controllers covered in this document. There are two packet flows: MAC-transmit and receive. The device's DMA engine bus-masters packets from host memory to device local storage, and vice-versa. The host bus interface is compliant with PCIe standards. The RX MAC moves packets from the integrated PHY into device internal memory. All incoming packets are checked against a set of QOS rules and then categorized. When a packet is transmitted, the TX MAC moves data from device internal memory to the PHY. Both flows operate independently of each other in full-duplex mode. An on-chip RISC processor is provided for running value-added firmware that can be used for custom frame processing. The on-chip RISC operates independently of all the architectural blocks; essentially, RISC is available for the auxiliary processing of data streams.

Figure 1: Functional Block Diagram

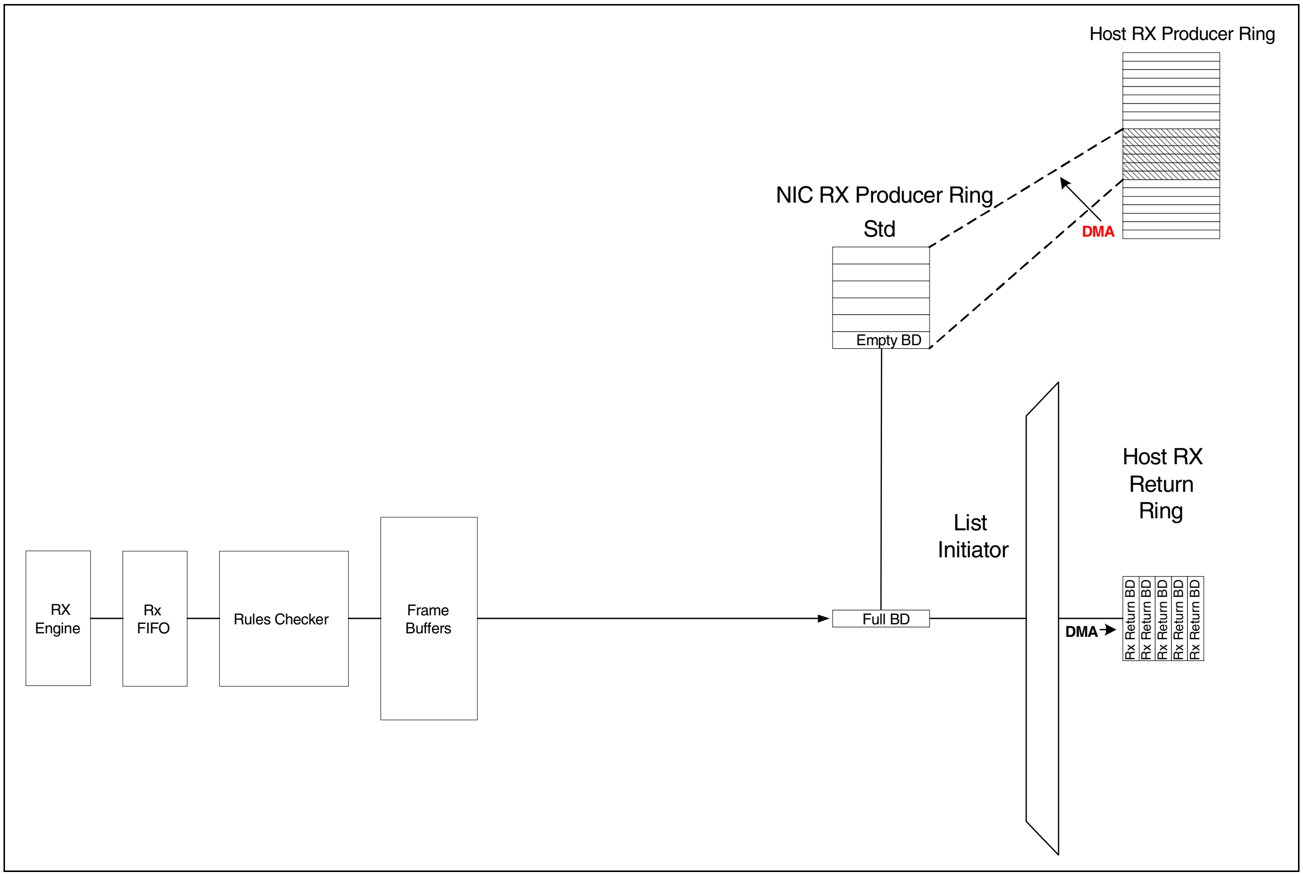


Receive Data Path

RX Engine

The receive engine (see Figure 2) activates whenever a packet arrives from the PHY.

Figure 2: Receive Data Path



The receive engine performs the following four functions:

- Moves the data from the PHY to an internal FIFO
- Moves the data from the FIFO to NIC internal memory
- Classifies the frame and checks it for rules matches
- Performs the offloaded checksum calculations

RX FIFO

The RX FIFO provides elasticity while data is read from PHY transceiver and written into internal memory. There are no programmable settings for the RX FIFO. This FIFO's operation is completely transparent to host software.

Rules Checker

The rules checker examines frames. After a frame has been examined, the appropriate classification bits are set in the buffer descriptor. The rules checker is part of the RX data path and the frames are classified during data movement to NIC memory. The following frame positions may be established by the rules checker:

- IP Header Start Pointer
- TCP/UDP Header Start Pointer
- Data Start Pointer

RX List Initiator

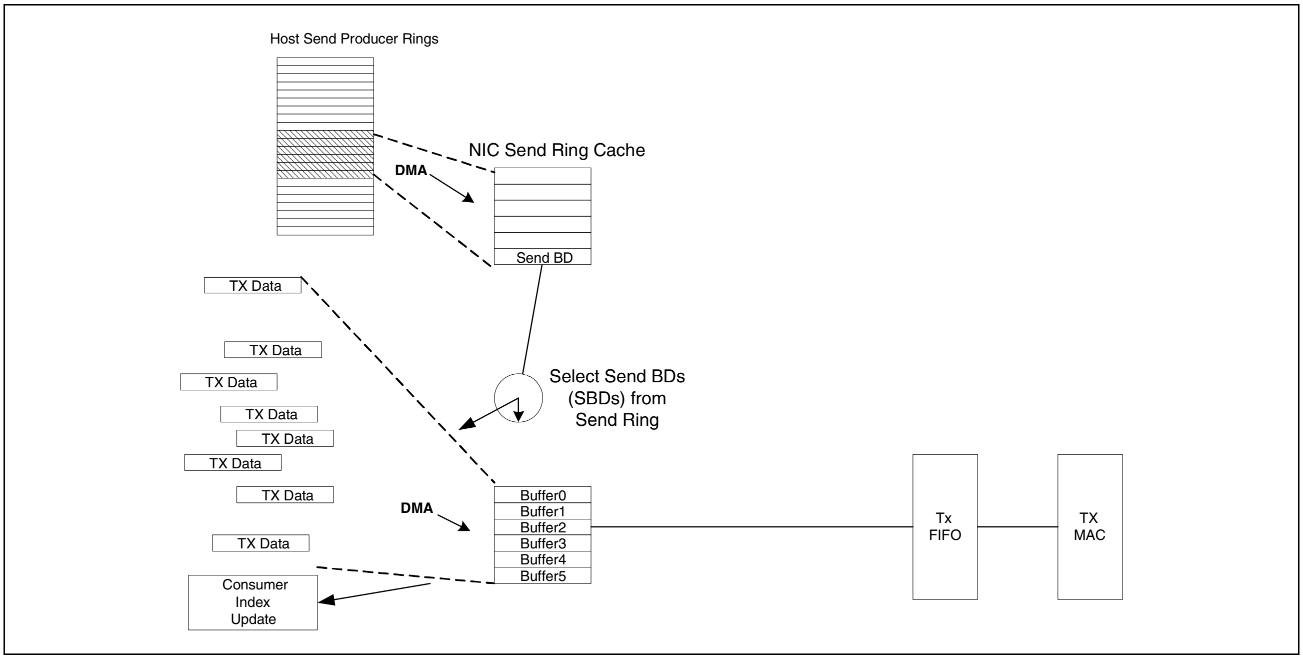
The RX List Initiator function activates whenever the receive producer index for any of receive buffer descriptor (BD) rings is written. This value is located in one of the receive BD producer mailboxes. The host software writes to the producer mailbox and causes the RX Initiator function to enqueue an internal data structure/request, which initiates the DMA of one or more new BDs to the NIC. The actual DMAs generated depend on the comparison of the value of the received BD host producer index mailbox, the NIC copy of the received BD consumer index, and the local copy of the received BD producer index.

Transmit Data Path

TX MAC

The Read DMA engine moves packets from host memory into internal NIC memory (see Figure 3). When the entire packet is available, the transmit MAC is activated.

Figure 3: Transmit Data Path



The transmit MAC is responsible for the following functions:

- Moving data from NIC internal memory into TX FIFO
- Moving data from TX FIFO to PHY
- Checksum substitutions (not calculation)
- Updating statistics

TX FIFO

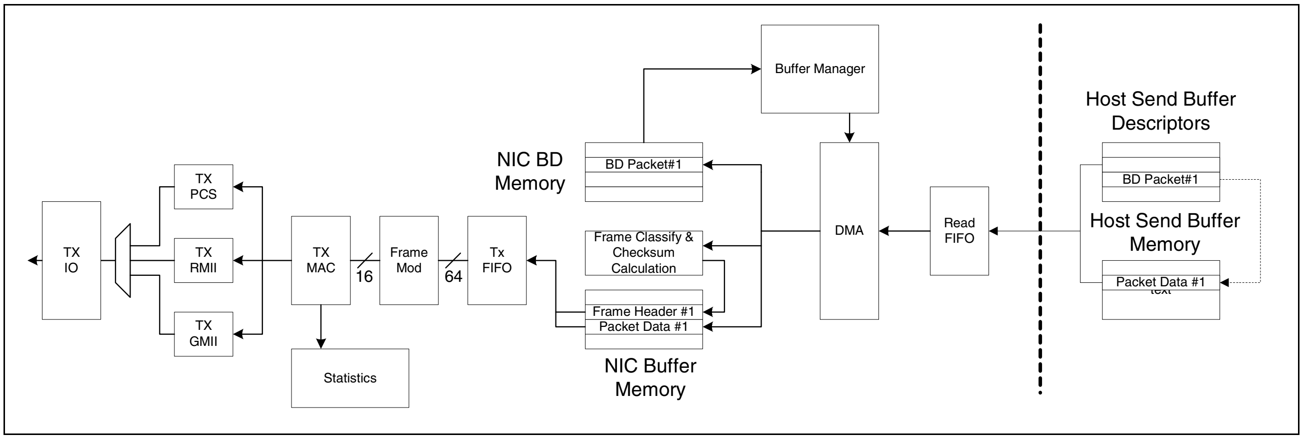
The TX FIFO provides elasticity while data is moved from device internal memory to PHY. There are no programmable settings for the TX FIFO. This FIFO's operation is completely transparent to host software.

DMA Read

Read Engine

The DMA read engine (see Figure 4) activates whenever a host read is initiated by the send or receive data paths.

Figure 4: DMA Read Engine



The DMA read engine de-queues an internal data structure/request and performs the following functions:

- DMAs the data from the host memory to an internal Read DMA FIFO
- Moves the data from the Read DMA FIFO to NIC internal memory
- Classifies the frame
- Performs checksum calculations
- Copies the VLAN tag field from the DMA descriptor to the frame header

Read FIFO

The read FIFO provides elasticity during data movement from host memory to device local memory. The memory arbiter is a gatekeeper for multiple internal blocks; several portions of the architecture may simultaneously request internal memory. The PCI read FIFO provides a small buffer for the data read from host memory while the Read DMA engine requests internal memory via the memory arbiter. The data is moved out of the read DMA FIFO into device local memory once a memory data path is available. The FIFO isolates the PCI clock domain from the device clock domain. This reduces latency internally and externally on the PCI bus. The PCIe Read DMA FIFO holds 1024 bytes. The operation of the read DMA FIFO is transparent to host software.The Read DMA engine makes sure there is enough space in internal TX Packet Buffer Memory before initiating a DMA request for transfer of TX packet data from host memory to device internal packet memory.

Buffer Manager

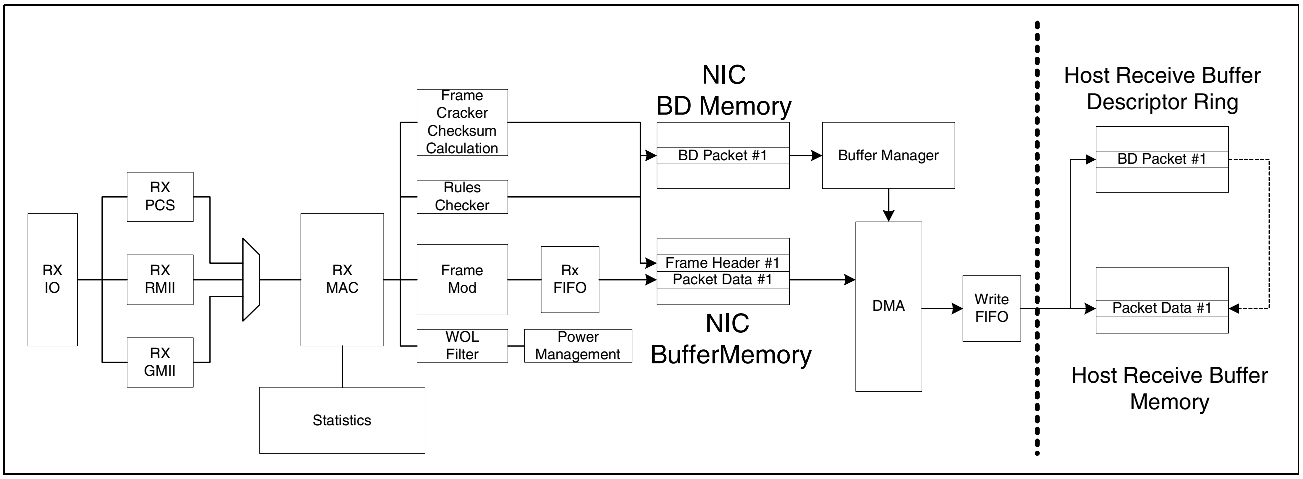
The buffer manager maintains pools of internal memory used by transmit and receive engines. The buffer manager has logic blocks for allocation, free, control, and initialization of internal memory pools. The DMA read engine requests internal memory for BDs and frame data. Figure 4 on page 55 shows the transmit data path using the DMA Read Engine. The read DMA engine also fetches RX BDs for the receive data path.

DMA Write

Write Engine

The DMA write engine, as shown in Figure 5, activates when a host write is initiated by the send or receive data paths.

Figure 5: DMA Write Engine



The DMA write engine de-queues an internal request and performs the following functions:

- Gathers the data from device internal memory into the write DMA FIFO
- DMAs the data to the host memory from the write FIFO
- Performs byte and word swapping
- Interrupts the host using a line or message signaled interrupt

Write FIFO

The write FIFO provides elasticity during data movement from device memory to the host memory. The write FIFO absorbs small delays created by PCIe bus arbitration. The NetXtreme family uses the write FIFO to buffer data, so internal memory arbitration is efficient. Additionally, the FIFO isolates the PCI clock domain from the device's clock domain. This reduces latency on the PCI bus during the write operation (wait states are not inserted while data is fetched from internal memory). The operation of the write DMA FIFO is transparent to host software.

Buffer Manager

The buffer manager maintains pools of internal memory used in transmit and receive functions. The buffer manager has logic blocks for allocation, free, control, and initialization of internal memory pools. The receive MAC requests NIC RX Mbuf memory so inbound frames can be buffered. The read DMA engine requests the device TX Mbuf memory for buffering the packets from host memory before they are sent out on the wire. The DMA write engine requests a small amount of internal memory for DMA and interrupt operations. The usage of this internal memory is transparent to host software, and does not affect device/system performance.

LED Control

Refer to section "LED Control" in the applicable data sheet.

Memory Arbiter

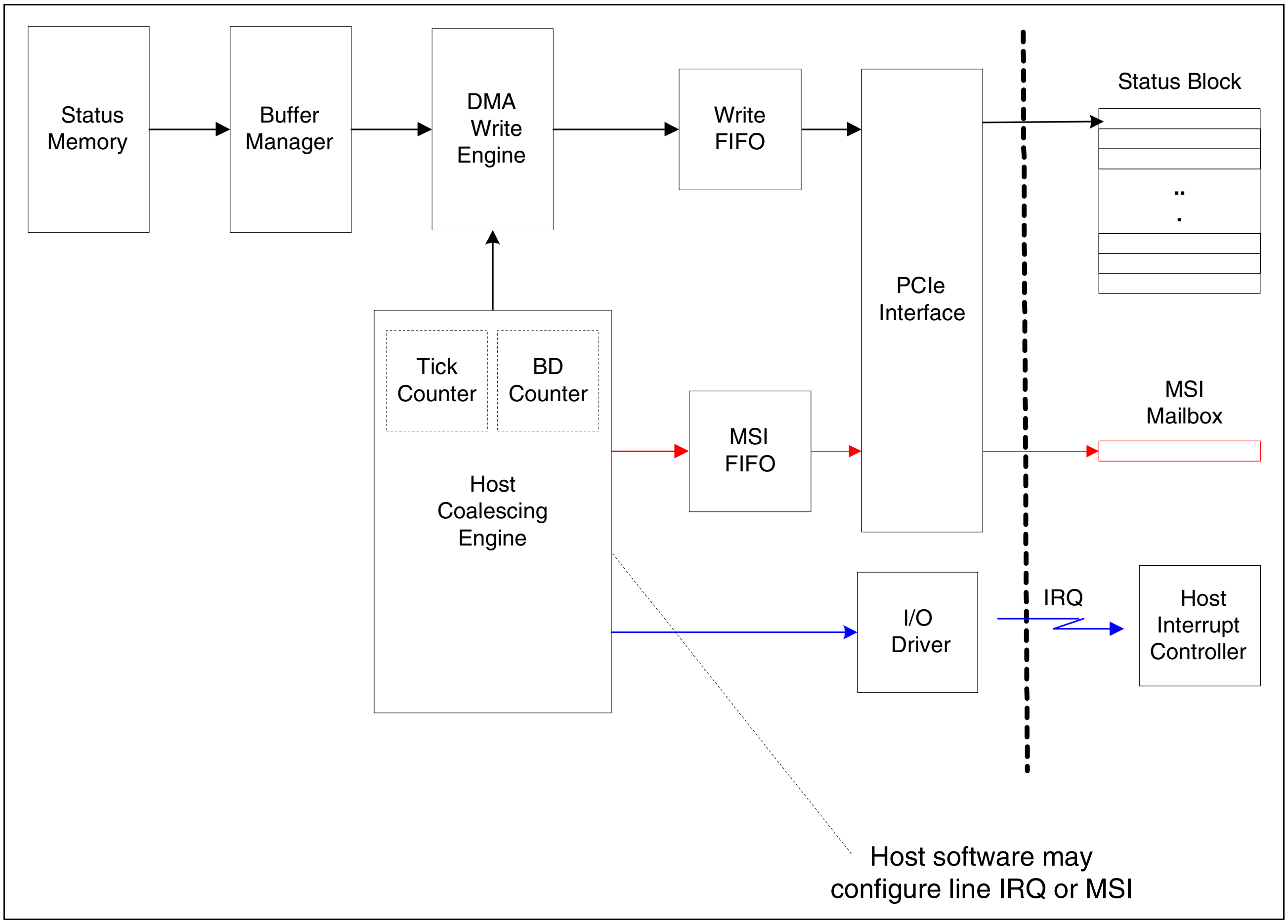
The Memory Arbiter (MA) is a gatekeeper for internal memory access. The MA is responsible for decoding the internal memory addresses that correspond to Ethernet controller data structures and control maps. If a functional block faults or traps during access to internal memory, the MA handles the failing condition and reports the error in a status register. In addition to architectural blocks, the MA provides a gateway for the RISC processor to access local memory. The RISC has an MA interface that pipelines up to three access requests. The MA negotiates local memory access, so all portions of the architecture are provided with fair access to memory resources. The MA prevents starvation and bounds access latency. Host software may enable/disable/reset the MA, and there are no tunable parameters.

Host Coalescing

Host Coalescing Engine

The Host Coalescing Engine is responsible for pacing the rate at which the NIC updates the send and receive ring indices located in host memory space. The completion of a NIC update is reflected through an interrupt on the Ethernet controller INTA pin or a Message Signalled Interrupt (MSI). Although update criteria are calculated separately, all updates occur at once. This is because all of the ring indices are in one status block, and any host update updates all ring indices simultaneously. The Host Coalescing Engine triggers based on a tick and/or a frame counter.

Figure 6: Host Coalescing Engine



A host update occurs whenever one of the following criteria is met:

- The number of BDs consumed for frames received, without updating receive indices on the host, is equal to or has exceeded the threshold set in the Receive_Max_Coalesced_BD register (see "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446).
- The number of BDs consumed for transmitting frames, without updating the send indices, on the host is equal to or has exceeded the threshold set in the Send_Max_Coalesced_BD register (see "Send Max Coalesced BD Count During Interrupt (offset 0x3C1C)" on page 447). Updates can occur when the number of BDs (not frames) meets the thresholds set in the various coalescing registers (see Section 10: "Interrupt Processing," on page 219 for more information).
- The receive coalescing timer has expired, and new frames have been received on any of the receive rings, and a host update has not occurred. The receive coalescing timer is then reset to the value in the Receive_Coalescing_Ticks register (see "Send Coalescing Ticks Register (offset: 0x3C0C)" on page 445).
- The send coalescing timer has expired, and new frames have been consumed from any send ring, and a host update has not occurred. The send coalescing timer is then reset to the value in the Send_Coalescing_Ticks register.

MSI FIFO

This FIFO is eight entries deep and four bits wide. This FIFO is used to send MSIs via the PCI interface. The host coalescing engine uses this FIFO to enqueue requests for the generation of MSI. There are no configurable options for this FIFO and this FIFOs operation is completely transparent to host software.

Status Block

This data structure contains consumer and producer indices/values. Host software reads this control block, to assess hardware updates in the send and receive rings. Two copies of the status block exist. The local copy is DMAed to host memory by the DMA write engine. Host software does not want to generate PCI transactions to read ring status; rather quicker memory bus transactions are desired. The host coalescing engine enqueues a request to the DMA write engine, so host software gets a refreshed copy of status. The status block is refreshed before a line IRQ or MSI is generated. See "Status Block Format" on page 83 for a complete discussion of the status block.

10BT/100BTX/1000BASE-T Transceiver

Auto-Negotiation

The Ethernet controller devices negotiate their mode of operation over the twisted-pair link using the auto-negotiation mechanism defined in the IEEE 802.3u and IEEE 802.3ab specifications. Auto-negotiation can be enabled or disabled by hardware or software control. When the auto-negotiation function is enabled, the Ethernet controllers automatically choose the mode of operation by advertising its abilities and comparing them with those received from its link partner. The Ethernet controllers can be configured to advertise 1000BASE-T full-duplex and/or half-duplex, 100BASE-TX full-duplex and/or half-duplex, and 10BASE-T full-duplex and/or half-duplex. The transceiver negotiates with its link partner and chooses the highest operating speed and duplex that are common between them. Auto-negotiation can be disabled for testing or for forcing 100BASE-TX or 10BASE-T operation, but is always required for normal 1000BASE-T operation.

Automatic MDI Crossover

During auto-negotiation, one end of the link must perform an MDI crossover so that each transceiver’s transmitter is connected to the other receiver. The Ethernet controllers can perform an automatic MDI crossover when the Disable Automatic MDI Crossover bit in the PHY Extended Control register is disabled, thus eliminating the need for crossover cables or cross-wired (MDIX) ports. During auto-negotiation, the Ethernet controllers normally transmit on TRD±{0} and receive on TRD±{1}. When connected to another device that does not perform the MDI crossover, the Ethernet controller automatically switches its transmitter to TRD±{1} and its receiver to TRD±{0} to communicate with the remote device. If two devices that both have MDI crossover capability are connected, an algorithm determines which end performs the crossover function. During 1000BASE-T operation, the Ethernet controllers swap the transmit symbols on pairs 0 and 1 and pairs 2 and 3 if auto-negotiation completes in the MDI crossover state. The 1000BASE-T receiver automatically detects pair swaps on the receive inputs and aligns the symbols properly within the decoder.

PHY Control

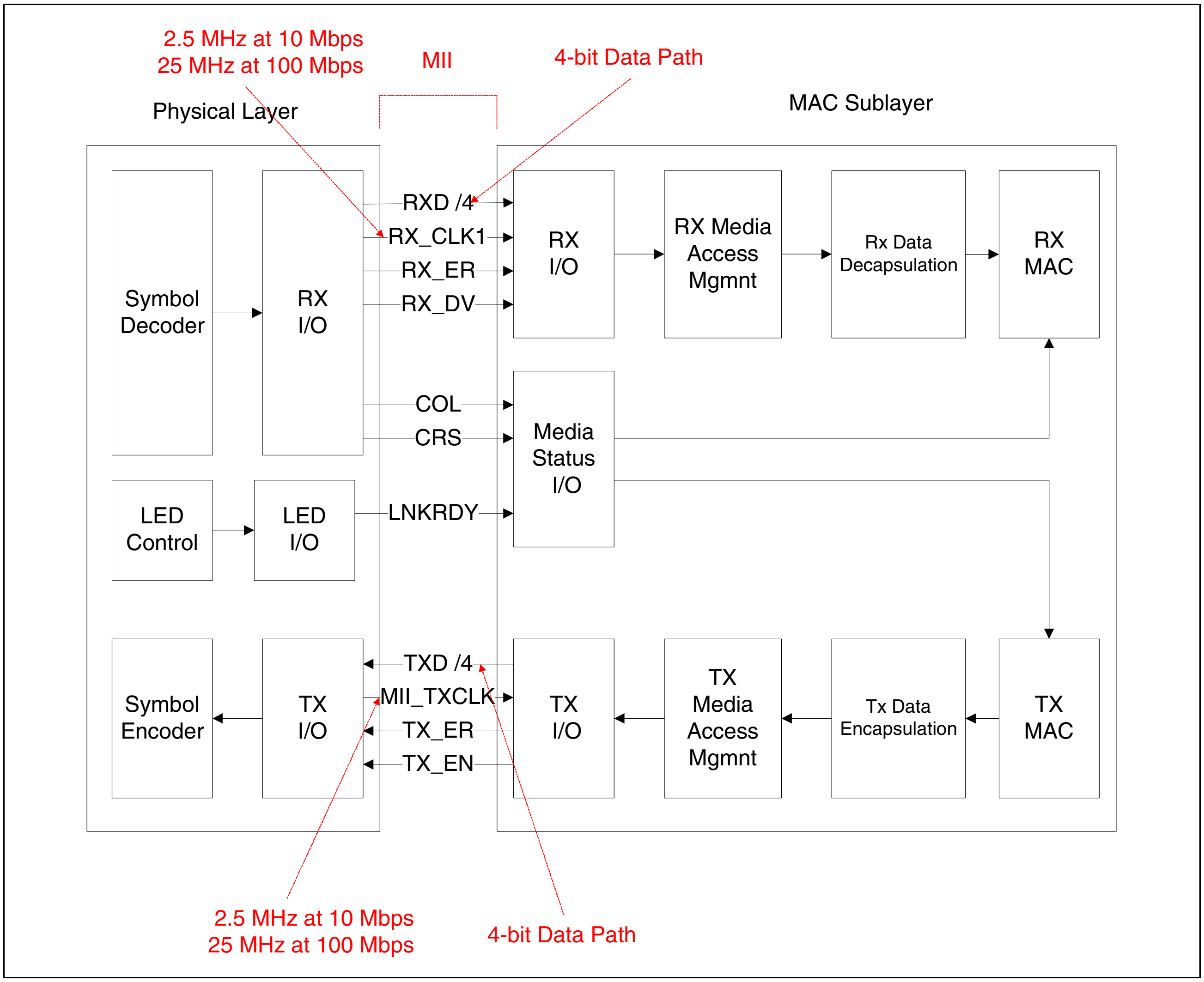
The NetXtreme/NetLink Ethernet controller supports the following physical layer interfaces:

- The MII is used in conjunction with 10/100 Mbps copper Ethernet transceivers.
- GMII supports 1000 Mbps copper Ethernet transceivers.

MII Block

The MII interconnects the MAC and PHY sublayers (as shown in Figure 7 on page 61).

Figure 7: Media Independent Interface



The specifics of MII may be located in section 22 of the IEEE 802.3 specification. RXD[3:0] are the receive data signals; TXD[3:0] are the transmit data signals. MII operates at both 10-Mbps and 100-Mbps wire-speeds. (Gigabit Ethernet uses the GMII standard.) When MAC and PHY are configured for 10 Mbps operation, the RX_CLK1 and MII_TXCLK clocks run at 2.5 MHz. Both RX_CLK1 and MII_TXCLK are sourced by the PHY. 100 Mbps wire speed requires RX_CLK1 and MII_TXCLK to provide a 25 MHz reference clock. Receive Data Valid (RX_DV) is asserted when valid frame data is received; at any point during data reception, the PHY may assert Receive Error (RX_ER) to indicate a receive error. The MAC will record this error in the statistics block. The MAC may discard a bad RX frame—exception being sniffer/promiscuous modes (see Allow_Bad_Frames bit in MAC mode register). The Transmit Enable (TX_EN) signal is asserted when the MAC presents the PHY with a valid frame for transmission. The MAC may assert TX_ER to indicate the remaining portion of frame is bad. The PHY will insert Bad Code symbols into the remaining portion of the frame. A detected collision in half-duplex mode may be such a scenario where TX_ER is asserted. The PHY will assert COL when a collision is detected. The COL signal is routed to both the RX and TX MACs. The transmit MAC will back off transmission and the RX MAC will throw away partial frames.

The 10 Mbps physical layer uses Differential Manchester encoding on the wire. Manchester encoding uses two encoding levels: 0 and 1. 100 Mbps Ethernet requires MLT-3 waveshaping on the transmission media. MLT-3 uses three encoding levels: –1, 0, and 1. Both physical signaling protocols are transparent to the MAC sublayer and are digitized by the PHY. The PHY encodes/decodes analog waveforms at its lower edge while the PHY presents digital data at its upper edge (MII).

GMII Block

The GMII is full-duplex (see Figure 8 on page 63); the send and receive data paths operate independently.

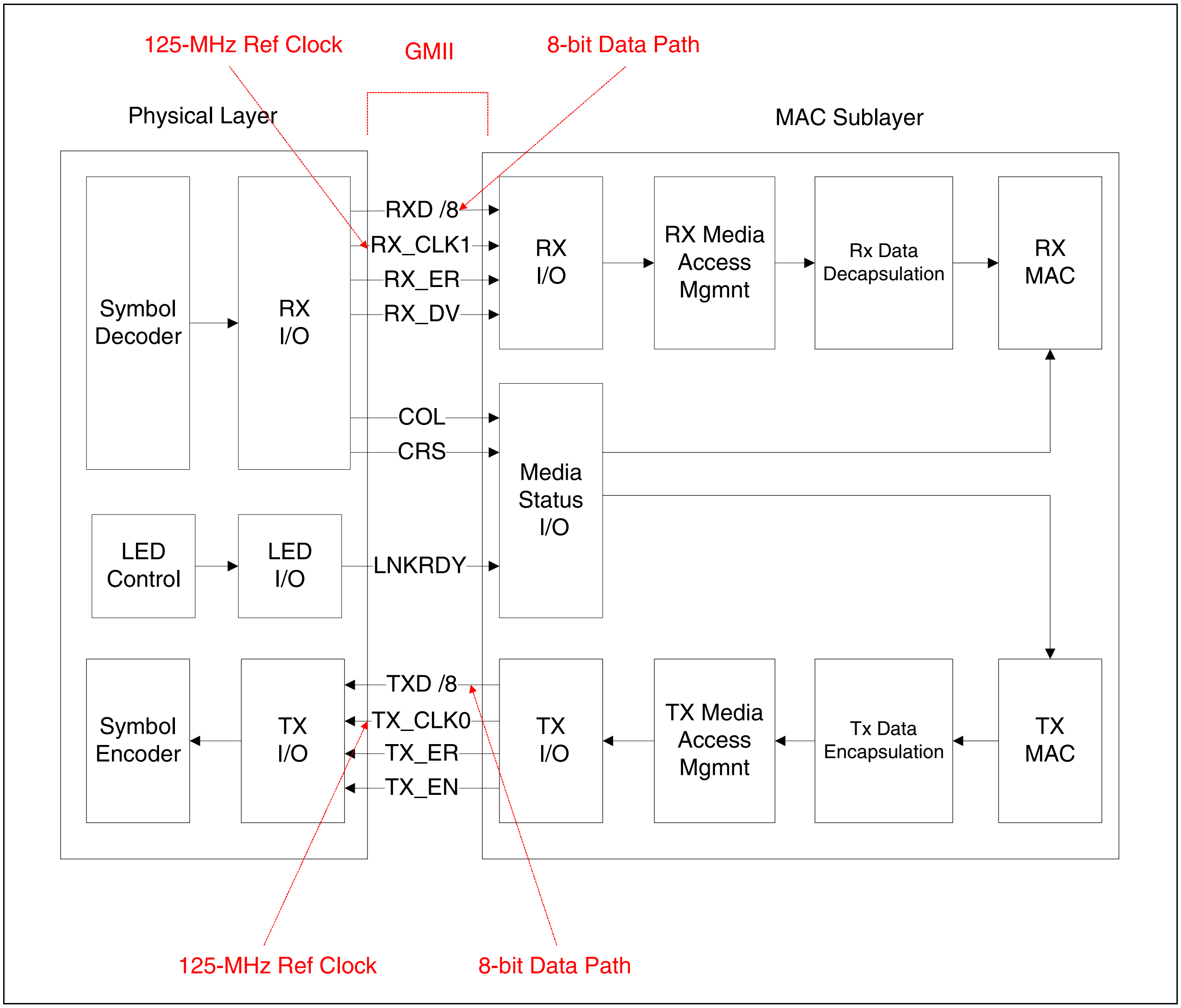
The transmit signals TXD[7:0] create a eight-bit wide data path. The TXD[7:0] signals are synchronized to the reference clockTX_CLK0. The TX_CLK0 clock runs at 125 MHz and is sourced by the MAC sublayer. Transmit Error (TX_ER) is asserted by the MAC sublayer. The PHY will transmit a bad code until TX_ER is de-asserted by the MAC. TX_ER is driven synchronously with TX_CLK0. The Transmit Enable (TX_EN) indicates that valid data is presented on the TXD lines. The TXD[7:0] data is framed on the rising edge of TX_EN.

The receive data path is also eight bits wide. RXD[7:0] are sourced by the PHY. When valid data is presented to the MAC sublayer, the PHY will also assert Receive Data Valid (RX_DV). The rising edge of RX_DV indicates the beginning of a frame sequence. The PHY drives the reference clock RX_CLK1, so the MAC sublayer can synchronize data sampling on RXD[7:0]. The PHY may assert RX_ER to indicate frame data is invalid; the MAC sublayer must consider frames in progress incomplete.

When the MAC operates in half-duplex mode, a switch or node may transmit a jamming pattern. The PHY will drive the Collision (COL) signal so the MAC may back off transmission and throw away partially received packet(s). The COL signal will also cause the TX MAC to stop the transmission of a packet. The COL signal is not driven for full-duplex operation since collisions are undefined. The PHY will drive Carrier Sense (CRS) as a response to traffic being sent/received. The MAC sublayer can monitor traffic and subsequently drive traffic LEDs.

Pulse Amplitude Modulated Symbol (PAM5) encoding is leveraged for Gigabit Ethernet wire transmissions. PAM5 uses five encoding levels: –2, –1, 0, 1, and 2. Four symbols are transmitted in parallel on the four twisted-wire pairs. The four symbols create a code group (an eight-bit octet). The process of creating the code-group is called 4D-PAM5. Essentially, eight data bits are represented by four symbols. Table 40-1 in the IEEE 802.3ab specification shows the data bit to symbol mapping. The code group representation is also referred to as a quartet of quinary symbols {TA, TB, TC, TD}. The modulation rate on the wire is measured at 125 Mbaud. The resultant bandwidth is calculated by multiplying 125 MHz by eight bits, for
1000 Mbps wire speed.

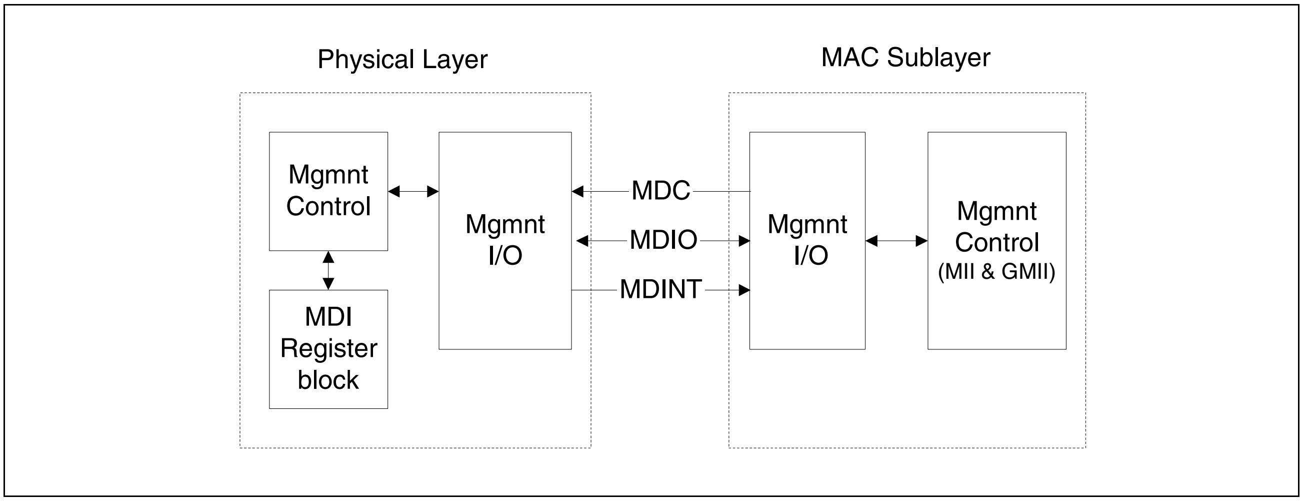
Figure 8: GMII Block



MDIO Register Interface

Figure 9 shows the MDI register interface.

Figure 9: MDI Register Interface



Management Data Clock

The Management Data Clock (MDC) is driven by the MAC sublayer. The PHY will sink this signal to synchronize data transfer on the MDIO signal—MDC is a reference clock. This clock is not functionally associated to either RX_CLK or TX_CLK. The minimum period for this clock is 400 ns with high and low times having 160 ns duration.

Management Data Input/Output

The Management Data Input/Output (MDIO) signal passes control and status data, between the MAC and PHY sublayers. MDIO is a bidirectional signal, meaning both the PHY and MAC may transfer data. The MAC typically transfers control information and polls status; whereas, the PHY transfers status back to the MAC, using MDIO.

Management Data Interrupt

The integrated Broadcom PHY may be programmed to generate interrupts. A PHY status change initiates a Management Data Interrupt (MDINT). A MDI mask register allows host software to selectively enable/disable status types, which cause MDINT notification. The PHY will assert INTR until software clears the interrupt. Reading the status register will clear the interrupt.

Management Register Block

The layout and configuration of MDI register block is device dependent. The MDI register block is the control/status access point, which host software may read/write. The IEEE 802.3 specification defines a basic register block for MII and GMII; the basic register set contains control and status registers. GMII also exposes an extended register set, used in 1000 Mbps configuration/status. The fundamental point is to understand that the MDC and MDIO signals are used to access the MDI register block.

Section 3: NVRAM Configuration

Overview

Broadcom NetXtreme and NetLink controllers require the use of an external non-volatile memory (NVRAM) device (Flash or SEEPROM), which contains a boot code program that the controller's on-chip CPU core loads and executes upon release from reset. This external NVRAM device also contains many configuration items that direct the behavior of the controller, enable/disable various management and/or value-add firmware components, etc.

All configuration settings are default-configured in the official release binary image files provided in Broadcom's CD software releases. However, the settings chosen as default by Broadcom may not be what best suits a particular OEM's application, so some settings may need to be changed by the OEM.

The BCM5725/BCM5762/BCM57767 supports the following boot code styles:

- "Legacy"—Boot code plus configuration options fully contained in an external 8k byte NVRAM device
- "Self-boot"—Uses a fixed internal ROM'd copy of the boot code program (requires only a very small external NVRAM, or none at all).

If using self-boot there are two options:

- Use only a very small (256 byte) external NVRAM to hold configuration items/self-boot code patches.
- Use no external NVRAM at all; configuration items/self-boot patches are stored in on-chip One-Time-Programmable memory (OTP).

The BCM5725/BCM5762/BCM57767 introduces a new feature which involves a limited amount of on-chip OTP containing the following configuration items (with limitations regarding how many times they may be reprogrammed).

Table 3: On-chip One-Time-Programmable Memory (OTP) Configuration Options

Configuration Item	*Reprogrammability*
Ethernet MAC address	4 times
PCIe Device ID	0 times (programmed at ATE)
PCIe Sub Device ID	4 times
PCIe Sub Vendor ID	4 times
WoL Enable	4 times
WoL Limit 10	2 times
LOM design	2 times
Phy Auto Power Down	2 times
LED mode	4 times
Reverse N-Way	2 times
Power Saving mode	2 times

Table 3: On-chip One-Time-Programmable Memory (OTP) Configuration Options

Configuration Item	*Reprogrammability*
Cable Sense Enable	2 times

The BCM5725/BCM5762/BCM57767 architecture utilizes a fixed number of OTP bits for each configuration item. The number of OTP bits allocated for each configuration item exceeds the minimum required number of bits to house a value for each configuration item. This method allows for a scheme to offer “reprogrammability” of configuration items (similar to legacy NVRAM), but with a strict limitation to the total number of reprogram cycles. Each time you reprogram a specific configuration item you use up one subset of the allocated OTP bits for that item.

Refer to Broadcom Application Note NetXtreme-AN40X-R (NetXtreme®/NetLink™ Software Self-Boot NVRAM Application Note) for additional detail regarding self-boot NVRAM structure.

Note: All 0's or all 1's cannot be used as valid values for PCI device ID/subsystem ID/sub vendor ID/ MAC address if using internal OTP to store these items.

Details relating to the legacy style NVRAM organization can be found in *NetXtreme/NetLink NVRAM Access* Broadcom application note (Netxtreme-AN50X-R). Some of the topics addressed by this application note include the following:

- Programming NVRAM (sample C code, x86 assembly)
- NVRAM map
- Configuration settings
- Boot code
- Multiple boot agent (MBA), PXE, etc.

Note: NVRAM CRC-32: There are multiple distinct regions contained within the NVRAM map. Each of these regions has its own CRC-32 checksum value associated with it. If any data element contained within a region is modified, then that region's CRC-32 value must also be updated. Details relating to calculating the CRC-32 can be found in *Calculating CRC32 Checksums for Broadcom NetLink, NetXtreme, and NetXtreme II Controllers* Broadcom application note (NetXtreme_NetXtremeII-AN20X-R).

Self-Boot

Some NetLink controllers offer a capability known as self-boot. Self-boot allows the controller to use a very small, low-cost, external NVRAM device that contains only a very condensed amount of configuration information, along with any small boot code patches that may be necessary to optimize the functionality of a particular controller.

Details relating to self-boot can be found in *Self Boot Option* (5754X_5787X-AN10X-R) and *NetXtreme/NetLink Software Self-Boot NVRAM* (NetXtreme-AN40X-R) Broadcom application notes.

Section 4: Common Data Structures

Theory of Operation

Several device data structures are common to the receive, transmit, and interrupt processing routines. These data structures are hardware-related and are used by device drivers to read and update state information.

Descriptor Rings

In order to send and receive packets, the host and the controller use a series of shared buffer descriptor (BD) rings to communicate information back and forth. Each ring is composed of an array of buffer descriptors that reside in host memory. These buffer descriptors point to either send or receive packet data buffers. The largest amount of data that a single buffer may contain is 65535 (64K-1) bytes (The length field in BD is 16 bits). Multiple descriptors can be used per packet in order to achieve scatter-gather DMA capabilities.

Note: The maximum number of Send BDs for a single packet is (0.75)*(ring size).

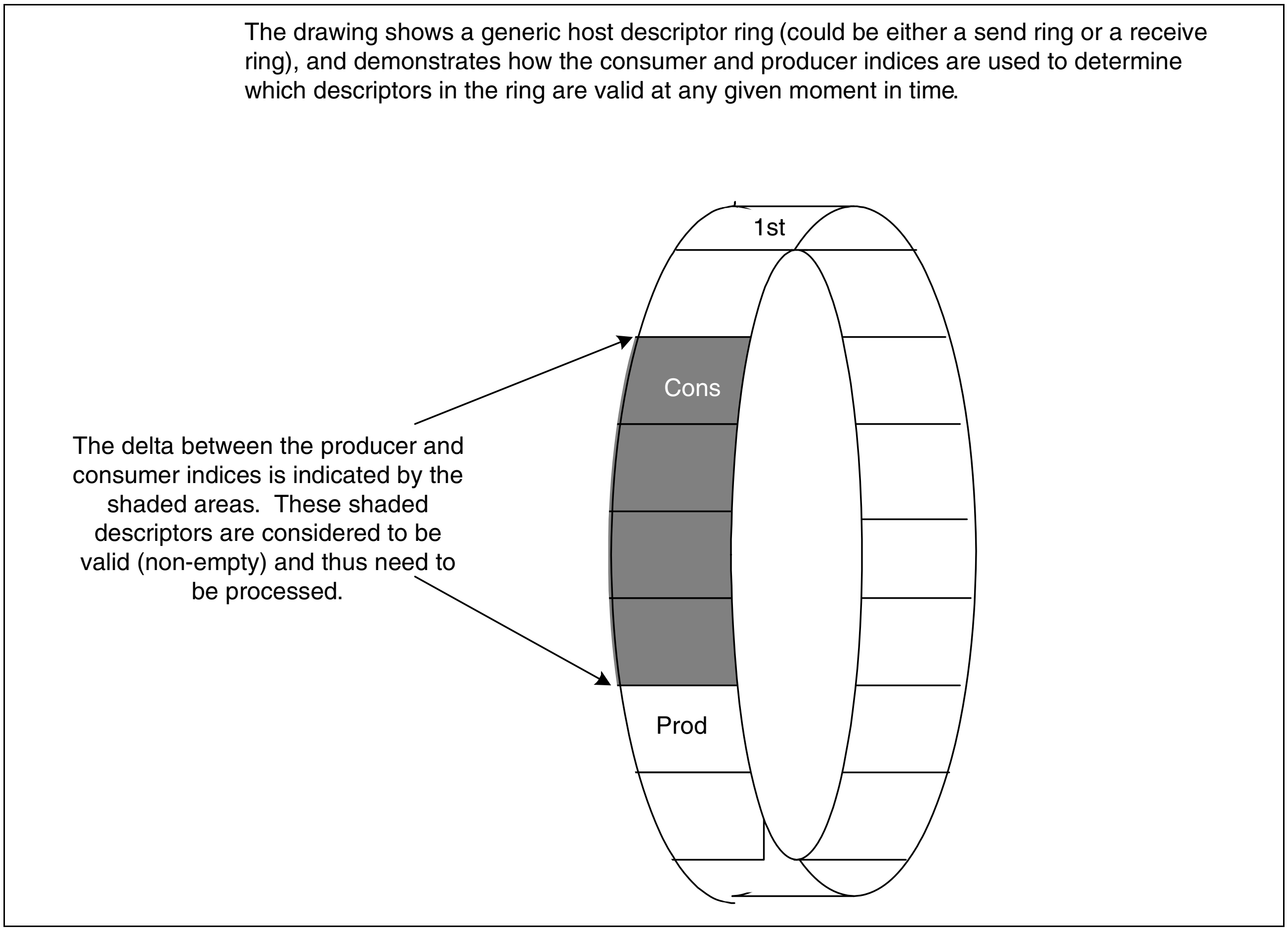
There are three main types of descriptor rings:

- Send Rings
- Receive Producer Rings
- Receive Return Rings

Producer and Consumer Indices

The Producer Index and the Consumer Index control which descriptors are valid for a given ring. Each ring has its own separate Producer and Consumer Indices. When incremented, the Producer Index can be used to add elements to the ring. Conversely, when incremented, the Consumer Index is used to remove elements from the ring. The difference between the Producer and Consumer Indices marks which descriptors are currently valid in the ring (see Figure 10). When the Producer and Consumer Index are equal, the ring is empty. When the producer is one behind the consumer, the ring is considered to be full.

Figure 10: Generic Ring Diagram



The drawing shows a generic host descriptor ring (could be either a send ring or a receive ring), and demonstrates how the consumer and producer indices are used to determine which descriptors in the ring are valid at any given moment in time.

The delta between the producer and consumer indices is indicated by the shaded areas. These shaded descriptors are considered to be valid (non-empty) and thus need to be processed.

Ring Control Blocks

Each ring (send or receive) has a Ring Control Block (RCB) associated with it. Each RCB has the format shown in Table 4.

Table 4: Ring Control Block Format

Offset (bytes)	***31***	***16***	***15***	***0***
0x00	Host Ring Address			
0x04				
0x08	Max_len		Flags	
0x0c	NIC Ring Address			

The fields are defined as follows:

- The Host Ring Address field contains the 64-bit host address of the first element in the ring. This field tells the controller where in host memory the ring is located. This field only applies to rings located in host memory. The Host Ring Address field contains the 64-bit address, in big-endian ordering, of the first Send BD in host memory.
- The Flags field contains bits flags that contain control information about a given ring. Table 5 shows the two defined flags.

Table 5: Flag Fields for a Ring

Bits	***Name***	***Description***
0	Reserved	Set to 0.
1	RCB_FLAG_RING_DISABLED	Indicates that the ring is not in use.
15:2	Reserved	Set to 0 for send rings and set to maximum receive frame size for receive rings. (See "Receive Producer Length/Flags Register (offset: 0x2458)" on page 379 for additional detail.)

- The Max_len field has a different meaning for different types of rings.
  - This field indicates the number elements in the ring.
  - The valid values for this field are 32, 64, 128, 256, and 512.
- The NIC Ring Address field contains the address where the BD cache is located in the internal NIC address space. This address is only valid for Receive Producer Rings. The Send Rings and Receive Return Rings do not require this field to be populated. The location within the NIC address map for Receive Producer Ring is provided in Section 8: "PCI," on page 160.

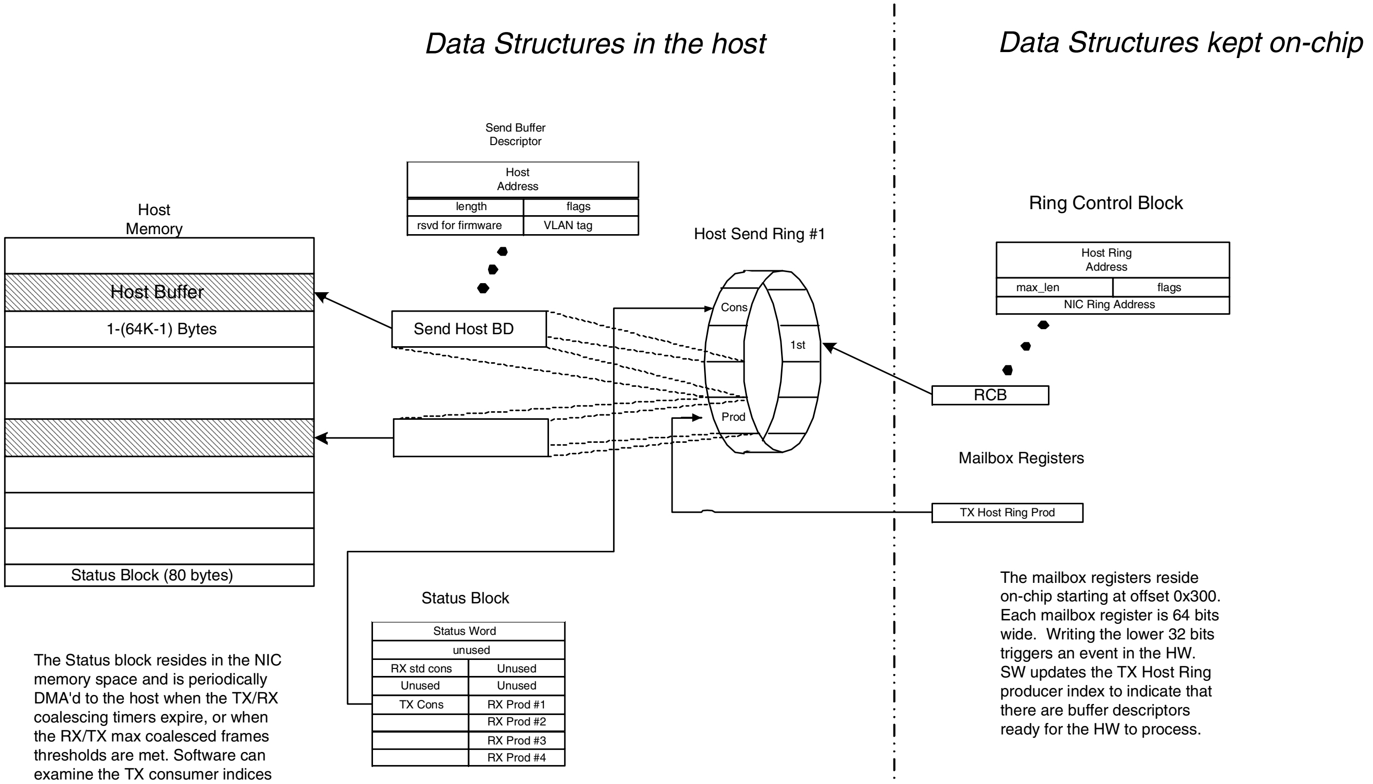
Send Rings

The controller devices covered in this document support only one host based Send Ring.

The Send Ring Producer Index is incremented by host software to add descriptors to the Send Ring (see Figure 11: “Transmit Ring Data Structure Architecture Diagram,” on page 70). By adding descriptors to the ring, the device is instructed to transmit packets that are composed of the buffers pointed to by the descriptors. A single transmit packet may be composed of multiple buffers that are pointed to by multiple send descriptors. The maximum number of send descriptors for a single packet is (0.75)*(ring size).

Figure 11: Transmit Ring Data Structure Architecture Diagram

Transmit Ring Data Scructure is located in the host (as shown below), and the device will keep a local (not shown) copy of the rings.



The Status block resides in the NIC memory space and is periodically DMA'd to the host when the TX/RX coalescing timers expire, or when the RX/TX max coalesced frames thresholds are met. Software can examine the TX consumer indices in the status block to determine which packets have been sent by the hardware.

The mailbox registers reside on-chip starting at offset 0x300. Each mailbox register is 64 bits wide. Writing the lower 32 bits triggers an event in the HW. SW updates the TX Host Ring producer index to indicate that there are buffer descriptors ready for the HW to process.

Send Buffer Descriptors

Standard (Not Large Segment Offload) Send BD

The format of an individual send buffer descriptor is shown in Table 6.

Table 6: Send Buffer Descriptors Format

Offset (Bytes)	*31 16*	*15 0*
0x00 0x04	Host Address [63:0]	
0x08	Length [15:0]	Flags [15:0]
0x0c	Reserved	VLAN Tag

The fields are defined as follows:

- The Host Address field contains the 64-bit host address of the buffer that the descriptor points to. A length of 0 indicates that the descriptor does not have a buffer associated with it.
- The Flags field contains bits flags that contain control information for the device for transmitting the packets. The defined flags are listed in Table 7.

Table 7: Defined Flags for Send Buffer Descriptors

Bits	*Name*	*Description*
0	TCP_UDP_CKSUM[a]	If set to 1, the controller replaces the TCP/UDP checksum field of TCP/UDP packets with the hardware calculated TCP/UDP checksum for the packet associated with this descriptor.
1	IP_CKSUM	If set to 1, the controller replaces the IPv4 checksum field of TCP/UDP packets over IPv4 with the hardware calculated IPv4 checksum for the packet associated with this descriptor. This bit should only be set in the descriptor that points to the buffer containing the IPv4 header. It is assumed that the IPv4 header is contained in a single buffer.
2	PACKET_END	If set to 1, the packet ends with the data in the buffer pointed to by this descriptor.
5:3	Reserved	Should be set to 0.
6	VLAN_TAG[a]	If set to 1, the device inserts an IEEE 802.1Q VLAN tag into the packet. The 16-bit TCI (Tag Control Information) field of four byte VLAN tag comes from the VLAN Tag field in the descriptor.
7	COAL_NOW	If set to 1, the device immediately updates the Send Consumer Index after the buffer associated with this descriptor has been transferred via DMA to NIC memory from host memory. An interrupt may or may not be generated according to the state of the interrupt avoidance mechanisms. If this bit is set to 0, then the Consumer Index is only updated as soon as one of the host interrupt coalescing conditions has been met.
8	CPU_PRE_DMA	If set to 1, the controller's internal CPU is required to act upon the packet before the packet is given to the internal *Send Data Initiator* state machine. Normally this bit should be set to 0.

Table 7: Defined Flags for Send Buffer Descriptors (Cont.)

Bits	Name	Description
9	CPU_POST_DMA[a]	If set to 1, the controller's internal CPU is required to act upon the packet before the packet is given to the internal *Send Data Completion* state machine. Normally this bit should be set to 0.
14:10	Reserved	Should be set to 0.
15	DON'T_GEN_CRC[a]	If set to 1, the controller will not append an Ethernet CRC to the end of the frame.

[a] Indicates that this bit should be set in all descriptors for a given packet if the desired capability is to be enabled for that packet.

Note: The UDP checksum engine does not span IP fragmented frames.

- The Length field specifies the length of the data buffer. The lengths for the buffers associated with a given packet will add up to the length of the packet.

  Note: The Ethernet controller does not validate the value of the Length field and may generate an error on the PCI bus if the Length field has a value of 0. The host driver must ensure that the Length field is nonzero before enqueueing the BD onto the Send Ring.

- The VLAN Tag field is only valid when the VLAN_TAG bit of Flags field is set. This VLAN Tag field contains the 16-bit VLAN tag that is to be inserted into an IEEE 802.1Q (and IEEE 802.3ac)-compliant packet by the controller. If VLAN tag insertion is desired, this field (and the flag) should be set in the first descriptor for that packet (i.e., the descriptor that points to the buffer that contains the Ethernet header).

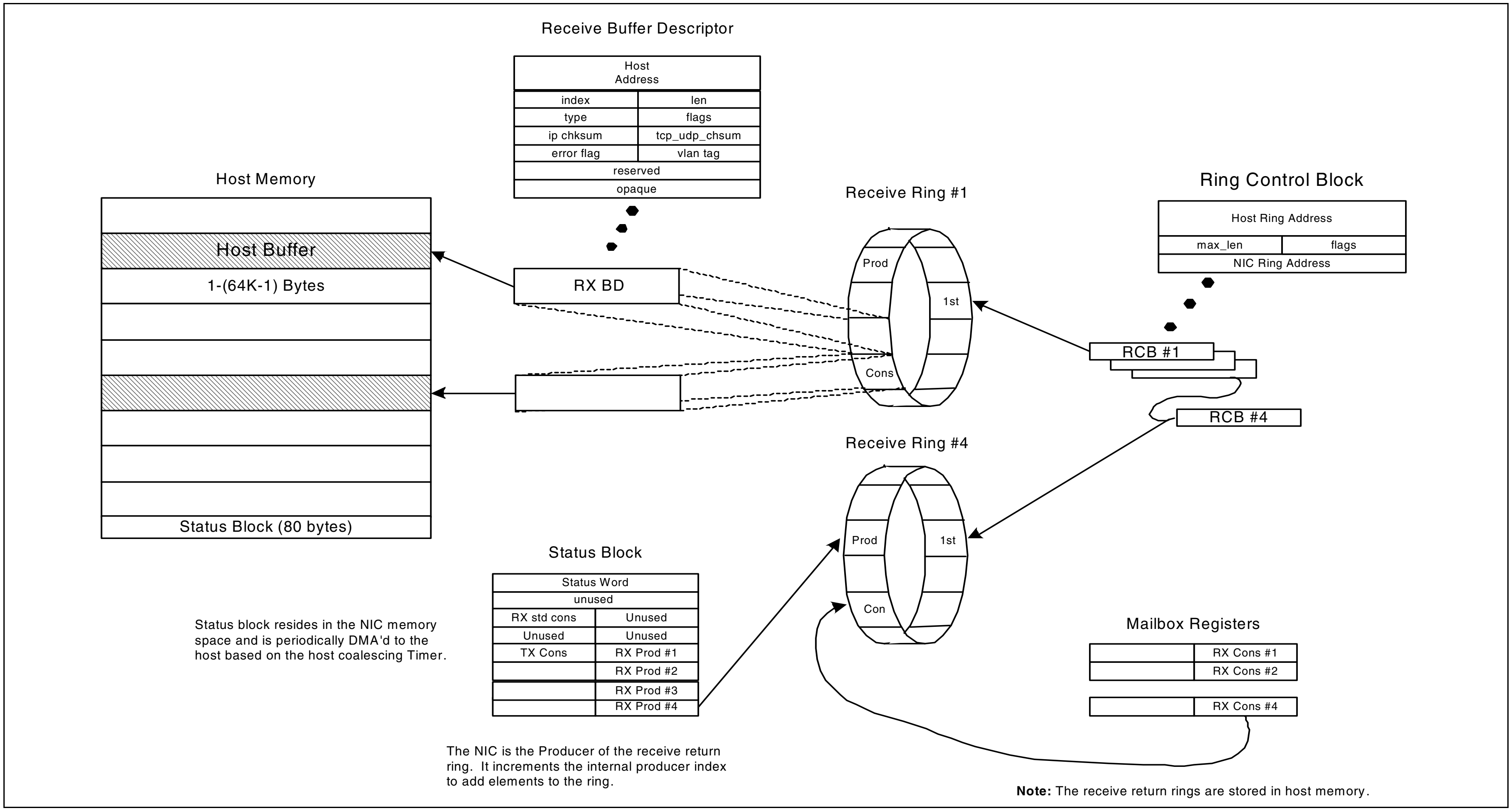
Large Segment Offload (LSO) Send BD

See "Large Segment Offload" on page 112.

Receive Rings

The Ethernet controllers support two types of Receive Descriptor Rings: Producer Rings and Return Rings (see Figure 12 on page 73). Descriptors in the Producer Rings point to free buffers in the host. When the controller receives a packet and consumes a receive buffer, the controller will modify and write back the descriptor for the consumed buffer into the given Receive Return Ring. Basically the Producer Rings contain descriptors that point to buffers that the controller is free to use, whereas the Return Rings contain descriptors that the device has used and await processing from host software.

Figure 12: Receive Return Ring Memory Architecture Diagram

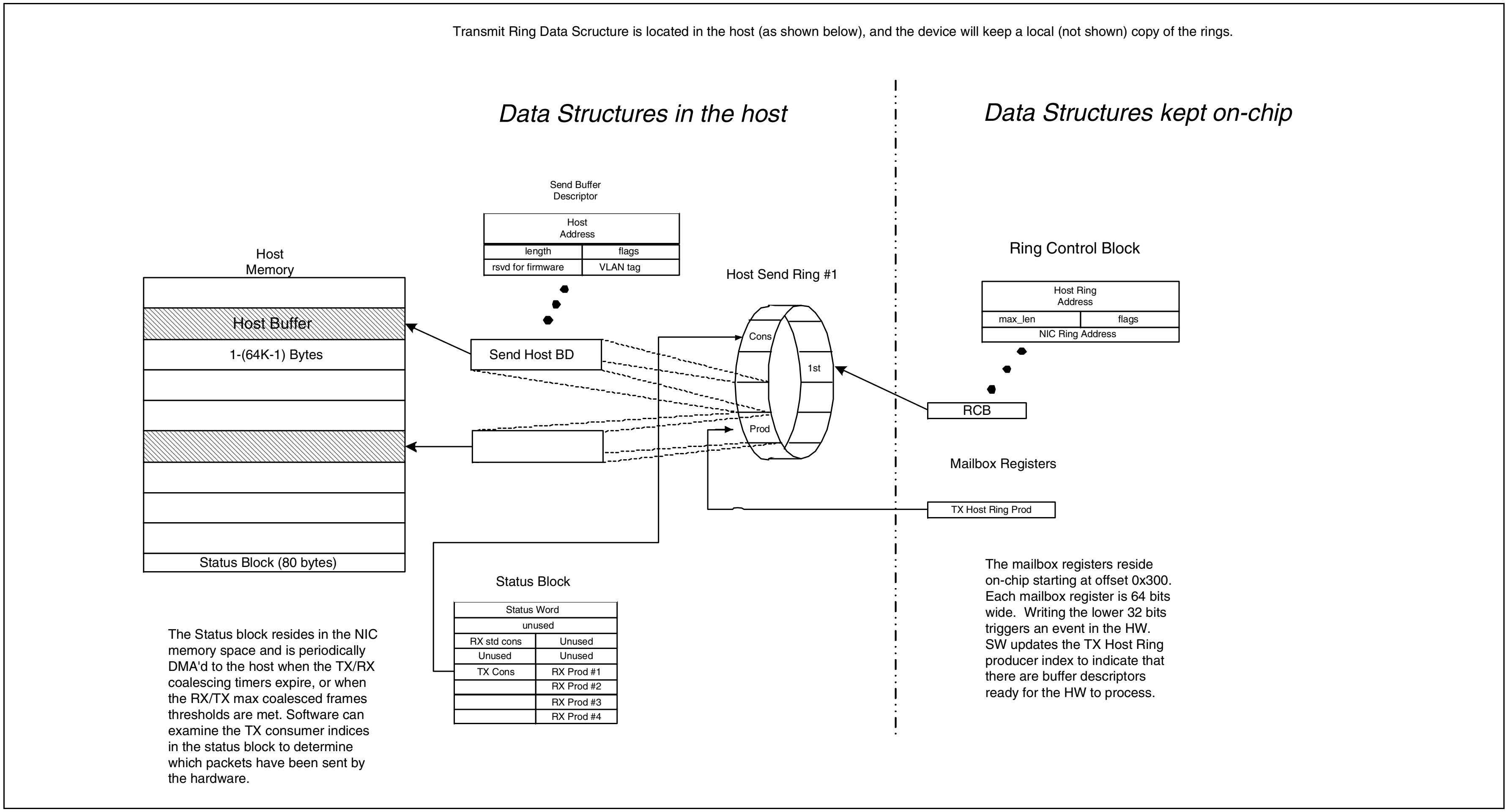


Status block resides in the NIC memory space and is periodically DMA'd to the host based on the host coalescing Timer.

The NIC is the Producer of the receive return ring. It increments the internal producer index to add elements to the ring.

Note: The receive return rings are stored in host memory.

Figure 13: Transmit Ring Data Structure Architecture Diagram



Transmit Ring Data Scructure is located in the host (as shown below), and the device will keep a local (not shown) copy of the rings.

The Status block resides in the NIC memory space and is periodically DMA'd to the host when the TX/RX coalescing timers expire, or when the RX/TX max coalesced frames thresholds are met. Software can examine the TX consumer indices in the status block to determine which packets have been sent by the hardware.

The mailbox registers reside on-chip starting at offset 0x300. Each mailbox register is 64 bits wide. Writing the lower 32 bits triggers an event in the HW. SW updates the TX Host Ring producer index to indicate that there are buffer descriptors ready for the HW to process.

Send Buffer Descriptors

Standard (Not Large Segment Offload) Send BD

The format of an individual send buffer descriptor is shown in Table 6.

Table 8: Send Buffer Descriptors Format

Offset (Bytes)	*31 16*	*15 0*
0x00	Host Address [63:0]	
0x04		
0x08	Length [15:0]	Flags [15:0]
0x0c	Reserved	VLAN Tag

The fields are defined as follows:

- The Host Address field contains the 64-bit host address of the buffer that the descriptor points to. A length of 0 indicates that the descriptor does not have a buffer associated with it.
- The Flags field contains bits flags that contain control information for the device for transmitting the packets. The defined flags are listed in Table 7.

Table 9: Defined Flags for Send Buffer Descriptors

Bits	*Name*	*Description*
0	TCP_UDP_CKSUM[a]	If set to 1, the controller replaces the TCP/UDP checksum field of TCP/UDP packets with the hardware calculated TCP/UDP checksum for the packet associated with this descriptor.
1	IP_CKSUM	If set to 1, the controller replaces the IPv4 checksum field of TCP/UDP packets over IPv4 with the hardware calculated IPv4 checksum for the packet associated with this descriptor. This bit should only be set in the descriptor that points to the buffer containing the IPv4 header. It is assumed that the IPv4 header is contained in a single buffer.
2	PACKET_END	If set to 1, the packet ends with the data in the buffer pointed to by this descriptor.
5:3	Reserved	Should be set to 0.
6	VLAN_TAG[a]	If set to 1, the device inserts an IEEE 802.1Q VLAN tag into the packet. The 16-bit TCI (Tag Control Information) field of four byte VLAN tag comes from the VLAN Tag field in the descriptor.
7	COAL_NOW	If set to 1, the device immediately updates the Send Consumer Index after the buffer associated with this descriptor has been transferred via DMA to NIC memory from host memory. An interrupt may or may not be generated according to the state of the interrupt avoidance mechanisms. If this bit is set to 0, then the Consumer Index is only updated as soon as one of the host interrupt coalescing conditions has been met.
8	CPU_PRE_DMA	If set to 1, the controller's internal CPU is required to act upon the packet before the packet is given to the internal *Send Data Initiator* state machine. Normally this bit should be set to 0.

Table 9: Defined Flags for Send Buffer Descriptors (Cont.)

Bits	*Name*	*Description*
9	CPU_POST_DMA[a]	If set to 1, the controller's internal CPU is required to act upon the packet before the packet is given to the internal *Send Data Completion* state machine. Normally this bit should be set to 0.
14:10	Reserved	Should be set to 0.
15	DON'T_GEN_CRC[a]	If set to 1, the controller will not append an Ethernet CRC to the end of the frame.

[a] Indicates that this bit should be set in all descriptors for a given packet if the desired capability is to be enabled for that packet.

Note: The UDP checksum engine does not span IP fragmented frames.

- The Length field specifies the length of the data buffer. The lengths for the buffers associated with a given packet will add up to the length of the packet.

  Note: The Ethernet controller does not validate the value of the Length field and may generate an error on the PCI bus if the Length field has a value of 0. The host driver must ensure that the Length field is nonzero before enqueueing the BD onto the Send Ring.

- The VLAN Tag field is only valid when the VLAN_TAG bit of Flags field is set. This VLAN Tag field contains the 16-bit VLAN tag that is to be inserted into an IEEE 802.1Q (and IEEE 802.3ac)-compliant packet by the controller. If VLAN tag insertion is desired, this field (and the flag) should be set in the first descriptor for that packet (i.e., the descriptor that points to the buffer that contains the Ethernet header).

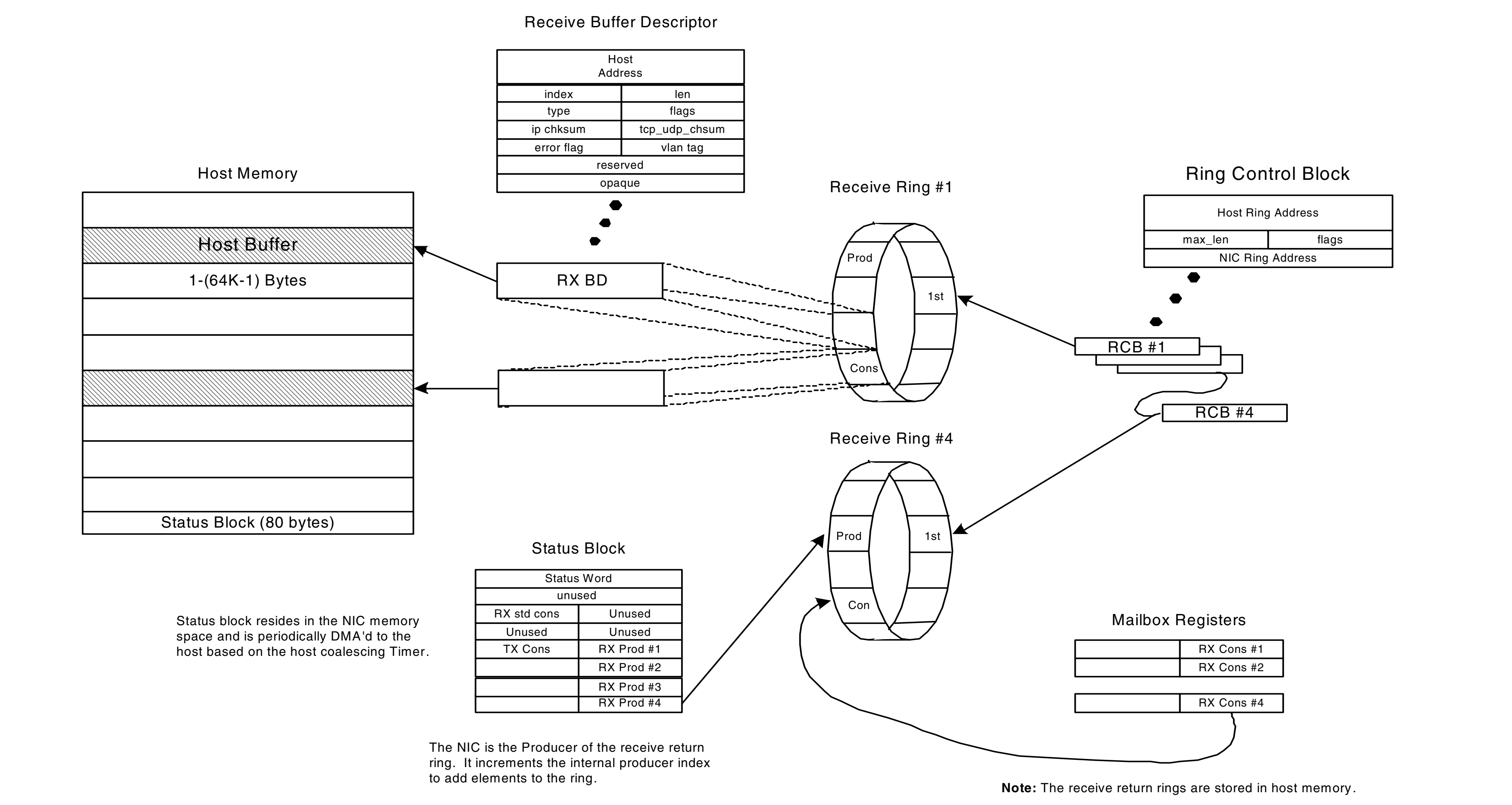
Large Segment Offload (LSO) Send BD

See "Large Segment Offload" on page 112.

Receive Rings

The Ethernet controllers support two types of Receive Descriptor Rings: Producer Rings and Return Rings (see Figure 12 on page 73). Descriptors in the Producer Rings point to free buffers in the host. When the controller receives a packet and consumes a receive buffer, the controller will modify and write back the descriptor for the consumed buffer into the given Receive Return Ring. Basically the Producer Rings contain descriptors that point to buffers that the controller is free to use, whereas the Return Rings contain descriptors that the device has used and await processing from host software.

Figure 14: Receive Return Ring Memory Architecture Diagram



Status block resides in the NIC memory space and is periodically DMA'd to the host based on the host coalescing Timer.

The NIC is the Producer of the receive return ring. It increments the internal producer index to add elements to the ring.

Note: The receive return rings are stored in host memory.

Receive Producer Ring

The receive producer ring resides in the host and points to empty host receive buffers that will later be filled with received packet data. The controller will internally cache a copy of the producer ring. When the host software driver has a free host receive packet buffer available for incoming packets, it will fill out a receive buffer descriptor and have that descriptor point to the available buffer. Host software will then update the producer index for that receive producer ring to indicate to the controller that there is a newly available receive buffer. After the controller fetches and caches (e.g., consumes) this receive producer descriptor, the controller will update the consumer index of the receive producer ring.

Receive Return Rings

When the NIC receives a packet, it will DMA that packet to a host receive packet data buffer pointed to by the available receive buffer descriptor (see Section 5: "Receive Data Flow," on page 90). Earlier the NIC will have received ownership of that data buffer via an update of the producer index of receive producer ring. After the controller does the packet data write DMA, it will DMA a corresponding buffer descriptor into the appropriate receive return ring. The buffer descriptor that is returned in the receive return ring will be slightly modified from the original buffer descriptor that the controller fetched out of the receive producer ring. After the controller has completed the DMA of the receive return ring descriptor, the controller will update its internal copy of the producer index for that particular receive return ring. That new value for that receive return ring producer index will be included in the next status block update that is made to the host. The updated value of receive return ring producer index in status block will be used by host software in determining whether new packets have been received.

Table 10: Receive Return Rings

Description	*BCM5725/BCM5762/BCM57767*
Number of Rings	4
Buffer Descriptor Size (bytes)	32
Host Ring Size (# of Buffer Descriptors)	Can be configured for 32 or 64 or 128 or 256 or 512
NIC Cache Size (# of Buffer Descriptors)	0

Receive Buffer Descriptors

The format of Standard Receive Buffer Descriptors (in both producer ring and return rings) is shown in Table 11.

Table 11: Receive Descriptors Format

Offset (bytes)	*31*	*16*	*15*	*0*
0x00	Host Address			
0x04				
0x08	Index		Length	
0x0c	Type[a]		Flags	
0x10	IP_Cksum[a]		TCP_UDP_Cksum	
0x14	Error_Flags		VLAN tag	

Table 11: Receive Descriptors Format (Cont.)

Offset (bytes)	*31*	*16*	*15*	*0*
0x18	RSS Hash[b]			
0x1C	Opaque			

a. Reserved if using EAV[c] mode

b. Receive Time Stamp if using EAV[c] mode

c. Ethernet Audio Video

The fields are defined as follows:

- Host Address — Contains the 64-bit host address of the buffer that the descriptor points to. A length of 0 indicates that the descriptor does not have a buffer associated with it.
- Length — Specifies the length of the data buffer. For Producer Rings this value is set by the host software to correspond to the size of the buffer that is available for a receive packet. Once a packet has been received, the controller will modify this length field to correspond to the length of the packet received. A value of 0 indicates that there is no valid data in the buffer.
- Index — Is set by host software in the descriptors in the producer rings. When the controller uses a given buffer descriptor, it will opaquely pass the Index field for that buffer descriptor through to the corresponding descriptor in the return ring. This index field of the BD in Return Ring is then used by the host software to associate the BD in Return Ring with the BD in Producer Ring that points to the given receive buffer.
- Flags — Contains bits flags that contain control information about a given descriptor. The defined flags are listed in Table 12.

Table 12: Defined Flags for Receive Buffers

Bits	*Name*	*Description*
15	IP Version	Indicates whether the received IP packet is an IPv6 or IPv4 packet. This bit will be 1 for IPv6 packet and 0 for IPv4 packet.
14	TCP_UDP_IS_TCP	In producer rings this bit should be set to 0. In return rings this bit will be set to 1 by the controller if the controller calculated that the incoming packet was a TCP packet. If the packet is UDP or a non IP frame, then this bit should be set to 0.
13	TCP_UDP_CHECKSUM	In producer rings this bit should be set to 0. In return rings this bit will be set to 1 by the controller if the controller calculated that the TCP or UDP checksum in the corresponding incoming packet was correct.
12	IP_CHECKSUM	In producer rings this bit should be set to 0. In return rings this bit will be set to 1 by the controller if the controller calculated that the IP checksum in the corresponding incoming packet was correct.
11	Reserved	–
10	FRAME_HAS_ERROR	If set to 1 in a return ring, it indicates that the controller detected an error. The specific type of error is specified in the Error_Flag field of the receive return descriptor.

Table 12: Defined Flags for Receive Buffers (Cont.)

Bits	Name	Description
9:7	RSS Hash Type	Indicates the hash type used in RSS hash calculation for a received packet. Available hash types are: • 0 2_TUPLE_IPV4 • 1 4_TUPLE_IPV4 • 2 2_TUPLE_IPV6 • 3 4_TUPLE_IPV6 • 4 Reserved • 5 Reserved • 6 Reserved • 7 Reserved See "Receive MAC Mode Register (offset: 0x468)" on page 332 for additional information about enabling the different RSS hash types.
6	VLAN_TAG*	If set to 1 in a return ring, it indicates that the packet associated with this buffer contained a four-byte IEEE 802.1Q VLAN tag. The 16 VLAN ID is stripped from the packet and located in the VLAN tag field in the descriptor.
5	Reserved	Should be set to 0.
4	Reserved	Should be set to 0.
3	RSS_Hash Valid	If set to 1, indicates host that the RSS_Hash in Receive BD of return ring is valid.
2	PACKET_END	If set to 1, the packet ends with the data in the buffer pointed to by this descriptor.
1:0	Reserved	Should be set to 0.

- Type — Used internally by the controller. In producer rings it should be set to 0, and in return rings it should be ignored by host software.
- TCP_UDP_Cksum — Holds the TCP/UDP checksum that the controller calculated for all data following the IP header given the length defined in the IP header. If the Receive No Pseudo-header Checksum bit is set (see "Mode Control Register (offset: 0x6800)" on page 521) to 1, then the pseudo-header checksum value is not added to this value. Otherwise, the TCP_UDP_Cksum field includes the pseudo-header in the controller's calculation of the TCP or UDP checksum. If the packet is not a TCP or UDP packet, this field has no meaning. Host software should zero this value in the producer ring descriptors. If the host is capable of TCP or UDP checksum off load, then host software may examine this field in the return rings to determine if the TCP or UDP checksum was correct.
- IP_Cksum — Holds the IPv4 checksum that the controller calculated for the IPv4 header of the received packet. If the packet is not an IPv4 packet, this field has no meaning. Host software should zero this value in the producer ring descriptors. If the host is capable of IPv4 checksum off load, then host software may examine this field in the return rings to determine if the IPv4 checksum was correct. A correct value would be 0 or 0xFFFF.
- VLAN — Only valid when the VLAN_TAG bit is set. This field contains the 16-bit VLAN ID that was extracted from an incoming packet that had an IEEE 802.1Q (and IEEE 802.3ac) -compliant header.
- Error_Flags — Contains bits flags that contain error information about an incoming packet that the descriptor is associated with. The bits in this field are only valid if the FRAME_HAS_ERROR bit is set in the Flags field in the descriptor. The defined error flags are listed in Table 13 on page 81.

Table 13: Defined Error Flags for Receive Buffers

Bits	***Name***	***Description***
31:9	Reserved	Should be set to 0.
8	GIANT_PKT_RCVD	If set to 1, the received packet was longer than the maximum packet length value set in the Receive MTU Size register (see "Receive MTU Size Register (offset: 0x43C)" on page 327). The data in the received packet was truncated at the length specified in the Receive MTU Size register.
7	TRUNC_NO_RES	If set to 1, the received packet was truncated because the controller did not have the resources to receive a packet of this length.
6	LEN_LESS_64	If set to 1, the received packet was less than the required 64 bytes in length.
5	MAC_ABORT	If set to 1, the MAC aborted due to an unspecified internal error while receiving this packet. The packet could be corrupted.
4	ODD_NIBBLE_RX_MII	If set to 1, the received packet contained an odd number of nibbles. Thus, packet data could be corrupt.
3	PHY_DECODE_ERR	If set to 1, while receiving this packet the device encountered an unspecified frame decoding error. This packet could be corrupted. This bit is set for valid packets that are received with a dribble nibble. True alignment errors will be dropped by that MAC and never show up to the driver.
2	LINK_LOST	If set to 1, link was lost while receiving this frame. Therefore, this packet is incomplete.
1	COLL_DETECT	If set to 1, a collision was encountered while receiving this packet.
0	BAD_CRC	If set to 1, the received packet has a bad Ethernet CRC.

- When the RSS Hash Valid flag bit is 1, the RSS Hash field holds the 32-bit RSS hash value calculated for a packet. This field should be ignored when the RSS Hash Valid flag bit is zero.
- The Opaque field is reserved for the host software driver. Any data placed in this field in a producer ring descriptor will be passed through unchanged to the corresponding return ring descriptor.

Status Block

The Status Block is another shared memory data structure that is located in host memory. The Status Block is 32 bytes in length. Host software will need to allocate 32 bytes of non-paged memory space for the Status Block and set the Status Block Host Address register to point to the host memory physical address reserved for this structure.

The controller will update the Status Block to host memory prior to a host coalescing interrupt or MSI. The frequency of these Status Block updates is determined by the host coalescing logic (see "Host Coalescing Engine" on page 225). Using the software configurable coalescing parameters, the device driver can optimize the frequency of status block updates for a particular application or operating system.

The Status Block contains some of the Producer and Consumer indices for the rings described in "Descriptor Rings" on page 67. These Producer and Consumer indices allow host software to know what the current status of the controller is regarding its processing of the various send and receive rings. From information in the status block a software driver can determine:

- Whether the Status Block has been recently updated (via a bit in the status word).
- Whether the Link State has changed (via a bit in the status word).
- Whether the controller has recently received a packet and deposited that packet into host memory for a given ring (via the Receive Return Ring Producer Indices).
- Which host receive descriptors that controller has fetched, and it will consume when future packets are received (via the Receive Producer Ring Consumer Indices).
- Whether the controller has recently completed a transmit descriptor buffer DMA for a given ring (via the Send Ring Consumer Indices).

Status Block Format

Note: Reference registers 0x3C50–0x3CC0 for debug access to the various ring indices.

Each MSI-X vector is associated with a status-block structure. A status block is DMAed to the host memory immediately prior to raising a legacy style interrupt (INTx, MSI) or MSI-X interrupt. Status block formats vary depending on RSS, EAV mode choices, as well as MSI-X vector number.

The various status block formats are shown in this section.

INTx/MSI — Legacy Mode Status Block Format

INTx and MSI use this status-block format in non-RSS, non-EAV mode.

Table 14: Status Block Format (MSI-X Single-Vector or INTx — RSS Mode)

Offset	*31 16*	*15 0*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0]Status Tag
0x08	Receive Standard Producer Ring Consumer Index	Reserved 0x0
0x0C	Reserved 0x0	Reserved 0x0
0x10	Send BD Consumer Index	Receive Return Ring Producer Index
0x14	Reserved 0x0	Receive Jumbo Producer Ring Consumer Index

Status-Block [0] Status Word Format (single-vector RSS):

- Bit [0]: Update bit
- Bit [1]: Link status change
- Bit [2]: Error/attention
- Bits [31:3]: Reserved 0x0

Single-Vector or INTx — RSS Mode Status Block Format

In the single-vector RSS mode, the status-block format used by Vector#0 is shown below. INTx and MSI also use this status-block format.

Table 15: Status Block Format (MSI-X Single-Vector or INTx — RSS Mode)

Offset	*31 16*	*15 0*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0]Status Tag
0x08	Receive Standard Producer Ring Consumer Index	Receive Return Ring 1 Producer Index
0x0C	Receive Return Ring 2 Producer Index	Receive Return Ring 3 Producer Index
0x10	Send BD Consumer Index	Receive Return Ring 0 Producer Index
0x14	Reserved 0x0	Receive Jumbo Producer Ring Consumer Index

Status-Block [0] Status Word Format (single-vector RSS):

- Bit [0]: Update bit
- Bit [1]: Link status change
- Bit [2]: Error/attention
- Bits [31:3]: Reserved 0x0

Single-Vector or INTx — EAV Mode Status Block Format

In Single-vector EAV mode, all events are reported in the same status block used by Vector#0. INTx and MSI also use this status-block format.

Table 16: Status Block Format (MSI-X Single-Vector or INTx — EAV Mode)

Offset	*31 16*	*15 0*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0]Status Tag
0x08	Receive Standard Producer Ring Consumer Index	Receive Return Ring 1 Producer Index
0x0C	Receive Return Ring 2 Producer Index	Receive Return Ring 3 Producer Index
0x10	Send BD Ring 1 Consumer Index	Receive Return Ring 0 Producer Index
0x14	Send BD Ring 2 Consumer Index	Receive Jumbo Producer Ring Consumer Index

Status-Block Status Word Format (single-vector EAV):

- Bit [0]: Update bit
- Bit [1]: Link status change
- Bit [2]: Error/attention
- Bits [31:3]: Reserved — always 0x0

Multivector RSS Mode Status Block Format

There are five slightly different status-block formats used by the multivector RSS mode. Each of these formats associate with their respective vector numbers as shown in the tables below.

Table 17: Status Block [0] Format (MSI-X Multivector RSS Mode)

<table>
<tr><th>Offset</th><th>31 16</th><th colspan="2">15 0</th></tr>
<tr><td>0x00</td><td colspan="3">Status Word</td></tr>
<tr><td>0x04</td><td colspan="2">[31:8] Reserved 0x0</td><td>[7:0]Status Tag</td></tr>
<tr><td>0x08</td><td>Receive Standard Producer Ring Consumer Index</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x0C</td><td>Reserved 0x0</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x10</td><td>Send BD Consumer Index</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x14</td><td>Reserved 0x0</td><td colspan="2">Receive Jumbo Producer Ring Consumer Index</td></tr>
</table>

Status-Block [0] Status Word Format (multivector RSS):

- Bit [0]: Update bit
- Bit [1]: Link status change
- Bit [2]: Error/attention
- Bits [3]: Reserved — always 0
- Bits [4]: Reserved — always 0
- Bits [5]: Reserved — always 0
- Bits [31:6]: Reserved 0x0

Table 18: Status Blocks [1 thru 4] Formats (MSI-X Multivector RSS Mode)

<table>
<tr><th>Offset</th><th>31 16</th><th colspan="2">15 0</th></tr>
<tr><td>0x00</td><td colspan="3">Status Word {Valid for all Status Blocks}</td></tr>
<tr><td>0x04</td><td colspan="2">[31:8] Reserved 0x0</td><td>[7:0] Status Tag[n] {independent for each status blocks}</td></tr>
<tr><td>0x08</td><td>Reserved 0x0</td><td colspan="2">Receive Return Ring 1 Producer Index
Valid only for Status Block2 else RSVD 0x0</td></tr>
<tr><td>0x0C</td><td>Receive Return Ring 2 Producer Index
Valid only for Status Block3 else RSVD 0x0</td><td colspan="2">Receive Return Ring 3 Producer Index
Valid only for Status Block4 else RSVD 0x0</td></tr>
<tr><td>0x10</td><td>Reserved 0x0</td><td colspan="2">Receive Return Ring 0 Producer Index
Valid only for Status Block1 else RSVD 0x0</td></tr>
<tr><td>0x14</td><td>Reserved 0x0</td><td colspan="2">Reserved 0x0</td></tr>
</table>

Status-Block [1–4] Status Word Format (multivector RSS):

- Bit [0]: Update bit
- Bit [31:1]: Reserved 0x0

Multivector EAV Mode Status Block Format

There are six slightly varying status-block formats used in this mode, one format for Status Block#0 and the other for Status Block#1 through Status Block#5 as shown in the tables below.

Table 19: Status Block [0] Format (MSI-X Multivector EAV Mode)

<table>
<tr><th>Offset</th><th>31 16</th><th colspan="2">15 0</th></tr>
<tr><td>0x00</td><td colspan="3">Status Word</td></tr>
<tr><td>0x04</td><td colspan="2">[31:8] Reserved 0x0</td><td>[7:0]Status Tag</td></tr>
<tr><td>0x08</td><td>RBD Standard Producer Ring Consumer Index</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x0C</td><td>Reserved 0x0</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x10</td><td>Send BD Ring 1 Consumer Index</td><td colspan="2">Reserved 0x0</td></tr>
<tr><td>0x14</td><td>Reserved 0x0</td><td colspan="2">Receive Jumbo Producer Ring Consumer Index</td></tr>
</table>

Status-Block [0] Status Word Format (multivector EAV):

- Bit [0]: Update bit
- Bit [1]: Link status change
- Bit [2]: Error/attention
- Bits [31:3]: Reserved — always 0x0

Table 20: Status Block [1 thru 4] Format (MSI-X Multivector EAV Mode)

<table>
<tr><th>Offset</th><th>31 16</th><th colspan="2">15 0</th></tr>
<tr><td>0x00</td><td colspan="3">Status Word {Valid for all Status Blocks}</td></tr>
<tr><td>0x04</td><td colspan="2">[31:8] Reserved 0x0</td><td>[7:0] Status Tag[n] {independent for each status blocks}</td></tr>
<tr><td>0x08</td><td>Reserved 0x0</td><td colspan="2">Receive Return Ring 1 Producer Index
Valid only for Status Block2 else RSVD 0x0</td></tr>
<tr><td>0x0C</td><td>Receive Return Ring 2 Producer Index
Valid only for Status Block3 else RSVD 0x0</td><td colspan="2">Receive Return Ring 3 Producer Index
Valid only for Status Block4 else RSVD 0x0</td></tr>
<tr><td>0x10</td><td>Reserved 0x0</td><td colspan="2">Receive Return Ring 0 Producer Index
Valid only for Status Block1 else RSVD 0x0</td></tr>
<tr><td>0x14</td><td>Reserved 0x0</td><td colspan="2">Reserved 0x0</td></tr>
</table>

Status-Block [1–16] Status Word Format (multivector EAV):

- Bit [0]: Update bit
- Bits [31:1]: Reserved — always 0x0

Table 21: Status Block [5] Format (MSI-X multivector EAV Mode)

Offset	*31 16*	*15 0*	
0x00	Status Word		
0x04	[31:8] Reserved 0x0		[7:0]Status Tag
0x08	Reserved 0x0	Reserved 0x0	
0x0C	Reserved 0x0	Reserved 0x0	
0x10	Send BD Ring 2 Consumer Index	Reserved 0x0	
0x14	Reserved 0x0	Reserved 0x0	

Status-Block [5] Status Word Format (multivector EAV):

- Bit [0]: Update bit
- Bits [31:1]: Reserved — always 0x0

Status Block and INT MailBox Addresses

Each status block may be placed in an independent host memory address (64-bit). Each vector may be acknowledged via associated INT MailBoxes.

Table 22: Status Block Host Addresses and INT MailBox Addresses

		EAV Mode		*RSS Mode*		
Status Block #	*Status Block Host Address Register (64-Bit)*	*INT MailBox Register Address*	*Indication Items*	*INT MailBox Register Address*	*Indication Items*	*Comments*
Legacy	0x3C3C, 0x3C38	0x200 (64-bit)	All	0x200	All	Legacy status block Used by INTx or MSI
0	x3C3C, 0x3C38	0x200	Link-Status change Error/Attention SBD Ring 1 Cons Index Std RBD Cons Index Jmb RBD Cons Index	0x200	Link-Status change Error/Attention SBD Ring 1 Cons Index Std RBD Cons Index Jmb RBD Cons Index	Used in all MSI-X modes for Vector#0

Table 22: Status Block Host Addresses and INT MailBox Addresses (Cont.)

		EAV Mode		RSS Mode		
Status Block #	Status Block Host Address Register (64-Bit)	INT MailBox Register Address	Indication Items	INT MailBox Register Address	Indication Items	Comments
1	0x3D00, 0x3D04	0x208	RX Return Ring 0 Prod Index	0x208	RX Return Ring 0 Prod Index	Used only in MSI-X multivector RSS mode or multivector EAV mode for Vector#1–Vector#4
2	0x3D08, 0x3D0C	0x210	RX Return Ring 1 Prod Index	0x210	RX Return Ring 1 Prod Index	
3	0x3D10, 0x3D14	0x218	RX Return Ring 2 Prod Index	0x218	RX Return Ring 2 Prod Index	
4	0x3D18, 0x3D1C	0x220	RX Return Ring 3 Prod Index	0x220	RX Return Ring 3 Prod Index	
5	0x3D20, 0x3D24	0x228	SBD Ring 2 Cons Index	N/A	N/A	Used only in MSI-X multivector EAV mode for Vector#5

The Status word field contains bit flags that contain error information about the status of the controller. The defined flags are listed in Table 23 on page 88.

Table 23: Status Word Flags

Bits	Name	Description
0	Updated	This bit is always set to 1 each time the status block is updated in the host via DMA. It is expected that host software clear this bit in the status block each time it examines the status block. This provides the host driver with a mechanism to determine whether the status block has been updated since the last time the driver looked at the status block.
1	Link State Changed	Indicates that link status has changed. This method of determining link change status provides a small performance increase over doing a PIO read of the Ethernet MAC Status register (see "EMAC Status Register (offset: 0x404)" on page 322. See "Wake on LAN Mode/Low-Power" on page 202 for a description of the PHY setup required when link state changes.

Table 23: Status Word Flags (Cont.)

Bits	Name	Description
2	Error	When this bit is asserted by the chip, the following conditions may have occurred. Bit 2 of the status word is the OR of: • All bits in Flow Attention register (0x3c48) (see "Flow Attention Register (offset: 0x3C48)" on page 448. • MAC_ATTN—Events from the MAC block (see "EMAC Status Register (offset: 0x404)" on page 322. • DMA_EVENT—Events from the following blocks: – MSI (see "MSI Status Register (offset: 0x6004)" on page 520. – DMA_RD (see "Read DMA Status Register (offset: 0x4804)" on page 471. – DMA_WR (see "Write DMA Status Register (offset: 0x4C04)" on page 505. • RXCP_ATTN—Events from RX RISC (see "RX RISC Status Register (offset: 0x5004)" on page 506.

The Status Block format for these devices is as follows:

- Status Tag — Contains an unique eight-bit tag value in bits 7:0 when the Status Tagged Status mode bit of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256) is set to 1. This Status Tag can be returned to the Mailbox 0 register at location 31:24 (see"Interrupt Mailbox 0 (High Priority Mailbox) Register (offset: 0x200-207)" on page 319) by host driver. When the remaining Mailbox 0 register bits 23:0 are written as 0, the tag field of the Mailbox 0 register is compared with the tag field of the last status block to be DMAed to host. If the tag returned is not equivalent to the tag of the last status block DMAed to the host, the interrupt state is entered.
- Receive Producer Ring Consumer Index — Contains the controller's current Consumer Index value for the Receive Producer Ring. This field indicates how many receive descriptors are in the receive producer ring that the controller has consumed. For more information regarding this ring, see "Receive Producer Ring" on page 78.
- Receive Return Rings 0–3 Producer Indices — Contain controller's current Producer Index value for the each of the Receive Return Rings. When the controller receives a packet and writes that packet data into host memory via DMA, it will increment the Producer Index for the corresponding Receive Return ring. When a Producer Index is incremented, it is a signal to software that a newly arrived receive packet is ready to be processed.
- Send Ring Consumer Index — Contains controller's current Consumer Index value for the Send Ring. When the controller completes the read DMA of the host buffer associated with a send BD, the controller will update the Send Ring Consumer Index. This provides the host software with an indication that the controller has buffered this send data and, therefore, the host software may free the buffer that was just consumed by the device.

Section 5: Receive Data Flow

Introduction

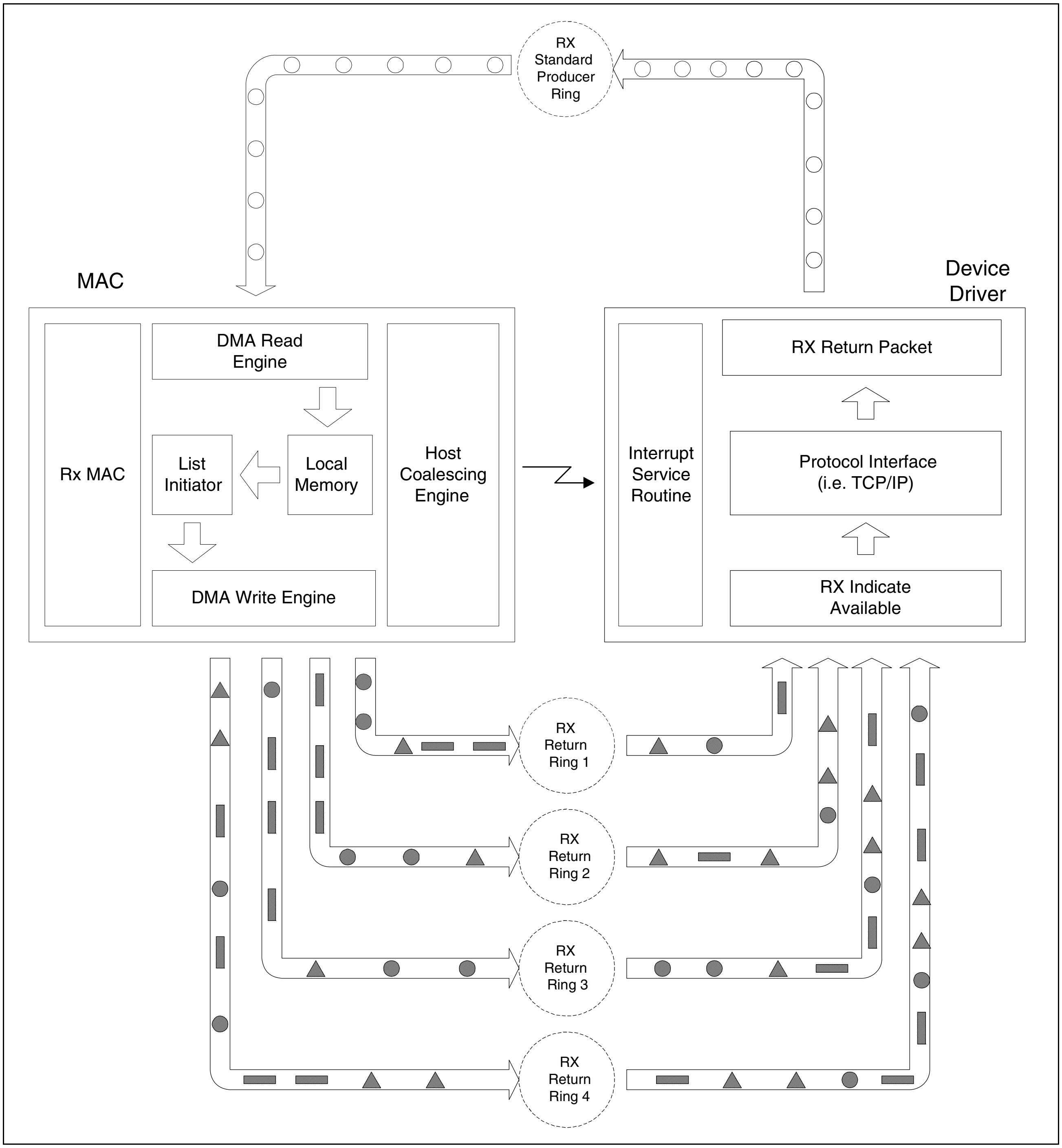
The RX MAC pulls BDs from RX producer rings. The RX BD specifies the location(s) in host memory where packet data may be written. Figure 15 on page 91 shows the receive buffer descriptor cycle.

All ingress Ethernet frames are classified by the RX rules engine. A class ID is associated to each frame based on QOS rules setup in the RX MAC (see "Receive Rules Setup and Frame Classification" on page 97). The Receive List Placement and Receive List Initiator portions of the MAC architecture move BDs to the RX return rings; the class ID associated to the packet is examined to route the BD to a specific RX return ring.

Once the packet is queued to the RX return ring, the device driver will wait for indication of the same through the status block update and interrupt from the host coalescing engine. The host coalescing engine will update the status block and generate a line interrupt or MSI (see "Host Coalescing" on page 219 for further details regarding interrupts) when a specified host coalescence criteria is met. Once the interrupt is generated, the host device driver will service the interrupt. The ISR will determine if new BDs have been completed on the RX Return Rings. Next, the device driver will indicate to the network protocol that the completed RX packets are available. The network protocol will consume the packets and return physical buffers to the network driver at a later point.

The BDs may then be reused for new RX frames. The device driver must return the BD to an RX producer ring. For this purpose, the driver should fill out either the opaque field or index field of the RX BD when inserting/ initializing the BD in an RX Producer ring. When the BD is returned by the device through Return Ring, the opaque or index data field of the BD will be used by the driver to identify the BD in Producer Ring that corresponds to the Returned BD in Return Ring. The device driver will then reinitialize the identified BD in Producer Ring with a new allocated buffer and replenish the Receive Producer Ring with this BD.

Figure 15: Receive Buffer Descriptor Cycle



Receive Producer Ring

A Receive Producer Ring is an array containing a series of Receive Buffer Descriptors (BD). The Receive Producer Ring is host-based and 25% of the available buffer descriptors are cached in Ethernet controller internal memory.

A receive producer ring contains a series of buffer descriptors which in turn contain information of host memory locations to where packets are placed by the Ethernet controller at reception. The limit on the number of buffer descriptors in receive producer ring is 512.

Setup of Producer Rings Using RCBs

A Ring Control Block (RCB) is used by the host software to set up the shared rings in host memory. In the context of producer ring, the Receive Producer Ring RCB is a register that is used to define the Receive Producer Ring. The host software must initialize the Receive Producer Ring RCB.

Receive Producer Ring RCB — Register Offset 0x2450–0x245f

Other Considerations Relating to Producer Ring Setup

Other registers that affect the producer rings must be initialized by the host software. These registers include the Receive BD Ring Replenish Threshold register, the Receive MTU register, and the Accept Oversized bit (bit 5) in the Receive MAC Mode register.

- Receive BD Producer Ring Replenish Threshold registers:
  - "Standard Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C18)" on page 382
  - "Receive Data Completion Control Registers" on page 381.

  These registers are used for setting the number of BDs that the Ethernet controller can use up before requesting that more BDs be DMAed from a producer ring. In other words, the device does not initiate a DMA for fetching the RX BDs until the number of BDs made available to the device by the host is at least the value programmed in this register.
- Receive MTU register ("Receive MTU Size Register (offset: 0x43C)" on page 327). This 32-bit register is set to a value that is the maximum size of a packet that the Ethernet controller receives. Any packets above this size is labeled as an oversized packet. The value for this register is typically set to 1518, which is the Standard Producer Ring RCB typical setting. If Jumbo frames are supported, the MTU would be set to the maximum Jumbo frame size.
- Receive MAC Mode register ("Receive MAC Mode Register (offset: 0x468)" on page 332). If the Accept Oversized bit (bit 5) of this register is set, the Ethernet controller accepts packets (of size up to 64 KB) larger than the size specified in the MTU.

RCB Setup Pseudo Code

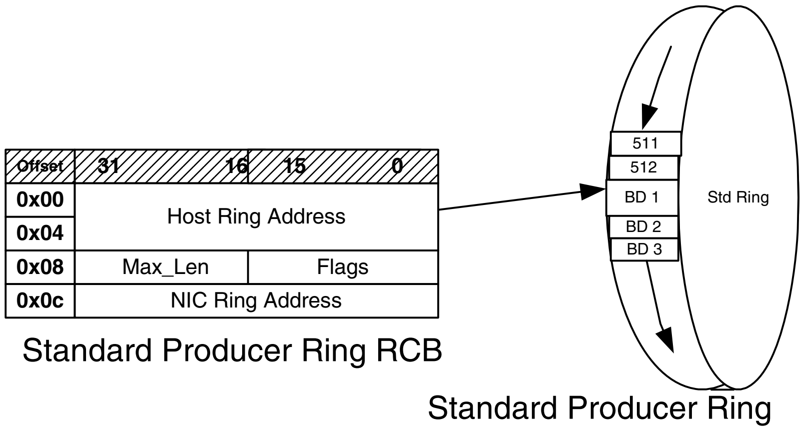
An example of setting up receive producer ring RCB:

```
Content of Pointer_to_RX_PRODUCER_RING_RCB + 0x00 = Host address of standard receive producer ring
high 32.
Content of Pointer_to_RX_PRODUCER_RING_RCB + 0x04 = Host address of standard receive producer ring
low 32.
Content of Pointer_to_RX_PRODUCER_RING_RCB + 0x0a = No flags.
Content of Pointer_to_RX_PRODUCER_RING_RCB + 0x08 = Max packet size of 1518.
Content of Pointer_to_RX_PRODUCER_RING_RCB + 0x0c = Internal Memory address for device copy of ring.
```

Figure 16 shows the standard ring RCB for the setup of a host-based standard producer ring.

Receive Buffer Descriptors (BDs) begin on the Receive Producer Ring. The host device driver will populate the receive producer ring with a specified number of BDs supported by the receive producer ring (see “Receive Producer Ring” on page 92). When a packet is received, the RX MAC moves the packet data into internal memory. The Receive MTU Size register (see “Receive MTU Size Register (offset: 0x43C)” on page 327) specifies the largest packet accepted by the RX MAC; packets larger than the Receive MTU are marked oversized and are discarded.

Figure 16: Receive Producer Ring RCB Setup



Offset	31	16	15	0
0x00	Host Ring Address			
0x04				
0x08	Max_Len		Flags	
0x0c	NIC Ring Address			

Standard Producer Ring RCB

Receive Buffer Descriptors

The Receive Buffer Descriptor is a data structure in host memory. It is the basic element that makes up each receive producer and receive return ring. The format of receive buffer descriptors can be seen in Table 11: “Receive Descriptors Format,” on page 78. A receive buffer descriptor has a 64-bit memory address and may be in any memory alignment and may point to any byte boundary. For performance and CPU efficiency reasons, it is recommended that memory be cache-aligned. The Ethernet controller supports cache line sizes of 8, 16, 32, 64, 128, 256, and 512 bytes. The cache line size value is important for the controller to determine when to use the PCI memory write and invalidate command. There are no requirements for memory alignment or cache line integrity for the Ethernet controller.

Unlike send buffer descriptors, the receive buffer descriptors cannot be chained to support multiple fragments.

Management of RX Producer Rings with Mailbox Registers and Status Block

Status Block

The host software manages the producer rings through the Mailbox registers and by using the status block. It does this by writing to the Mail Box registers when a BD is available to DMA to the Ethernet controller and reading the status block to see how many BDs have been consumed by the Ethernet controller. The status block can be seen in "Status Block" on page 82.

The status block is controlled and updated by the Ethernet controller. The status block in host memory is constantly updated through a DMA copy by the Ethernet controller from an internal status block. The updates occur at specific intervals and host coalescence conditions that are specified by host software during initialization of the Ethernet controller. The registers for setting the intervals and conditions are in the Host Coalescing Control registers (see "Host Coalescing" on page 219) starting at memory offset 0x3c00. The Ethernet controller DMAs an updated status block to the 32-bit address that is set by the host software in the Host Coalescing Control registers, 0x3c38.

Among other status, the status block displays the last 16-bit value, BD index that was DMAed to the Ethernet controller from receive producer ring. The Ethernet controller updates these indices as the recipient or consumer of the BD from the producer rings.

Mailbox

The host software is responsible for writing to the Mailbox registers (see Table 24: "Mailbox Registers," on page 94) when a BD is available from the producer rings for use by the Ethernet controller. Host software should use the high-priority mailbox region from 0x200– 0x3FF for host standard and the low-priority mailbox region from 0x5800–0x59FF for indirect register access mode.

The Mailbox registers (starting at memory offset 0x200 for host standard and offset 0x5800 for indirect mode) contain the following receive producer index register.

Receive BD Producer Ring Producer Index

- Host standard: memory offset 0x268–0x26F
- Indirect mode: memory offset 0x5868–0x586F

Table 24: Mailbox Registers

Offset (High-Priority Mailboxes for Host Standard Mode)	*Offset (Low-Priority Mailboxes for Indirect Mode)*	*Register*	*Access*
0x200 - 0x207	0x5800 - 0x5807	Interrupt Mailbox 0	RW
0x208 - 0x20F	0x5808 - 0x580F	Interrupt Mailbox 1	RW
0x268 - 0x26F	0x5868 - 0x586F	Receive BD Standard Producer Ring Producer Index	RW
0x280 - 0x287	0x5880 - 0x5887	Receive BD Return Ring 1 Consumer Index	RW

Table 24: Mailbox Registers

Offset (High-Priority Mailboxes for Host Standard Mode)	*Offset (Low-Priority Mailboxes for Indirect Mode)*	*Register*	*Access*
0x288 - 0x28F	0x5888 - 0x588F	Receive BD Return Ring 2 Consumer Index	RW
0x290 - 0x297	0x5890 - 0x5897	Receive BD Return Ring Consumer Index	RW

The Receive Producer Ring Producer Index register contains the index value of the next buffer descriptor from the producer ring that is available for DMA to the Ethernet controller from the host. When the host software updates the Receive Producer Ring Producer Index, the Ethernet controller is automatically signaled that a new BD is waiting for DMA. At initialization time, these values must be initialized to zero. These indices are 64-bit wide; however, the highest index value is only 512 for the receive Producer Ring.

Receive Return Rings

Receive Return Rings (RR) are host-based memory blocks that are used by host software to keep track of the where the Ethernet controller is putting the received packets related receive buffer descriptors. Unlike the producer rings, the return rings reside only in host memory. The Ethernet controller uses the BDs in the NIC memory that are previously copied from the producer rings to use when packets are received from the LAN. It places the BDs that correspond to received packets in the return rings.

Return rings are the exact opposite of producer rings, except that they are not categorized by the maximum length receive packets supported. They are actually categorized by priority or class of received packet. The highest priority return ring is ring 1, and the lowest priority is the last ring (Return Ring 2–Return Ring 4 depending on how many rings are set up by the host software). The Receive Return Ring is configurable to a value of either 32, 64, 128, 256, or 512.

The Receive Return Ring RCBs are used to set up return rings in much the same way the Receive Producer Ring RCB is used to set up the receive producer ring. These RCBs for the return rings are set in the Miscellaneous memory region (SSRAM) at offset 0x200 (this region should not be confused with the register space in the chip). The RCB max_len field is used to indicate the number of buffer descriptor entries in a return ring. If an invalid value is set, the Ethernet controller indicates an attention error in the Flow Attention register.

Management of Return Rings with Mailbox Registers and Status Block

The return rings are managed by the host using the Mailbox registers and status block.

When a packet is received from the LAN, the Ethernet controller DMAs the packet to a location in the host, and then DMAs the related BD to a return ring. As the producer of this packet to the host, the Ethernet controller updates the status block producer indices for the related return ring (i.e., return ring 1 to return ring 4 that was DMAed the BD received packet). These return ring indices can then be read by the host software to determine the last BD index value of a particular ring that has information of the last received packet.

As the consumer of the received packet, the host software must update the return ring consumer indices in Mailbox registers Receive BD Return Ring 1 Consumer Index (memory offset 0x280–0x287 for host standard and 0x5880–0x5887 for indirect mode) through Receive BD Return Ring 4 Consumer Index.

Host Buffer Allocation

The allocation of memory in the host is dependent on the operating system in which the controller is being used. There are two crucial items:

- The use of non-cached and physically contiguous memory is best for adapter performance.
- Physical memory mapping is required for the controller’s internal copies of logical host memory.

Receive Rules Setup and Frame Classification

The Ethernet controller has a feature that allows for the classification of receive packets based on a set of rules. The rules are determined by the host software and then input into the Ethernet controller.

A packet can be accepted or rejected based on the rules initialized into two rules register areas. The packets can also be classified into groups of packets of higher to lower priority using the rules registers. This occurs when the packet is directed to a specific return ring. Return rings 1–4 have an inherent priority associated with them. The priority is from lowest ring number to highest ring number; return ring 1 being the highest priority ring and return ring 4 being the lowest. The implementation of priority class is based on how many rings the host software has initialized and made available to the Ethernet controller. As packets arrive, the Ethernet controller may classify each packet based on the rules. When the host services the receive packet, it can service the lower numbered rings first.

A rule can be changed by first disabling it by setting 0 into Enable bit (bit 31) in Receive BD Rules Control register (see Table 26). Wait about 20 receive clocks (rx_clock) and then reenable it when it is programmed with a new rule. Otherwise, changing the rules dynamically during runtime may cause the rule checker to output erroneous results because the rule checker is a pipelined design and uses the various fields of the rules at different clock cycles.

Receive Rules Configuration Register

The Receive Rules Configuration register (memory offset 0x500–0x503, see Table 25) uses bits 3:7 to specify the ring where a received packet should be placed into if no rules are met, or if the rules have not been set up. A value of 0 means the received packet will be discarded. A value of 1–16 specifies a corresponding ring. This ring should be initialized to at least a value of 1 if the rules are not being used to ensure that all received packets will be DMAed to return ring 1.

Table 25: Receive Rules Configuration Register

Bits	***Field***	***Access***
31:8	Reserved	RO
7:3	Specifies the default class (ring) if no rules are matched	R/W
2:0	Reserved	RO

Receive List Placement Rules Array

The Receive List Placement Rules Array (memory offset 0x480–0x4ff) is made up of 16 combined element registers. The combined element is actually two 32-bit registers called the Receive BD Rules Control register (see Table 26) and the Receive BD Rules Value/Mask register (see Table 27). The element can be looked at as a single 64-bit entity with a Control part and Value/Mask part since they use a single element. Bit 26 of the control part determines how the value/mask part is used. The Receive BD Rules Value/Mask register can be used as either a 32-bit left-justified Value or a 16-bit Mask followed by a 16-bit Value.

Note: Receive rules cannot be used to match VLAN headers because the VLAN tag is stripped from the Ethernet frame before the rule checker runs.

Table 26: Receive BD Rules Control Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
E	&	P1	P2	P3	M	D	Map	Reserved						Op	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Header			Class					Offset							

Bit	*Name*	*R/W*	*Description*	*Default*
31	E	R/W	Enable. Enabled if set to 1	–
30	&	R/W	And With Next. This rule and next must both be true to match. The class fields must be the same. A disabled next rule is considered true. Processor activation bits are specified in the first rule in a series.	–
29	P1	R/W	If the rule matches, the processor is activated in the queue descriptor for the Receive List Placement state machine.	–
28	P2	R/W	If the rule matches, the processor is activated in the queue descriptor for the Receive Data and Receive BD Initiator state machine.	–
27	P3	R/W	If the rule matches, the processor is activated in the queue descriptor for the Receive Data Completion state machine.	–
26	M	R/W	Mask If set, specifies that the value/mask field is split into a 16-bit value and 16-bit mask instead of a 32-bit value.	–
25	D	R/W	Discard Frame if it matches the rule.	–
24	Map	R/W	Map Use the masked value and map it to the class.	–
23:18	Reserved	R/W	Must be set to zero.	0
17:16	Op	R/W	Comparison Operator specifies how to determine the match: • 00 = Equal • 01 = Not Equal • 10 = Greater than • 11 = Less Than	–

Bit	Name	R/W	Description	Default
15:13	Header	R/W	Header Type specifies which header the offset is for: • 000: Start of Frame (always valid) • 001: Start of IP Header (if present) • 010: Start of TCP Header (if present) • 011: Start of UDP Header (if present) • 100: Start of Data (always valid, context sensitive) • 101–111: Reserved	–
12:8	Class	R/W	The class this frame is placed into if the rule matches. 0:4, where 0 means discard. The number of valid classes is the Number of Active Queues divided by the Number of Interrupt Distribution Groups. Ring 1 has the highest priority and Ring 4 has the lowest priority.	–
7:0	Offset	R/W	Number of bytes offset specified by the header type.	–

Table 27: Receive BD Rules Value/Mask Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Mask															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value															

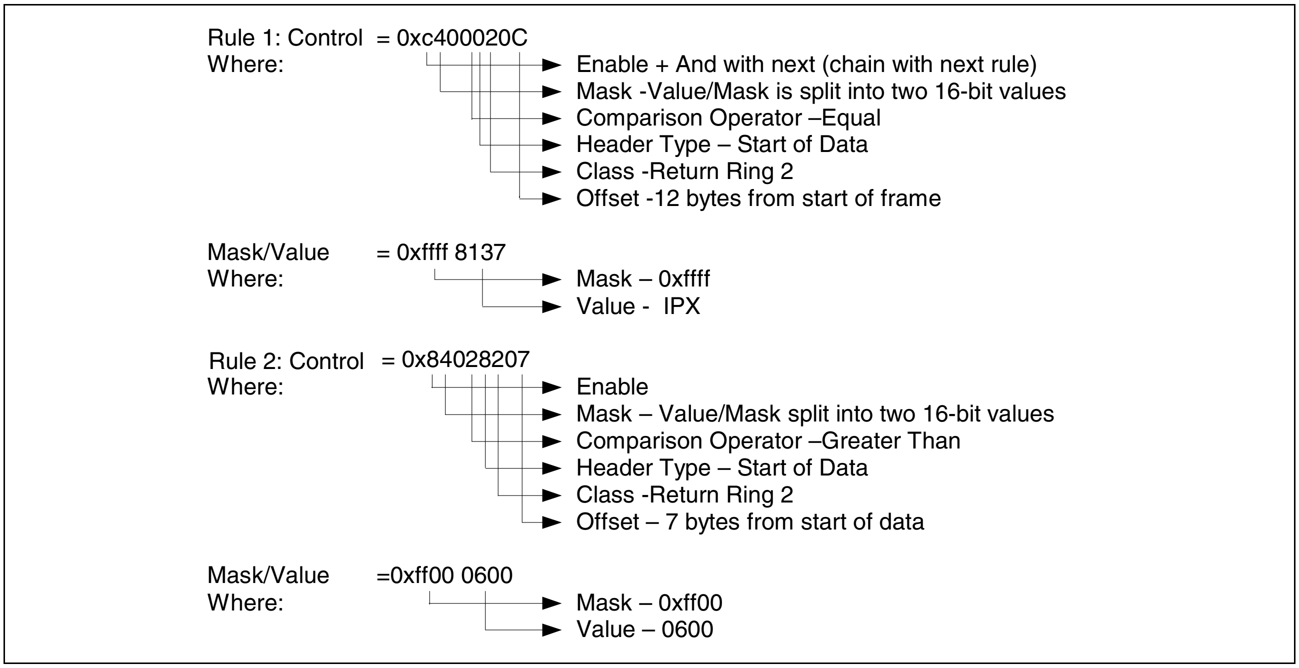
Bit	Name	R/W	Description	Default
31:16	Mask	–	–	–
15:0	Value	–	–	–

Class of Service Example

If either Start of IP Header, Start of TCP Header, or Start of UDP Header is specified, and the frame has no IP, TCP, or UDP header, respectively, there is no frame match. The full set of rules provides a fairly rich selection and filtering criteria.

> ***Example:*** If you wanted to set a Class of Service (CoS) of 2 based on the eighth byte in the data portion of an encapsulated IPX frame using Ethernet Type 2 having a value greater than 6, you could set up the rules shown in Figure 17.

Figure 17: Class of Service Example


Rule 1: Control = 0xc400020C
Where:
Enable + And with next (chain with next rule)
Mask -Value/Mask is split into two 16-bit values
Comparison Operator –Equal
Header Type – Start of Data
Class -Return Ring 2
Offset -12 bytes from start of frame
Mask/Value = 0xffff 8137
Where:
Mask – 0xffff
Value - IPX
Rule 2: Control = 0x84028207
Where:
Enable
Mask – Value/Mask split into two 16-bit values
Comparison Operator –Greater Than
Header Type – Start of Data
Class -Return Ring 2
Offset – 7 bytes from start of data
Mask/Value =0xff00 0600
Where:
Mask – 0xff00
Value – 0600

Checksum Calculation

Whether the host software NOS supports checksum offload or not, the Ethernet controller automatically calculates the IP, TCP, and UDP of received packets as described in RFC 791, RFC 793, and RFC 768, respectively.

Which protocol checksum value is produced can be determined by reading the status flag field in the Receive Return Ring. The valid flag values in the status flag field are IP_CHECKSUM and TCP_UDP_CHECKSUM. When a valid checksum is produced, the values of the checksums are found in the corresponding receive buffer descriptor register. These values should be 0xFFFF for a valid checksum or any other value if the checksum was incorrectly calculated. Assert the Receive No Pseudo-header Checksum bit of the Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521) to not to include Pseudo-header in TCP/UDP checksums.

VLAN Tag Strip

Receiving VLAN-tagged (IEEE 802.1q-compliant) packets are automatically supported by the Ethernet controller. There is no register or setting required to receive packets that are VLAN-tagged. The VLAN tag is automatically stripped from the IEEE 802.1q-compliant packet at reception and then placed in a receive buffer descriptor's two byte VLAN tag field. The flag field has the BD_FLAGS_VLAN_TAG bit set when a valid VLAN packet is received. After the packet has been serviced by the host software, these fields should be zeroed out.

In the Receive MAC Mode register (offset 0x468–0x46b), the Keep VLAN Tag Diag Mode bit (bit 10) can be set to force the Ethernet controller to not strip the VLAN tag from the packet. This is only for diagnostic purposes.

Table 28 shows the frame format with IEEE 802.1Q VLAN tag inserted.

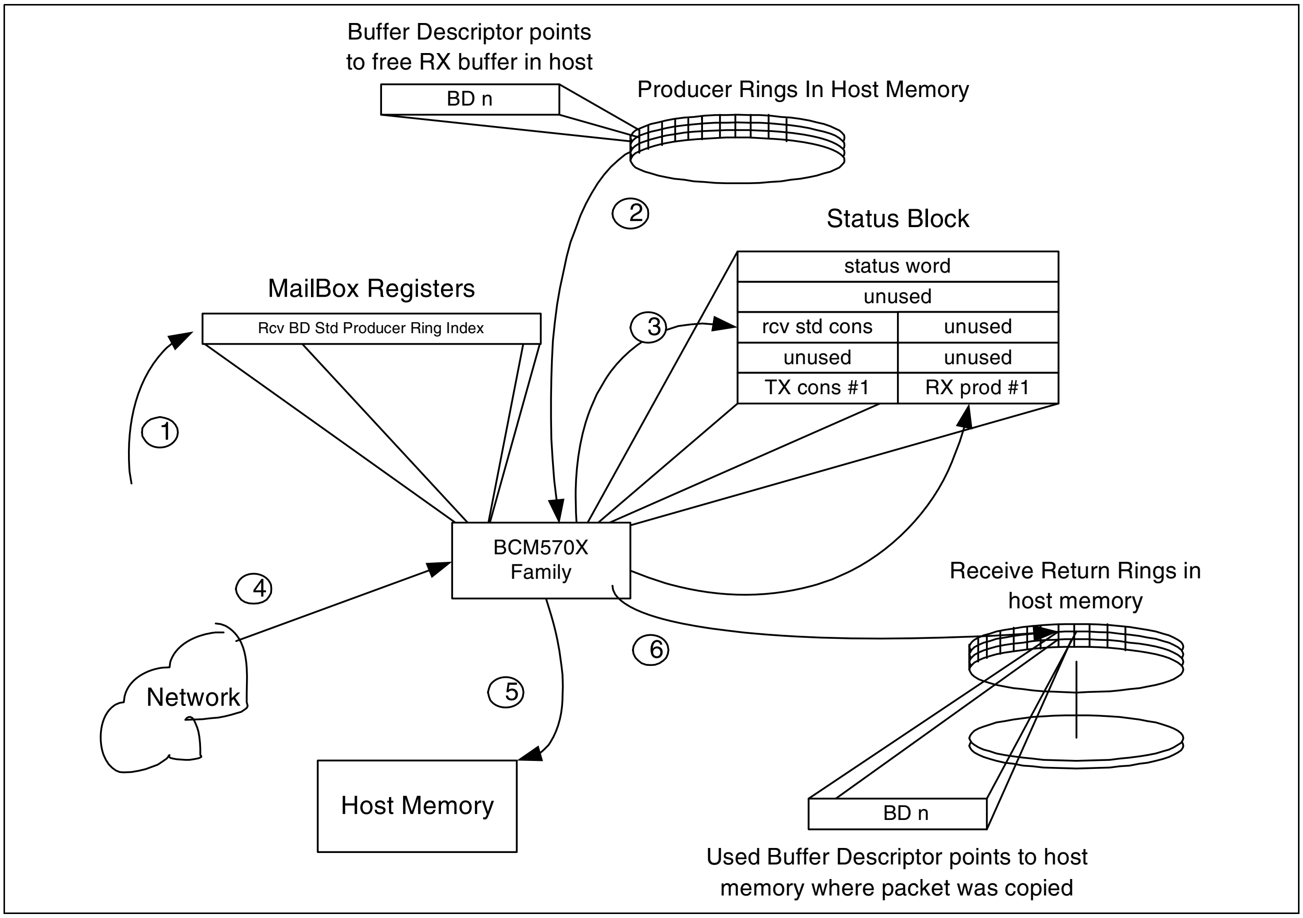
Table 28: Frame Format with 802.1Q VLAN Tag Inserted

Offset	*Description*
0:5	MAC destination address
6:11	MAC source address
12:13	Tag Protocol ID (TPID)—0x8100
14:15	Tag Control Information (TCI): • Bit 15:13—IEEE 802.1P priority • Bit 12—CFI bit • Bit 11:0—VLAN ID
16:17	The original EtherType
18:1517	Payload

RX Data Flow Diagram

The receive data flow can be summarized in Figure 18. The Receive Producer Ring, Receive Buffer Descriptors, Receive Return Rings, Mailbox registers, and status block registers are the main areas of the receive data flow.

Figure 18: Overview Diagram of RX Flow


Buffer Descriptor points to free RX buffer in host
BD n
Producer Rings In Host Memory
2
Status Block
status word
unused
rcv std cons
unused
unused
unused
TX cons #1
RX prod #1
MailBox Registers
Rcv BD Std Producer Ring Index
3
1
BCM570X Family
4
Receive Return Rings in host memory
6
Network
5
Host Memory
BD n
Used Buffer Descriptor points to host memory where packet was copied

The RX flow sequence is as follows:

1. The host software updates a Receive Producer Ring Index in the Mailbox registers.
2. A receive BD or series of BDs with the corresponding index is DMAed to the Ethernet controller from the host-based Receive Producer Ring.
3. The Ethernet controller updates the Receive Consumer Index in the Host Block register and stores copy of the BD.
4. A valid Ethernet packet is received from the network into the device.
5. The Ethernet packet is DMAed to host memory using a BD previously DMAed from a Receive Producer Ring.
6. The BD used for the received packet is DMAed from the Ethernet controller to one of the receive return rings, and the Receive Return Ring Producer Index register in the host status block is updated by the Ethernet controller.

The host software must create an array of BD structures in host memory, referred to as a receive producer ring. Each receive buffer descriptor within a producer ring describes, among other things, the location of a host memory buffer that is used to store the packets received from the network. When the host software (as the producer) updates the mailbox register's producer ring index that corresponds to the receive producer ring, the Ethernet controller automatically DMAs the BD to itself from the host. When the DMA is completed, the Ethernet controller (as the consumer) updates the status block's receive consumer ring index to signal it successfully consumed the BD. The Ethernet controller keeps this BD in internal memory to know where to put a packet that is received from the network.

When a packet is received from the network, a BD gets updated with information regarding the received packet and the packet is DMAed to a location in host memory described by the BD. The Ethernet controller (as the producer) then updates the receive return ring producer index in the Status Block register corresponding to one of host memory's receive return rings, and DMAs the BD to that receive return ring.

It is the responsibility of the host software to setup, initialize, and manage the data structures in host memory, namely, the receive producer rings and the receive return rings. The producer/consumer indices in the mailbox and status block are read and updated by the host and Ethernet controller for this purpose.

Receive Side Scaling

Overview

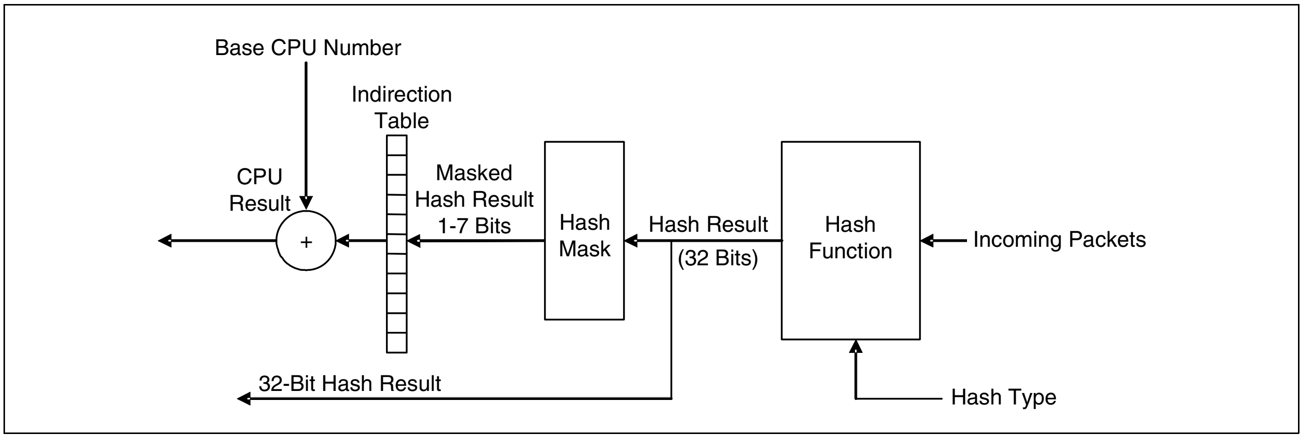
RSS is a scalable networking technology that enables receive packet processing to be balanced across multiple processors in the system while maintaining in-order delivery of the data. The RSS enables packets from a single network adapter to be processed in parallel on multiple CPUs/cores while preserving in-order delivery to TCP connections.

Functional Description

The figure below shows the processing of received packets when RSS is enabled. The RSS algorithm is based on a load-sharing algorithm and performs the following steps.

- Computes a hash on the incoming packet to produce a 32-bit Hash result.
- Performs a lookup in the load balancing table (also called indirection table) using the one to seven least significant bits of the Hash result to determine which of the *n* CPUs are processing the packet, where *n* is the number of CPUs assigned to process received packets.
- Adds a Base CPU Number to determine the exact CPU that will process the packet.

Figure 19: RSS Receive Processing Sequence



The devices implement the above RSS algorithm in hardware except for the step of adding the Base CPU Number to the value from Indirection Table. If required, the step of adding the Base CPU Number to the CPU Result can be done in the main Interrupt Service Routine to determine which CPU will process the packet.

RSS Parameters

Hash Function

The default hash function is the Toeplitz hash. No other hash functions are currently supported, so there is no configurable parameter.

Hash Type

The fields that are used to hash across the incoming packet. The 5 devices support all the four hash types given below and the configuration bits for enabling/disabling these hash types are provided in Receive MAC Mode register at offset 0x468. Any combination of these hash types can be enabled:

- Four-tuple of source TCP Port, source IP version 4 (IPv4) address, destination TCP port, and destination IPv4 address.
- Four-tuple of source TCP Port, source IP version 6 (IPv6) address, destination TCP port, and destination IPv6 address.
- Two-tuple of source IPv4 address and destination IPv4 address.
- Two-tuple of source IPv6 address and destination IPv6 address.

Hash Mask

The RSS Hash Mask bits (bits 22:20 of the Receive MAC Mode register at offset 0x468) allow the configuration of number of hash-result bits that are used to index into the indirection table.

Indirection Table

The table of CPU numbers used for balancing the receive traffic across multiple processors. The Indirection Table registers 0–15 at offset 0x630–0x66F are implemented for the required 128 entries of the Indirection Table. The devices support only four Receive Return Rings so each entry of Indirection Table is implemented as 2 bits.

Secret Hash Key

The hash key that will be used for RSS hash. For the Toeplitz hash, the hash key size is 40 bytes for IPv6 and 16 bytes for IPv4. The host software should program the hash key in hash key registers at offset 0x670 to 0x697.

RSS Initialization

The host protocol stack should configure the above RSS parameters before enabling the RSS engine. The RSS can be enabled by setting the bit-23 of the Receive MAC Mode register at offset 0x468. Normally the RSS parameters except the Indirection Table are static and will be initialized only during device driver initialization. Though extremely rare, the protocol stack may change the RSS parameters any time. The devices require a reset to change any of the hash type, hash mask, and hash key parameters.

If the hash type flags in Receive MAC Mode register (offset 0x468) enable only one type of hash, then any received packet that does not match the enabled hash type is not hashed. If multiple flags are set, such as If the TCP/IPv4 and IPv4 hash types (bits 17 and 16 of Receive MAC Mode register at offset 0x468) are enabled, then if the packet is not a TCP/IPv4 packet but is an IPv4 packet, the hash is performed on just the IPv4 2-tuple. Further, for this setting of the hash type flags, if the incoming packet is not an IPv4 packet, then no hash is performed. Because a variety of hash types can be applied on a per-packet basis (including no hash), the hash type is indicated to the host protocol stack on a per-packet basis. If no hash was performed, then none of the hash type flags in the receive BD will be set.

Once RSS is initialized and enabled, data transfer can begin. Over a period of time, the host protocol stack may modify the indirection table to rebalance the processing load. When the indirection table is changed, it is possible for a short period of time (while the current receive descriptor queues are being processed) for packets to be processed on the wrong CPU. This is a normal transient condition and should not be a problem.

RSS RX Packet Flow

Each CPU or CPU core in multi processor systems is assigned one receive return ring. Only a single interrupt is initiated at a time.

1. As packets arrive, the device parses each packet, calculates the RSS Hash, and derives the CPU number (i.e., receive return ring number) from Indirection Table using the Masked Hash Result as the Indirection Table Index.
2. The packet data is DMAed to the host memory at the location specified by the receive buffer descriptor (RBD) of the receive producer ring.
3. Based on the derived CPU number, the device DMAs the used RBDs into appropriate receive return rings in host memory.
4. The device fires the interrupt via MSI, which causes the device driver ISR to run.
5. The ISR disables further interrupts from the device, determines which CPUs have receive packets to be handled and uses inter processor communication mechanisms to start packet receive handlers on CPUs whose return rings have new RBDs.
6. Each CPU processes the new RBDs in it's receive return ring when its packet handler routine is started by main ISR.
7. Once the main ISR determines that all new RBDs have been processed by the CPUs, it enables the interrupts from the device and exits.

Section 6: Transmit Data Flow

Introduction

Send Buffer Descriptors (BDs) begin on the Send Producer rings. The device driver updates the Mailbox to reflect available Send BDs.

- The MAC moves the available Send BDs to device local memory—a cache.
- Next, the MAC selects a BD from the internal cache using priority scheduling.

The physical address, programmed in the Send BD by the host device driver prior to the Mailbox update, contains the host memory location of the TX packet buffer. The MAC reads the address from Send BD and schedules a bus master DMA for reading the packet data from host buffer. The packet data will be moved into device internal buffers from host buffers by Read DMA engine, and all the read buffers of 1 packet are chained together into a cluster. This cluster is then sent to the transmit MAC which sends the packet data to the integrated PHY for transmission on Ethernet media.

The write DMA engine will subsequently update the status block to indicate that the Send BD was consumed. The host driver normally returns the packet buffers to the NOS/protocol so the next packet can reuse that host physical memory. The send BD is now available for the next TX packet.

Send Rings

The send rings are shared data structures that are used to describe a series of data buffers that will be transferred onto the network. The shared data structure is called the Ring Control Block (RCB), and the entries within a ring for describing the data buffers are called the Send Buffer Descriptors (Send BDs).

Note: The maximum number of Send BDs (buffer descriptors) for a single packet is (0.75)*(ring size).

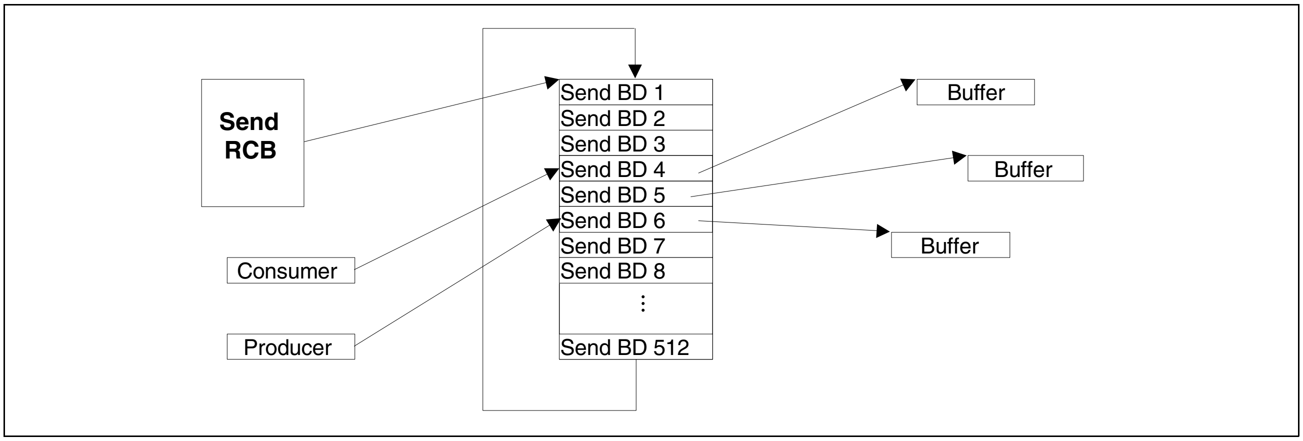
Associated with each ring are two indices that control its operation. These indices are the producer index and the consumer index, which are not shared between the host software and the Ethernet controller. In the case of send rings, the host software controls the producer index by adding elements (initializing a Send BD) to the ring. Similarly, the Ethernet controller controls the consumer index by removing elements (processing a Send BD) from the ring.

The host software is responsible for maintaining its producer index and updating it by writing to the send ring producer index mailbox register. The mailbox registers are described in "Mailbox" on page 94, "High-Priority Mailbox Registers" on page 319 (for offsets 0x200 through 0x3FF), and "Low Priority Mailboxes" on page 513 (for offsets 0x5800 through 0x59FF). The update actually triggers the Ethernet controller to process the send descriptors starting at its consumer index. As a descriptor is processed, the consumer index is incremented, and the new index is reflected in a new status block update. Status block is described in "Status Block" on page 82.

When the producer and consumer indices are equal, the ring is empty. When the producer index is one behind the consumer, the ring is full. Because of this configuration, the producer index always points to an empty slot. Thus, there will always be at least one empty slot in a ring.

Figure 20 illustrates the relationships between all the components of a send ring.

Figure 20: Relationships Between All Components of a Send Ring

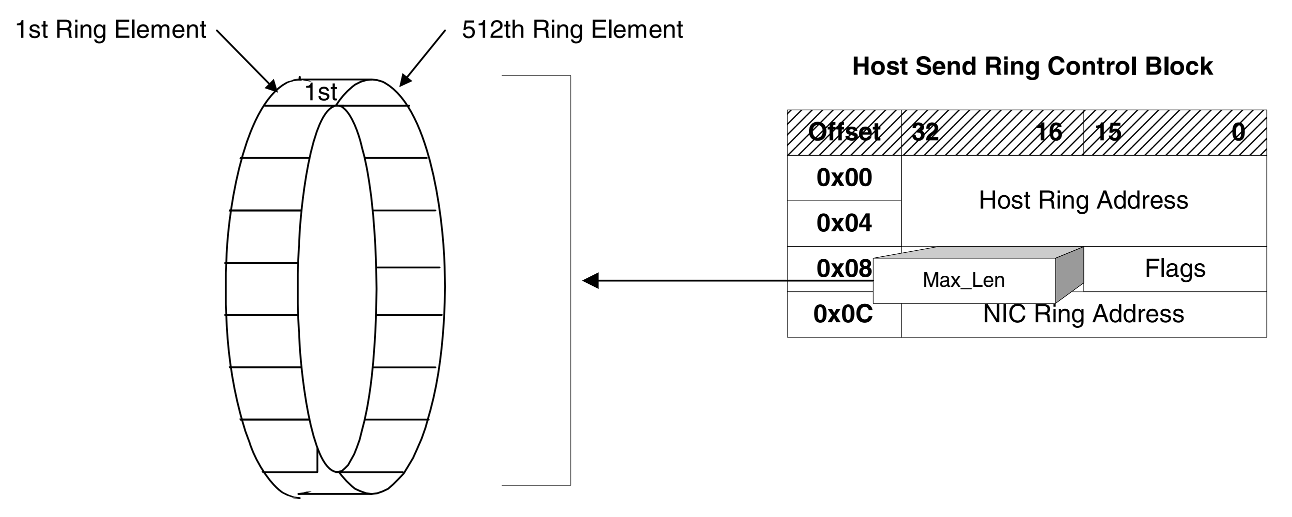


Ring Control Block

The Send Ring RCB contains a pointer to the first Send BD in the device and host memory, number of send BDs in the ring, and control flags (see "Send Rings" on page 107 for a full discussion of the send RCB). All the fields are in big-endian ordering as required by the Ethernet controller. The RCBs of the send rings are located in the device Miscellaneous Memory Region at offset 0x0100.

The devices support a host based send ring. The Send BDs of the host based Send Ring will be bus-mastered from host memory into device local memory. The device driver will program the BDs directly in its memory space and avoid programmed I/O to the MAC. The Max_Len field in the RCB (see Figure 21) indicates the maximum number of BDs in the Send Ring. This field can be programmed to either 32, 64, 128, 256, or 512.

Figure 21: Max_Len Field in Ring Control Block



Offset	32 16	15 0
0x00	Host Ring Address	
0x04		
0x08	Max_Len	Flags
0x0C	NIC Ring Address	

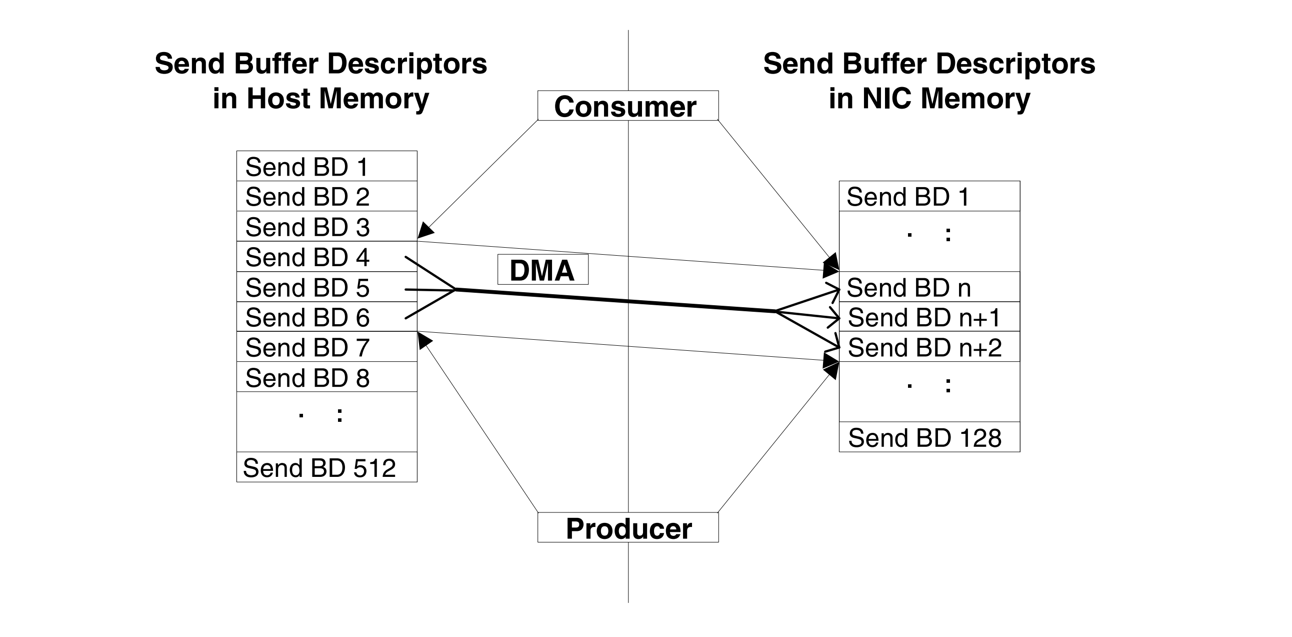
Host-Based Send Ring

The send buffer descriptors of host based send ring reside in host memory.

The host-based send ring will have up to 512 buffer descriptors, which are periodically and transparently DMAed to a staging area inside the NIC internal memory where they are waiting to be consumed. The staging area can hold up to 128 entries per-ring, and Ethernet controller tries to keep the staging area full at all times by constantly monitoring the consumer and producer index (the algorithm for accomplishing this is beyond the scope of this manual). The staging areas are located at a starting offset 0x4000 of NIC memory. Figure 22 illustrates the relationship between the send buffer descriptors in host memory and the staging area in NIC memory.

When the host software initializes new buffer descriptors, its send ring producer index is incremented by the number of descriptors. The new index is then written to the corresponding send ring host producer index mailbox register (starting at offset 0x300 for host standard and offset 0x5900 for indirect mode—see "Send BD Ring Host Producer Index (High Priority Mailbox) Register (offset: 0x300-0x307)" on page 320, which may trigger the Ethernet controller to DMA the descriptors to its staging area. Eventually, the buffer descriptors are processed, and the data associated with these descriptors is transferred onto the network.

Figure 22: Relationship Between Send Buffer Descriptors



The Ethernet controller maintains the send ring consumer index, which is incremented as it processes the descriptors. The Ethernet controller informs the host software of its progress by updating the send ring consumer index in the status block. The host software uses the send ring consumer index and its producer index to determine the empty slots in the ring. The Ethernet controller implements an algorithm that periodically DMAs the status block to host memory in an efficient manner.

Checksum Offload

As network speed increases, offloading is becoming an important feature, and the ability to offload tasks from the host processor aids in the efficiency of the host and in overall system performance. To achieve a significant performance boost, most operating systems now a days offer a mechanism for the TCP/IP protocol stack to offload checksum calculations to the device.

The host software can configure the Ethernet controller to calculate IP, TCP, and UDP checksum as described in RFC 791, RFC 793, and RFC 768 respectively. The first step in checksum calculation is determining the start of an IP and UDP datagram and TCP segment within a frame, which could vary depending on whether the frame is tagged (VLAN) or encapsulated with LLC/SNAP header. Then the checksum is computed from the start to the end of the datagram and inserted into the appropriate location in protocol header. Ethernet controller is designed to support checksum calculation on all frame types and also on IP datagram and TCP segments containing options.

For the Ethernet controller to compute the checksum and insert it into the outgoing frame, the host software must set the appropriate control bits in the send buffer descriptors associated with the frame and seed the checksum field with zero or with the pseudo header checksum.

The host software enables IP checksum calculation by setting the IP_CHKSUM bits in all the send buffer descriptors associated with the frame. The Ethernet controller inserts the checksum into the checksum field of the IP header.

To enable TCP or UDP checksum calculation, the host software must set the TCP_UDP_CKSUM bit in all the send buffer descriptors associated with the frame containing the entire UDP datagram or TCP segment. The TCP and UDP checksum engines do not span IP fragmented frames.

The host software can configure the Ethernet controller to disable TCP or UDP pseudo-header checksum calculation by setting the Mode_Control.Send_No_Pseudo_Header_Checksum bit. When set, the host software must seed the checksum field in the TCP or UDP header with the pseudo-header checksum. If the Mode_Control.Send_No_Pseudo_Header_Checksum is cleared, the Ethernet controller computes the checksum including the pseudo header and inserts it into the checksum field.

Large Segment Offload

In computer networking, large segment offload (LSO) is a technique for increasing outbound throughput of high-bandwidth network connections by reducing host CPU overhead. This is done by queuing up large TCP packets letting the Ethernet controller split them into separate (smaller) TCP packets to be transmitted onto the network. The technique is also called TCP segmentation offload (TSO) or, more generically, LSO.

When large blocks of data are to be sent over a computer network they must be first broken down to smaller segments that can pass through all of the network elements such as routers and switches between the source and destination computers. This process is referred to as segmentation.

For example, a large TCP packet of 64KB (65,536 bytes) of data is usually segmented into 46 segments of 1448 bytes each before being sent over the network through the Ethernet controller chip. With some intelligence in the controller, the host CPU can hand over a 64k byte TCP packet directly to the controller in a single transmit request and the controller can break the large TCP packet down into smaller segments of 1448 bytes, add the TCP, IP, and data link layer protocol headers to each segment, and send the resulting frames over the network. This significantly reduces the work done by the host CPU.

Some Broadcom Ethernet controllers, such as the BCM5725/BCM5762/BCM57767, also support using jumbo sized frames (up to 9,216 bytes) as the individual frame size into which a large offloaded TCP packet is segmented into.

Note: The UDP checksum engine does not span IP fragmented frames.

Note: The Ethernet controller does not validate the value of the Length field and may generate an error on the PCI bus if the Length field has a value of 0. The host driver must ensure that the Length field is nonzero before enqueueing the BD onto the Send Ring.

QuickStart

Follow the steps to enable the LSO:

1. Zero TCP checksum field in offloaded packet (leave IP checksum field alone)
2. Set register 0x0C00[3]=1: Enable HW LSO pre-DMA processing
3. Set register 0x4800[27]=1: Enable hardware processing of LSO IPv4 packets
4. Set register 0x4800[28]=1: Enable hardware processing of LSO IPv6 packets (if desired)
5. Set Send BD Flags[8]=1: CPU pre-DMA
6. Set Send BD Flags[9]=1: CPU post-DMA
7. See LSO Limitations section below

Note: In Broadcom controllers that have a physically separate isochronous (ISO) TX queue, there is a parallel set of register fields, which mirror that of the normal TX path, for controlling LSO on the ISO TX path.

LSO-Related Hardware Control Bits

Table 29: Send Data Initiator Mode Register (Offset: 0xC00)

Name	Bits	Access	Default Value	Description
Hardware Pre-DMA Enable	3	RW	0	Enable hardware LSO pre-DMA processing

The ISO Send Data Initiator Mode register is applicable only to controllers that have a secondary TX ISO (Isochronous) queue.

Table 30: ISO Send Data Initiator Mode Register (Offset: 0xD00)

Name	Bits	Access	Default Value	Description
Hardware Pre-DMA Enable	3	RW	0	Enable hardware LSO pre-DMA processing

Table 31: Read DMA Mode Register (offset: 0x4800)

Name	Bits	Access	Default Value	Description
Hardware IPv6 Post-DMA Processing Enable	28	RW	1	Enable hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets. This bit when clear disables IPV6 Processing. This bit was not used for controllers before BCM5725/BCM5762/BCM57767.
Hardware IPv4 Post-DMA Processing Enable	27	RW	0	Enable hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets. This bit is the TCP Segmentation Enable bit.

The ISO Read DMA Mode register is applicable only to controllers that have a secondary TX ISO (Isochronous) queue.

Table 32: ISO Read DMA Mode Register (Offset: 0x4A00)

Name	Bits	Access	Default Value	Description
Hardware IPv6 Post-DMA Processing Enable	28	RW	1	Enable hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets. This bit when clear disables IPV6 Processing. This bit was not used for controllers before BCM5725/BCM5762/BCM57767.
Hardware IPv4 Post-DMA Processing Enable	27	RW	0	Enable hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets. This bit is TCP Segmentation Enable bit.

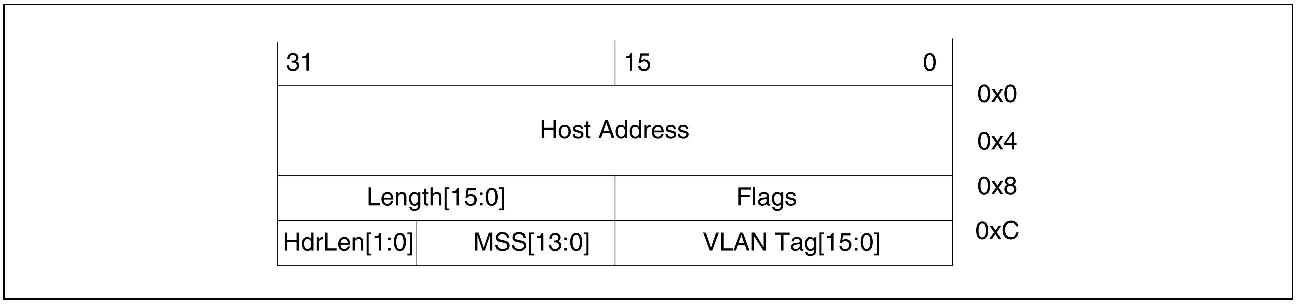
The ISO related registers are to allow for processing of ISO Ethernet Audio Video (EAV)-related TX traffic. A physically separate isochronous TX queue exists in some Broadcom Ethernet controllers to support audio/video traffic applications, which require very precisely timed isochronous launch of TX packets onto the wire.

Note: LSO using jumbo frames is permissible on some Broadcom controllers (i.e. BCM5725/BCM5762/BCM57767). This is accomplished by appropriately programming the MSS field of the Send BD.

Send Buffer Descriptor

The Send Buffer Descriptor (SBD) diagram is shown below.

Figure 23: Send Buffer Descriptor



Host Address

This field is a 64-bit address specifying where the Send Buffer is located in Host memory.

Length[15:0]

This field is the length of the frame or TCP large segment to be transmitted.

VLAN Tag[15:0]

This field is the VLAN Tag to be inserted into the frame if Flags[6] is set to 1.

HdrLen[7:0]

This field is the length of the Ether + IP + TCP Headers to be replicated in each segment arising out of a Large TCP Segment transmit operation.

The HDRLEN field is split into two fields within the SBD:

- A subfield within the Flags field at a word offset of 0x08
- Adjacent to the MSS field at an offset of 0x0c

This field is used only for LSO buffers. Its value specifies the combined L3 and L4 header length in 4-byte DWORDS for TCP/IPv4 or TCP/IPv6 packets. The value includes any option headers for IPv4, any extension headers for IPv6, and any TCP options.

For a TCP/IPv4 packet without IP or TCP options, this field would have a value of 10 (decimal).

For a TCP/IPv6 packet without extension headers or TCP options, this field would have a value of 15 (decimal).

For a TCP/IPv6 packet with a hop-by-hop options extension header of length 8 bytes plus a TCP MSS option (4 bytes), this field would have a value of 18 (decimal).

MSS[13:0]

This field is the size of the TCP segments into which a LSO segment is to be segmented into. Note that the MSS field has been increased for some NetXtreme controllers (i.e. BCM5725/BCM5762/BCM57767) to hold a value specifying a jumbo frame size.

Flags

See the table below.

Table 33: Flag Field Description

Bit #	*Flag Name*	*Flag Description*
0	TCP/UDP Checksum Offload Enable	This bit enables calculation of TCP or UDP checksums for IPv4 and IPv6 transmitted packets. The driver will set this bit only if the packet contained within a buffer is TCP or UDP over IPv4 or IPv6.
1	IP Checksum Offload Enable	This bit enables calculation of the IPv4 layer-3 checksum. This bit will be set only for IPv4 packets. The driver will never set it for IPv6 packets.
2	Packet End	This bit will be set for the last send buffer in a packet.
3	Jumbo Frame	Driver must set this bit to 1 if the MTU length of the Send Frame is > 1500B. The MTU length is the Ethernet payload length and excludes Header length (and Trailer length). All BDs belonging to a Send Packet must configure this bit identically.
4	HDRLEN[2]	The length of the Ethernet + IP + TCP Headers to be replicated in each segment arising from Large TCP Segment Offload (LSO) activity.
5	SNAP	This bit shall be set for packets that include a SNAP header within the Ethernet header. Do not set this bit for LSO packets.
6	VLAN TAG	When this bit is set, the NIC will insert a VLAN tag in the Ethernet header. The value for the inserted tag is taken from the VLAN Tag field in the send BD.
7	Coalesce Now	If this bit is set, then a status block with an updated send consumer index will be DMA'd to the host as soon as this buffer's data has been DMA'd from the host. An interrupt may or may not be generated depending on the present state of interrupt avoidance mechanisms.

Table 33: Flag Field Description (Cont.)

Bit #	Flag Name	Flag Description
8	CPU Pre-DMA	If this bit is set, then the CPU will be required to act upon the buffer before the send data initiator state machine is kicked off. Alternately, if hardware LSO is enabled and this bit is set in conjunction with CPU Post-DMA, then this buffer will be treated as part of an LSO segment to be further segmented by hardware.
9	CPU Post-DMA	If this bit is set, then the CPU will be required to act upon the buffer before the send data completion state machine is kicked off. Alternately, if hardware LSO is enabled and this bit is set in conjunction with CPU Pre-DMA, then this buffer will be treated as part of an LSO segment to be further segmented by hardware.
10	HDRLEN[3]	The length of the Ether + IP + TCP Headers (combined) to be replicated in each frame arising from Large TCP Segment Offload activity (LSO). Maximum HDRLEN could be 200B.
11	HDRLEN[4]	
12	HDRLEN[5]	
13	HDRLEN[6]	
14	HDRLEN[7]	
15	Do Not Generate CRC	If set to 1, the controller will not append an Ethernet CRC to the end of the frame.

LSO Limitations

The limitations of the SBD are listed below.

- MSS must not be less than 8 Bytes.
- LSO packet must be a TCP packet.
- IP length field must not be incorrect.
- TCP length field must not be incorrect.
- Total offloaded TCP payload length must be greater than the MSS selected in the SBD.
- For all LSO segments, SBD flag bit 8 and 9 (CPU pre-DMA and CPU post-DMA) must be set.
- LSO packet may not be IEEE 802.3 format with LLC and SNAP headers.
- L2 header must be contained within the very first SBD.
- IP header (IPv4 or IPv6), including IP options, must be contained within a single SBD.
- HdrLen[7:0] field must be correct in SBD.
- DONT_GEN_CRC must not be set in a SBD for an LSO packet.
- SNAP field must not be set in the SBD for an LSO packet.
- TCP Header, including TCP Options, must be contained within a single SBD.
- The total length for all headers (L2/L3/L4) combined, plus options, may not exceed 200 bytes.

Additional LSO Notes

The TCP/UDP and IP checksum offload enable bits in the SBD may be either set or cleared. The hardware still works as expected when LSO is in use (i.e., checksums are calculated/inserted by the hardware since this is a natural requirement for doing LSO).

The driver should zero the TCP checksum field in the offloaded TCP packet, but leave the IP checksum alone. This requirement may change with newer NetXtreme controllers.

Broadcom drivers enable long burst by default in the Read DMA Mode register (0x4800 bits 17:16 = 11 binary = 4k byte burst size).

Do not set TXFIFO Underrun Prevention Enable (bit 31) in the Buffer Manager Mode register 0x4400.

Do not set bit 5 (Multiple Segment Enable) in 0xC00 (Send Data Initiator Mode register).

Example TCP-segmentation-related (LSO) register values

Source: Broadcom tg3 Linux driver with BCM5764M NIC

```
0x4800     08033BFE      Read DMA Mode Register
0x0C00     0000000A      Send Data Initiator Mode Register
0x0CF4     50000020      Pre-DMA Command Exchange for Segmentation
0x0CEC     00040028      DMA Flag Register for TCP Segmentation
0x1008     A0000000      Pre-DMA Command Exchange for Segmentation
0x0CE8     00000036      Length/Offset Register for Segmentation
0x0CF0     00000000      VLAN Tag Register for TCP Segmentation
```

Jumbo Frames

The BCM5725/BCM5762/BCM57767 supports jumbo Ethernet frames in both the receive (RX) and transmit (TX) paths. The jumbo frame architecture and the software interface is nearly identical to that of the legacy NetXtreme family controllers which also supported jumbo frames. The related jumbo architectural changes for the BCM5725/BCM5762/BCM57767 are described in this section.

Following are the feature highlights.

The maximum jumbo frame length supported by the BCM5725/BCM5762/BCM57767 is 9622B, which may be broken down as follows:

- 9600B MTU payload
- 14B Ethernet Header
- 4B Ethernet FCS
- Optional 4B of VLAN Tag
- 9622B maximum size limit applies equally to both TX and RX paths

Transmit Side:

- CRC checksum offload of jumbo frames is permitted.
- TCP and IP checksum offload of jumbo frames is permitted.
- TCP segmentation offload (TSO), a.k.a. large segment offload (LSO), of jumbo frames is permitted.
- Jumbo frames are to be constructed out of standard send buffers.
- There is a single send buffer ring which jumbo and standard frames share. The driver may inter-mix jumbo and standard frames in the send ring without restriction.
- The TX MBUF on-chip memory has be upsized to 22K bytes.
- The behavior of TX jumbo frame processing remains identical to that of standard TX frame processing:
  - A TX frame is first completely DMAed into the TX MBUF memory and only then will it be transmitted onto the wire.
  - The TX EMAC treats jumbo frames exactly the same as it treats a standard frame, except for being cognizant of the length.
  - Full-Duplex and Half-Duplex behavior remains the same.
  - Host coalescing timing remains identical to that of standard frames.
  - A new jumbo frame flag is introduced in the send buffer descriptor (SBD). The driver must set this flag bit to indicate a jumbo frame (i.e., frame length > 1514 bytes without CRC and VLAN tag fields)
  - In Multiple Send Queue mode, all 16 send queues are permitted to post jumbo frames. Thus there are 16 send ring control blocks (RCBs) available in this mode.

Receive Side:

- An additional BD producer ring (the jumbo producer ring) has been introduced.
- The receive side retrieves buffers only from the jumbo producer ring in turn to post a received jumbo frame. Similarly, the receive side only retrieves buffers from the standard producer ring in turn to post standard sized frames.
- The jumbo producer ring is populated with BDs of a format known as the Extended Buffer descriptors. This is different from the standard buffer descriptor format.
- It is important to note that the driver may post extended BDs only in the jumbo producer ring and may post standard BDs only in the standard producer ring. BD formats may not be interchanged.
- The return rings are heterogeneous. That is, the controller returns both standard BDs and extended BDs in the same return ring. Intermixing of both types of BDs can happen without any restriction.
- RX MBUF memory has been upsized to at least 32 KB.
- Both 4-tuple and 2-tuple RSS computation and classification are performed over RX jumbo frames.
- There are 4 return rings as per RSS requirements (this is not affected by jumbo frame support).
- Receive CRC calculation and checking is performed on jumbo frames.
- Hardware calculates TCP, UDP, and IP checksums on jumbo frames.
- In IOV mode (I/O Virtualization), all 17 receive queues (virtual receive queues) must be provided with extended receive BDs. To that end, each VRQ is provided with a dedicated standard receive BD (RBD) ring and a dedicated jumbo RBD Ring. The BCM5725/BCM5762/BCM57767 does not support IOV, but this same jumbo architecture is present in other closely related NetXtreme controllers which do offer IOV support.

Other:

- Miscellaneous BD memory has been increased over legacy NetXtreme controllers from 6K (4K receive BD + 1K send BD + 1K gencomm) to 51KB (17 KB standard receive BD + 17K jumbo receive BD + 16KB send BD + 1K gencomm). Gencomm describes on-chip memory space used for driver to/from boot code/firmware communication.
- The structure of the status block has been updated in order to accommodate jumbo frame related information.
- The controllers memory map has also been updated.

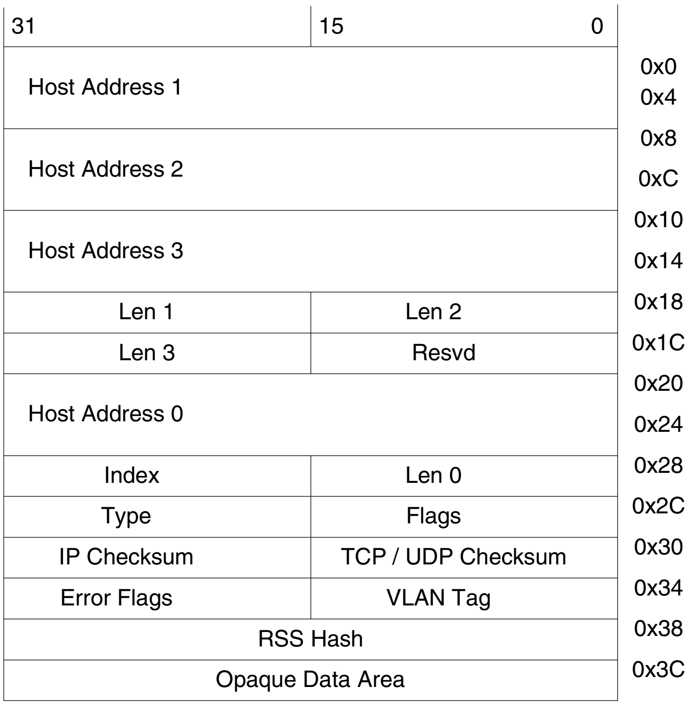
Affected Data Structures

The data structures introduced, updated, or affected due to jumbo frame support are discussed in this section. In IOV mode, some of these structures are instantiated 17 times in case of receive and 16 times in case of transmit.

Extended RX Buffer Descriptor (BD)

The extended buffer descriptors are only permitted to be posted to the jumbo producer ring. The structure is shown in Figure 24: "Extended RX Buffer Descriptor," on page 121. The main distinction of the extended BD structure is that it can point to a maximum of four pieces of a scattered receive buffer. Hence the structure contains four host addresses and four length fields.

Figure 24: Extended RX Buffer Descriptor



31	15 0	
Host Address 1		0x0 0x4
Host Address 2		0x8 0xC
Host Address 3		0x10 0x14
Len 1	Len 2	0x18
Len 3	Resvd	0x1C
Host Address 0		0x20 0x24
Index	Len 0	0x28
Type	Flags	0x2C
IP Checksum	TCP / UDP Checksum	0x30
Error Flags	VLAN Tag	0x34
RSS Hash		0x38
Opaque Data Area		0x3C

- The Host Address 0 field contains the address of the first buffer in host memory. The host address is in host address format (64-bit).
- The Host Address N field in the Extended Receive Buffer Descriptor contains the address of the Nth piece of the buffer in host memory.
- The Index field is used by the host to keep track of the position of the returned buffer descriptor. This field is passed through opaquely by the controller.
- The Len 0 field is initially set by the host and specifies the length of the first buffer available for receiving data; this length field is set to the length of the data pointed to by Host Address 0. When an extended BD is returned to the receive return ring, the Len 0 field is set to the entire length of the data associated with the buffer descriptor, which is so because the receive return ring contains only a single length field. All BDs posted to the return ring by the controller are of size 32 bytes, whereas extended receive BDs posted to the jumbo producer ring by the driver are of size 64 bytes.

Note: In the case of an extended BD, the host is permitted to make Len 0 > 4 KB and practically even beyond 9.6 KB, such that an entire jumbo frame could be held in a single buffer. In such a scenario, hardware attempts to post an entire jumbo frame in a single buffer designated by Len 0.

Note: Len 0 is not permitted to be 0. Len2, Len3, or Len 4 may be set to 0, but hardware ignores Len3 and Len 4 when Len2 = 0. Similarly, hardware ignores Len4 when Len3 = 0.

Note: None of the Len N values may be less than 8 bytes.

- The Len1, Len2 and Len3 fields contain the respective lengths of the remaining three piece of the buffer in host memory.
- The Type field is used by the controller internally and should be ignored and need not be set by the host.
- The Flags bits are used to indicate any special processing that is needed in the buffer. Bits that are not explicitly defined here must be set to zero. See Table 35.
- The IP Checksum field is the checksum of the entire IP header. A correct checksum is 0 or 0xffff.
- The TCP/UDP Checksum field is the checksum of all data following the IP header, for the length defined in the IP header. If the Receive No Pseudo-header Checksum bit is set, then the pseudo header checksum is not added to this value. If the bit is set this value includes the pseudo header.
- The Error Flags field contains a bitmask of possible errors. It is only valid if the BD_FLAGS_FRAME_HAS_ERROR bit (see Table 35) is set in the Flags field.

Table 34: Receive BD Error Flags

Bit#	Error Flag Name	Description
16	BD_ERR_BAD_CRC	This frame has a bad CRC.
17	BD_ERR_COLL_DETECT	This frame had a collision.
18	BD_ERR_LINK_LOST_DURING_PKT	The link was lost while this, incomplete frame was being received.
19	BD_ERR_PHY_DECODE_ERR	The frame had an unspecified frame decoding error.
20	BD_ERR_ODD_NIBBLED_RCVD_MII	The packet arrived with an odd number of nibbles.
21	BD_ERR_MAC_ABORT	The MAC aborted the packet due to an unspecified internal error.
22	BD_ERR_LEN_LT_64	The MAC received a runt packet.
23	BD_ERR_TRUNC_NO_RESOURCES	The MAC could not receive this entire packet due to a lack of internal resources. Note that this bit is not set for frame longer than MTU that have bad CRC.
24	BD_ERR_GIANT_FRAME_RCVD	This frame was longer than the maximum packet length value set in the Receive MTU register. The data is truncated at the length specified in the Receive MTU register. This bit must be set when bit 4 BD_FLAG_IP_FRAG_END is set for the last BD of the last fragment in a train of fragments.
25:31	Reserved	–

- The VLAN Tag field is filled in if the BD_FLAGS_VLAN_TAG bit is set in the flags field. It is the 2 byte VLAN tag that has been extracted from a 802.1Q compliant frame.
- The Reserved field is used internally by the controller. The host should ignore the value of this field.
- The Opaque Data field is reserved for the driver and any data placed here is passed opaquely by the driver from the receive buffer descriptor in the standard or jumbo receive ring to one of the receive return rings.

Table 35: Receive BD Flags

Bit #	Flag Name	Flag Description
0	Reserved	–
1	Reserved	–
2	BD_FLAG_END	The frame ends at the end of the data in this buffer descriptor.
3	RSS_HASH_VALID	If this bit is 1, then the RSS_HASH_TYPE field is valid. Else the RSS_HASH_TYPE field is meaningless and must be ignored for this frame.
4	Reserved	–
5	BD_FLAG_JUMBO_RING	Indicates that this packet came from the Jumbo Receive Ring, not the Standard Receive Ring (For receive BDs only). This must be set by the driver, it is just copied through opaquely by the controller firmware.
6	BD_FLAG_VLAN_TAG	The frame associated with this buffer descriptor has an 802.1Q VLAN tag associated with it.
7:8	RSS_HASH_TYPE	Hash type of the receive packet. It indicates which hash_type was used on the receive packet if multiple hash types are defined.
9	Reserved	–
10	BD_FLAG_FRAME_HAS_ERROR	An error was detected by the controller. The detected error type is set in the Error_Flag word of the receive buffer descriptor.
11	–	–
12	IP_CHECKSUM	Indicates that the IP Checksum field is valid.
13	TCP_UDP_CHECKSUM	Indicates that the TCP_UDP Checksum field is valid.
14	TCP_UDP_IS_TCP	Indicates that this frame has a TCP packet in it.
15	IPV6_PACKET	Indicates that this frame has an IPv6 packet in it.

Receive Jumbo Producer Ring

The jumbo RBD producer ring is reintroduced in the BCM5725/BCM5762/BCM57767 to support RX jumbo frames. The jumbo RBD producer ring is structured the same as the standard RBD producer ring, but the primary difference is that only extended BDs are permitted to be posted in the jumbo RBD producer ring. The jumbo RBD producer ring has a unique RCB associated with it.

The jumbo ring is managed by a producer index and a consumer index as in the case with the standard producer ring. Whenever host software adds more BDs to the jumbo producer ring, it writes the updated producer index to the controller via a high-priority mailbox located at the PCIE address range 0x208–0x20F. The producer index register is at 0x3008.

The controller keeps a local copy of the jumbo ring consumer index in register 0x2470. The jumbo ring consumer index is also reported to the host via the status block. See "Send Buffer Descriptor" on page 126).

There is also a jumbo ring replenishment threshold register 0x2C1C. The controller DMAes BDs from the jumbo ring in advance and caches them locally in controller memory. The controller initiates DMA of more BDs anytime the local number of cached BDs falls below the threshold programmed in this register.

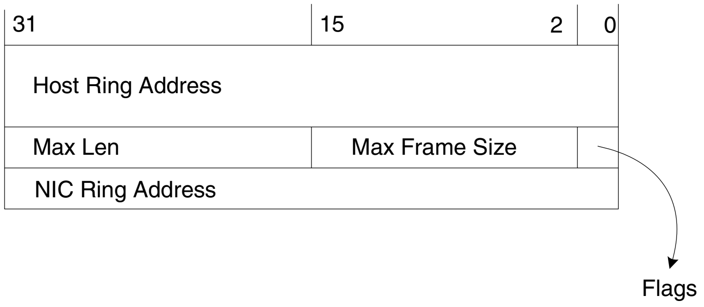
The maximum number of BDs that can be held in the jumbo ring is programmed in that ring's ring control block RCB.

Ring Control Blocks

Each host based ring has an RCB associated with it. The same structure applies to all types of host rings, (i.e., receive producer ring, receive return ring, and also send rings). In the case of the BCM5725/BCM5762/BCM57767, the RCB structure is enhanced to accommodate jumbo frames. However, the enhancement applies only to the Receive Standard Producer ring.

The RCB essentially points to the physical address of the host memory where a producer ring is placed. The BCM5725/BCM5762/BCM57767 uses two RCBs, one for the Standard Ring and one for the jumbo ring. The jumbo ring RCB is implemented over a set of four registers at the address range 0x2440–0x244F. The standard ring RCB register addresses remain the same at 0x2450–0x245F. The send ring RCB and return ring RCBs are memory-mapped.

Figure 25: Ring Control Block



The host ring address is the host address of the first ring element. The host ring address is in host address format. In controller-based send rings, the host address is ignored.

The NIC ring address is the address where a portion of a ring is cached in the controller's miscellaneous BD memory. The driver need not program anything to these fields in most NetXtreme controllers. However, the BCM5725/BCM5762/BCM57767 does require the driver to program non-hardware-default values here.

The Max Len field is interpreted differently for different types of rings. In the case of receive producer rings and receive return rings, this field indicates the maximum number of entries the ring can hold. The BCM5725/BCM5762/BCM57767 imposes constraints for different rings as shown below:

- Receive standard producer ring and send ring: Max Len should be programmed by the host to indicate the maximum number of entries the ring will hold. The allowable values for the Standard Producer Ring Max Len are 32, 64, 128, 256, 512, 1024, and 2048.
- Receive jumbo producer ring: Max Len should be programmed by the host to indicate the maximum number of entries the ring will hold. The allowable values for the jumbo producer ring Max Len are 32, 64, 128, 256 and 1024.

- Receive return rings: Max Len should be programmed by the host to indicate the maximum number of entries the ring will hold. In case of return rings, the host must program this field to be greater than or equal to the combined value of Max Len fields of the receive standard producer ring and the receive jumbo producer ring. For example, if Standard Ring Max Len == 32 and jumbo ring Max Len == 32, then return ring Max Len must be 64 or higher. This means the allowable values for the return ring Max Len are 32, 64, 128, 256, 512,1024, 2048, and 4096.

The receive standard ring RCB uses the Max Frame Size field to indicate the maximum length of each buffer to be described by the buffer descriptor placed into the standard ring. In this manner any frame received that is larger than this value causes the controller to attempt to use a jumbo ring buffer instead of a standard ring buffer. The Max Frame Size field is unused in the receive jumbo ring RCB, receive return ring RCB and send ring RCB. The Flags field is described below in Table 36.

Table 36: RCB Flags

Bit #	***Flag Name***	***Flag Description***
0	Reserved	Reserved
1	RCB_FLAG_RING_DISABLED	Indicates that the Ring is not in use

Receive Return Ring(s)

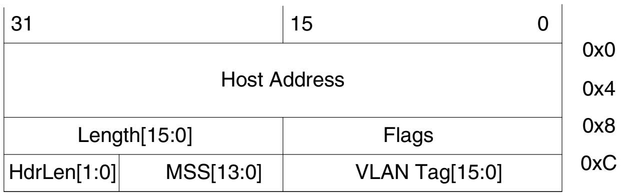
There are no structural changes to the return rings in the BCM5725/BCM5762/BCM57767. There are no functional changes from standard size frames or from a Receive Side Scaling (RSS) perspective. The impact of the jumbo frame feature to the return rings is clarified here:

- The same return ring carries jumbo frames as well as standard size frames. Intermixing can happen without any limitation.
- The total number of return rings remains 4. However, when RSS is disabled, return ring 0 is the only active return ring and all RX frames are returned over Ring 0.
- When an extended buffer descriptor is returned to a receive return ring, the extended portion of the descriptor is truncated by the controller. The offset range 0x00–0x01F is the extended portion (see Figure 25).
- Furthermore, when an extended BD is returned in a return ring, the length of the entire RX frame is consolidated in the Len 0 field even if any or all of the scatter buffer pieces of the extended BD were used to place the frame in host memory.

Send Buffer Descriptor

The send buffer descriptor (SBD) has been updated to accommodate LSO over jumbo frames. Figure 26 below illustrates the updated SBD format.

Figure 26: Send Buffer Descriptor



- The Host Address is the 64-bit address where the send buffer is located in host memory.
- The Length[15:0] is the length of the frame or TCP large segment to be transmitted.
- The VLAN TAG[15:0] field is the tag to be inserted in the frame if flags[6] is set to 1.
- The aggregate HDRLEN[7:0] field is the length of the Ethernet + IP + TCP headers to be replicated in each segment arising out of a large TCP segment (LSO). (See Flags also.)
- The MSS[13:0] field is the size of the TCP segments into which a LSO segment is to be chopped up into. Note that it has been increased to hold the value of a jumbo frame size. The Flags field of SBD is shown in Table 37 below.

Table 37: Send Buffer Descriptor Flags

Bit #	***Flag Name***	***Flag Description***
0	TCP/UDP Checksum Offload Enable	This bit enables calculation of TCP or UDP checksums for IPv4 and IPv6 transmitted packets. The driver sets this bit only if the packet contained within a buffer is TCP or UDP over IPv4 or IPv6.
1	IP Checksum Offload Enable	This bit enables calculation of the IPv4 layer-3 checksum. This bit is set only for IPv4 packets. The driver never sets it for IPv6 packets.
2	Packet End	This bit is set for the last send buffer in a packet.
3	Jumbo Frame	Driver must set this bit to 1 if the MTU length of the Send Frame is > 1500B. The MTU length is the Ethernet payload length and excludes header length (and trailer length). All BDs belonging to a send packet must configure this bit identically.
4	HDRLEN[2]	The length of the Ethernet+IP+TCP headers to be replicated in each segment arising out of a large TCP segment (LSO).
5	SNAP	This bit is set for packets that include a SNAP header within the Ethernet header. This bit is not implemented in HW for schedule reasons.
6	VLAN TAG	When this bit is set, the controller inserts a VLAN tag in the Ethernet header. The value for the inserted tag is taken from the VLAN Tag field in the send BD.

Table 37: Send Buffer Descriptor Flags (Cont.)

Bit #	Flag Name	Flag Description
7	Coalesce Now	If this bit is set, a status block with an updated send consumer index is DMA'd to the host as soon as this buffer's data has been DMA'd from the host. An interrupt may or may not be generated depending on the present state of interrupt avoidance mechanisms.
8	CPU Pre-DMA	If this bit is set, the CPU is required to act upon the buffer before the send data initiator state machine is kicked off. Alternately, if hardware LSO is enabled and this bit is set in conjunction with CPU post-DMA, then this buffer is treated as part of an LSO segment to be further segmented by hardware.
9	CPU Post-DMA	If this bit is set, the CPU is required to act upon the buffer before the send data completion state machine is kicked off. Alternately, if hardware LSO is enabled and this bit is set in conjunction with CPU pre-DMA, then this buffer is treated as part of an LSO segment to be further segmented by hardware.
10	HDRLEN[3]	The length of the Ether+IP+TCP headers (combined) to be replicated in each frame arising out of a large TCP segment (LSO). Maximum Header Length could be 256B.
11	HDRLEN[4]	
12	HDRLEN[5]	
13	HDRLEN[6]	
14	HDRLEN[7]	
15	Do Not Generate CRC	If set to 1, the controller will not append an Ethernet CRC to the end of the frame.

Status Block

The status block has been modified in order to accommodate the jumbo producer ring's consumer index.

The status block is a data structure in the host memory. The host driver uses this data structure to trace the packet receive and transmission status and resource usage. Its length is 24 bytes. The driver needs to configure the status block host address register to point to the physical address in host memory for this data structure. The BCM5725/BCM5762/BCM57767 will update the status block in host memory (via DMA) prior to a host coalescing interrupt or MSI/MSI-X. The frequency of these status block updates is determined by the host coalescing logic. The two status block update interrupt triggers are RX/TX coalescing timer and RX/TX maximum coalesced frame count threshold.

A new field to indicate the Receive Jumbo Producer Ring Consumer Index is added to the BCM5725/BCM5762/BCM57767 Legacy RSS mode status block. The updated structure of the status block is shown in Table 38 below.

Note that there are multiple formats of status blocks.

Table 38: Status Block

<table>
<tr><th>Offset</th><th>31 16</th><th colspan="2">15 0</th></tr>
<tr><td>0x00</td><td colspan="3">Status Word</td></tr>
<tr><td>0x04</td><td colspan="2">Reserved 0x0</td><td>Status Tag[7:0]</td></tr>
<tr><td>0x08</td><td>Receive Standard Producer Ring Consumer Index</td><td colspan="2">Receive Return Ring 1 Producer Index</td></tr>
<tr><td>0x0C</td><td>Receive Return Ring 2 Producer Index</td><td colspan="2">Receive Return Ring 3 Producer Index</td></tr>
<tr><td>0x10</td><td>Send BD Consumer Index</td><td colspan="2">Receive Return Ring 0 Producer Index</td></tr>
<tr><td>0x14</td><td>Reserved 0x0</td><td colspan="2">Receive Jumbo Producer Ring Consumer Index</td></tr>
</table>

The status tag field contains a unique 8-bit tag value in bits 7:0 when the status tagged status mode bit of the miscellaneous Host Control register 0x68 is set to 1. The status tag can be returned to mailbox 0 register 0x200 in field 31:24 by the host driver. When the remaining mailbox 0 register bits 23:0 are written to 0, the tag field of mailbox 0 is compared with the tag field of the last status block to be DMAed into host memory. If the tag returned is not equivalent to the tag of the last status block DMAed to the host, the interrupt state is entered.

Receive return ring 0 is the default return ring. If RSS is disabled all packets are assigned to this default ring.

There is no status block data structure in the controller memory space, but the host can access the current index through the register space.

Misc BD Memory

The Misc BD memory has been increased to 10 KB. The miscellaneous BD memory and the TX MBUF memory are physically the same memory, but a partition is hardwired. The miscellaneous BD memory holds four structures:

- On-chip send BD cache
- On-chip standard receive BD cache
- On-chip jumbo receive BD cache
- Software gencomm area (driver/firmware communication shared memory area)

Device Driver Interface

There are minimal driver interface modifications to support jumbo frames.

Send Interface

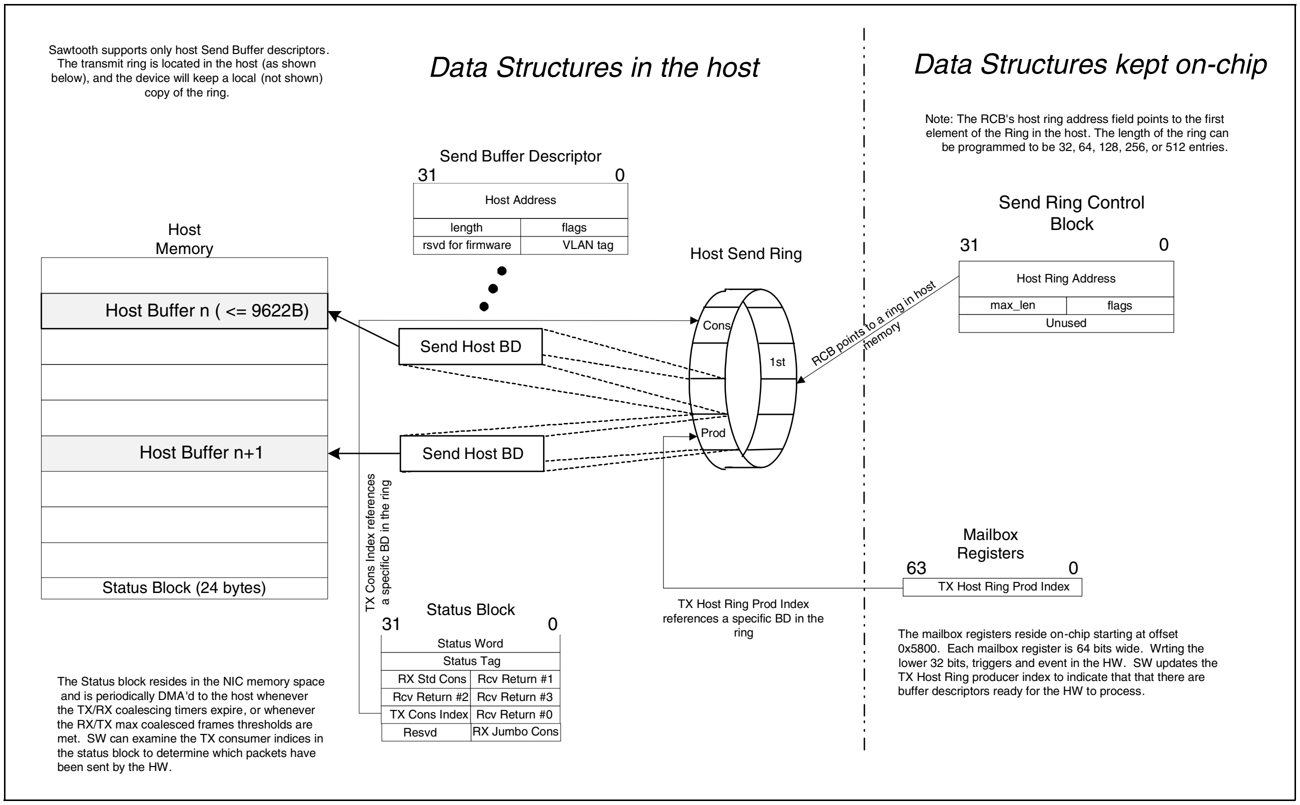
The driver essentially does not see any change in the send interface due to the jumbo frame feature. The only difference is that the stack is now allowed to construct larger frames (i.e., up to 9622 bytes long) out of send buffers.

Note: The controller is able to handle a single send buffer of length > 4K bytes and all the way up to 9622 bytes.

As in the case with previous controllers, the driver maintains a buffer descriptor ring (send ring), which allows for the "gather" management of transmit frame data. The production of send side packets by the driver is communicated to the controller via the Send Producer Ring Index Mailbox register. An update of the send BD ring producer index mailbox triggers the controller to begin DMA of the corresponding buffer descriptor. The consumption of send side packets by the controller is communicated back to the driver via the send consumer index, which is returned in the status block and periodically DMAed to the host by the controller.

A conceptual diagram of the Send Interface is shown in Figure 27 below.

Figure 27: Send Driver Interface



Sawtooth supports only host Send Buffer descriptors. The transmit ring is located in the host (as shown below), and the device will keep a local (not shown) copy of the ring.

The Status block resides in the NIC memory space and is periodically DMA'd to the host whenever the TX/RX coalescing timers expire, or whenever the RX/TX max coalesced frames thresholds are met. SW can examine the TX consumer indices in the status block to determine which packets have been sent by the HW.

Note: The RCB's host ring address field points to the first element of the Ring in the host. The length of the ring can be programmed to be 32, 64, 128, 256, or 512 entries.

The mailbox registers reside on-chip starting at offset 0x5800. Each mailbox register is 64 bits wide. Wrting the lower 32 bits, triggers and event in the HW. SW updates the TX Host Ring producer index to indicate that that there are buffer descriptors ready for the HW to process.

Receive Interface

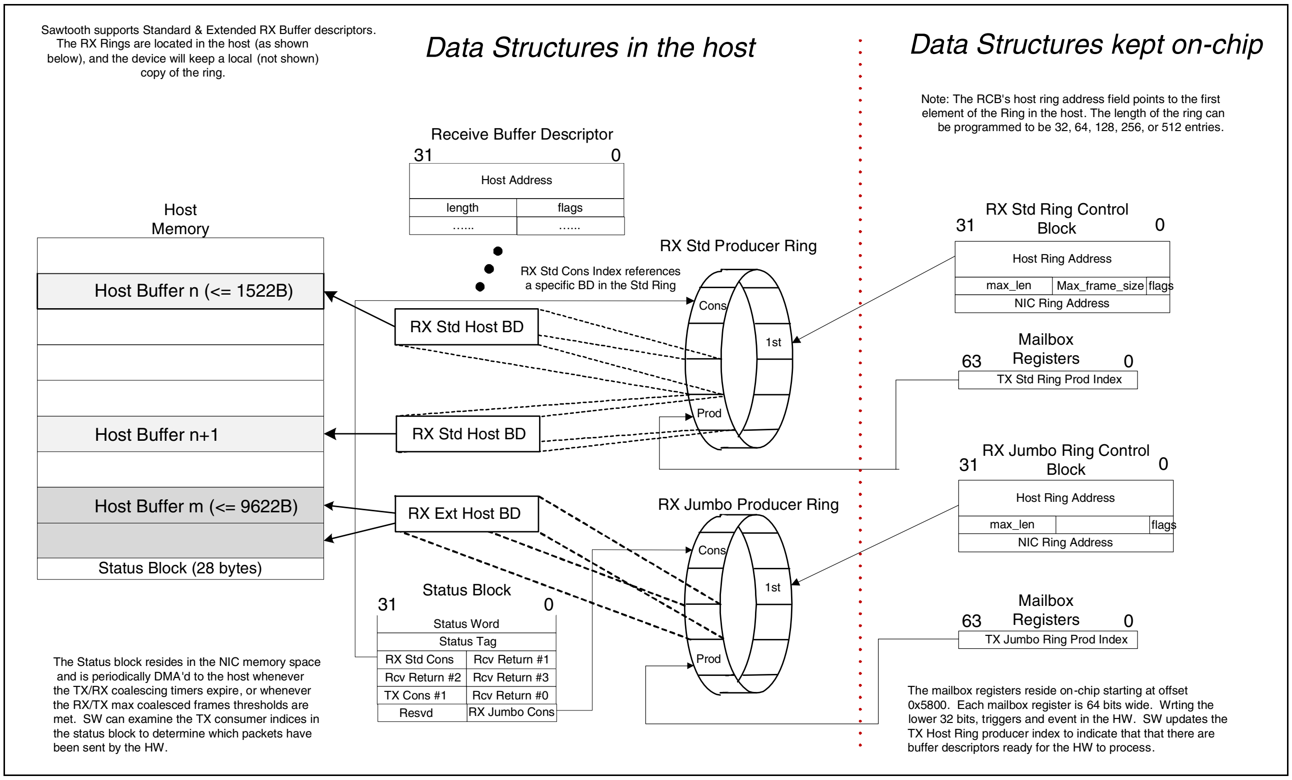
As mentioned previously, the receive side has added another producer ring called the jumbo producer ring. This means the driver maintains two buffer descriptor rings (receive producer rings) that provide free data buffer space into which the controller can place received frame data. The production of receive-side buffers by the driver is communicated to the device via the rEceive Producer Ring Index(s) Mailbox registers. An update of a receive BD ring producer index mailbox triggers the device to begin DMA of the corresponding buffer descriptor(s). The consumption of receive side buffers by the device is communicated to the driver via the receive consumer index, which is returned via the status block and periodically DMAed to the host by the controller.

The controller fetches RX BDs in anticipation of RX frames and caches the BDs in the controller. The cached BDs are stored in the Misc BD Memory. There is a set of rules that govern placement of packets in buffers:

- All standard sized frames, that is, frames less than or equal to the Max Frame Size field of the standard RCB, typically 1522 bytes or less, are placed in buffers retrieved from the RX standard producer ring.
- Hence the host must ensure that buffers pointed to by the standard BDs are at least of that size.
- In case the host creates a standard ring buffer that is smaller than Max Frame Size and an RX frame larger than the buffer size (but <= Max Frame Size) arrives, the controller attempts to place the frame in such a buffer and ends up truncating the frame.
- RX frames larger than Max Frame Size are placed in buffers retrieved from the RX jumbo producer ring.
- Extended buffers placed in the jumbo producer ring must provide an aggregated space of 9622 bytes or higher. Otherwise, jumbo frames might be truncated by the controller during placement.

A conceptual diagram of the Receive Producer Interface is shown in Figure 28.

Figure 28: Receive Producer Interface



Sawtooth supports Standard & Extended RX Buffer descriptors. The RX Rings are located in the host (as shown below), and the device will keep a local (not shown) copy of the ring.

Note: The RCB's host ring address field points to the first element of the Ring in the host. The length of the ring can be programmed to be 32, 64, 128, 256, or 512 entries.

RX Std Cons Index references a specific BD in the Std Ring

Status Block

31 0

Status Word	
Status Tag	
RX Std Cons	Rcv Return #1
Rcv Return #2	Rcv Return #3
TX Cons #1	Rcv Return #0
Resvd	RX Jumbo Cons

The Status block resides in the NIC memory space and is periodically DMA'd to the host whenever the TX/RX coalescing timers expire, or whenever the RX/TX max coalesced frames thresholds are met. SW can examine the TX consumer indices in the status block to determine which packets have been sent by the HW.

The mailbox registers reside on-chip starting at offset 0x5800. Each mailbox register is 64 bits wide. Wrting the lower 32 bits, triggers and event in the HW. SW updates the TX Host Ring producer index to indicate that that there are buffer descriptors ready for the HW to process.

The BCM5725/BCM5762/BCM57767 fills up receive buffers with RX frame data and returns the buffers to the host via receive return rings. The members of a return ring are simply RX buffer descriptors. Both types of descriptors, that is, standard or extended, are returned to the same return ring. As mentioned previously, due to practical reasons, extended buffer descriptors are truncated before posting into a return ring so that the actual size of all BDs posted to the return ring are the same (see "Receive Return Ring(s)" on page 125).

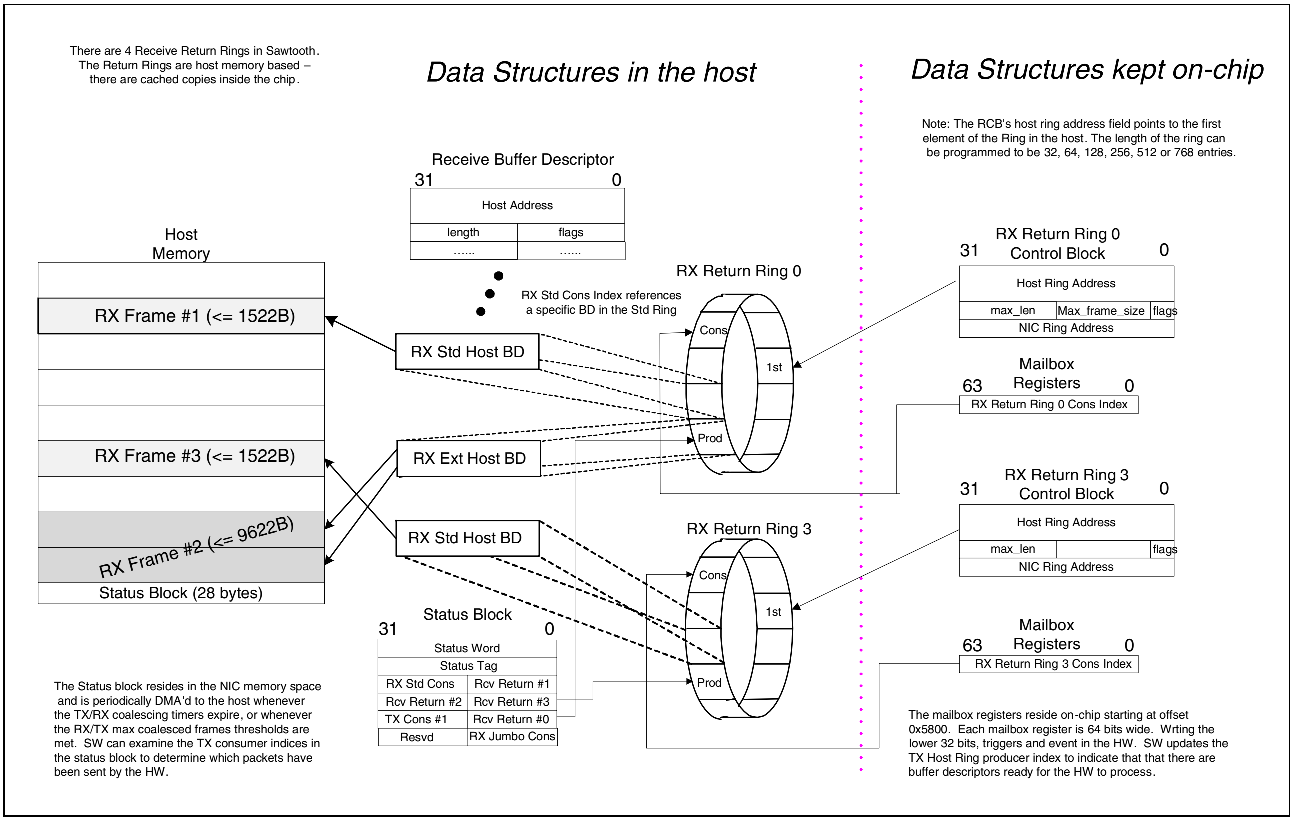
Due to RSS, there are four return rings in the BCM5725/BCM5762/BCM57767. However, when RSS is disabled all RX frames are posted to Ring 0 while the other three rings remain inactive. In any case, standard and jumbo frames may be intermixed in any return ring as the order of placement strictly follows the order of frame reception.

The controller maintains four producer indexes associated with the four return rings. The availability of receive-side packets by the device is communicated to the driver via the receive return ring producer indices, which is delivered via the status block periodically when it is DMAed to the host by the controller.

The consumption of receive return packets by the driver is communicated to the controller via the receive return consumer ring index mailbox register.

A conceptual diagram of the Receive Return Interface is shown in Figure 29.

Figure 29: Receive Return Interface


There are 4 Receive Return Rings in Sawtooth. The Return Rings are host memory based – there are cached copies inside the chip.
Data Structures in the host
Data Structures kept on-chip
Note: The RCB's host ring address field points to the first element of the Ring in the host. The length of the ring can be programmed to be 32, 64, 128, 256, 512 or 768 entries.
Receive Buffer Descriptor
31
0
Host Address
length
flags
......
......
Host Memory
RX Frame #1 (<= 1522B)
RX Frame #3 (<= 1522B)
RX Frame #2 (<= 9622B)
Status Block (28 bytes)
RX Std Cons Index references a specific BD in the Std Ring
RX Std Host BD
RX Ext Host BD
RX Std Host BD
RX Return Ring 0
Cons
1st
Prod
RX Return Ring 3
Cons
1st
Prod
RX Return Ring 0 Control Block
31
0
Host Ring Address
max_len
Max_frame_size
flags
NIC Ring Address
Mailbox Registers
63
0
RX Return Ring 0 Cons Index
RX Return Ring 3 Control Block
31
0
Host Ring Address
max_len
flags
NIC Ring Address
Mailbox Registers
63
0
RX Return Ring 3 Cons Index
Status Block
31
0
Status Word
Status Tag
RX Std Cons
Rcv Return #1
Rcv Return #2
Rcv Return #3
TX Cons #1
Rcv Return #0
Resvd
RX Jumbo Cons
The Status block resides in the NIC memory space and is periodically DMA'd to the host whenever the TX/RX coalescing timers expire, or whenever the RX/TX max coalesced frames thresholds are met. SW can examine the TX consumer indices in the status block to determine which packets have been sent by the HW.
The mailbox registers reside on-chip starting at offset 0x5800. Each mailbox register is 64 bits wide. Wrting the lower 32 bits, triggers and event in the HW. SW updates the TX Host Ring producer index to indicate that that there are buffer descriptors ready for the HW to process.

Large Segment Offload (LSO/TSO)

LSO may create jumbo frames instead of standard sized frames. This is accomplished by programming the MSS field of the send BD (see “Send Buffer Descriptor” on page 126).

The legacy NetXtreme design has limitations in LSO hardware. In the case of the BCM5725/BCM5762/BCM57767, these limitations also exist. Below is the list of such limitations:

- MSS may not be less than 8 bytes
- LSO packet must be a TCP packet
- IP length field must not be incorrect
- TCP length field must not be incorrect
- Total offloaded TCP payload length must be greater than the MSS selected by SBD
- For all LSO segments, the SBD flag bit 8 and 9 (CPU pre-DMA and CPU post-DMA fields) must be set.
- LSO packet may not be IEEE 802.3 format with LLC and SNAP headers
- The total length of IP header (including IP option for IPv4, extension headers for IPv6) and TCP header (including TCP option) may not be more than 200 bytes. Note: post_dma_proc can support only up to 2 MBUFs worth of packet header data. 1st Mbuf gives: 128, Mbuf-header (8B), Frame Header field (40B) = 80B; 2nd Mbuf gives: 128, Mbuf Header (8B) = 120B. Hence the total space for all headers, which includes all L2/L3/L4 combined, and options cannot exceed 200 bytes.
- L2 header must be contained within one SBD (the first one)

- IP header (IPv4 or IPv6), including IP Options, must be contained within a single SBD
- Hdrlen[7:0] field must be correct in SBD
- DONT_GEN_CRC field must not be set in a SBD for LSO packet
- SNAP field must not be set in SBD for LSO packet
- TCP header, including TCP Options, must be contained within a single SBD

The Read DMA (RDMA) engine cannot support LSO packets with the above listed attributes. Any such LSO configurations may cause the RDMA engine to lockup and/or exhibit abnormal behavior.

List of Affected Registers

The following registers are either added or modified in the BCM5725/BCM5762/BCM57767 to support jumbo frames:

Receive BD Jumbo Producer Ring Index (High Priority Mailbox) Register (offset: 0x270)

Table 39: Receive BD Jumbo Producer Ring Index (High Priority Mailbox) Register (offset: 0x270)

Name	Bits	Access	Default Value	Description
Received BD jumbo Producer Ring Index	7:0	RW	0	The Receive BD Extended Producer Ring Index register contains the index of the next buffer descriptor for the extended producer ring that will be produced in the host for the controller to DMA into controller memory. Host software writes this register whenever it updates the extended producer ring. This register must be initialized to 0.

Receive Data and Receive BD Ring Initiator Status Register (0x2404)

Bit 2: The received frame's size exceeds the capacity of the standard receive BD and jumbo receive BD ring is disabled.

Jumbo Producer Ring Host Address High Register (offset: 0x2440)

Table 40: Jumbo Producer Ring Host Address High Register (offset: 0x2440)

Name	Bits	Access	Default Value	Description
Host Address High	31:0	RW	0	The host ring address is the host address of the first ring element. The host ring address is in host address format.

Jumbo Producer Ring NIC Address Register (offset: 0x2444)

Table 41: Jumbo Producer Ring NIC Address Register (offset: 0x244C)

Name	Bits	Access	Default Value	Description
Host Address Low	31:0	RW	0	The host ring address is the host address of the first ring element. The host ring address is in host address format.

Jumbo Producer Length/Flags Register (offset: 0x2448)

Table 42: Jumbo Producer Length/Flags Register (offset: 0x2448)

Name	Bits	Access	Default Value	Description
Max Length	31:16	RW	0	Specifies the number of entries for Jumbo ring based on bit-mask
Reserved	15:2	R/O	0	–
Disable Ring	1	RW	0	Set to disable the use of the ring
Reserved	0	RO	0	Set to use the extended receive buffer descriptors

Jumbo Producer Ring NIC Address Register (offset: 0x244C)

Table 43: Jumbo Producer Ring NIC Address Register (offset: 0x244C)

Name	Bits	Access	Default Value	Description
NIC Address	31:0	RW	0	The NIC ring address is the controller address of the first ring element.
Note: Driver must write 0x00007000 to this register for the BCM5725/BCM5762/BCM57767 controllers. Unlike most legacy NetXtreme controllers, the hardware default value cannot be used in this register.				

Receive Diagnostic Data and Receive BD Ring Initiator Local NIC Jumbo Receive BD Consumer Index (offset: 0x2470)

This set of registers keeps track of the current DMAs queued to move receive data from the controller to the host. The receive data and receive BD initiator maintains the state of the indices by keeping two local copies, a copy of the controller's return ring producer index and a copy of the controller's receive BD consumer index. The local return ring producer index is set to the value placed in the DMA descriptor. The local controller receive return consumer index is also set to the value placed in the DMA descriptor.

Table 44: Receive Diagnostic Data and Receive BD Ring Initiator Local NIC Jumbo Receive BD Consumer Index (offset: 0x2470)

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0	Reserved
Local Received BD Jumbo Producer Ring consumed Index	7:0	RO	0	Current Jumbo Received BD Index consumed by RDI. It means that the RBD index have been used by incoming RX packets.

Receive BD Initiator Local NIC Jumbo Receive BD Producer Index (offset: 0x2C08)

Table 45: Receive BD Initiator Local NIC Jumbo Receive BD Producer Index (offset: 0x2C08)

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0	Reserved
Local Received BD Jumbo Producer Ring requested Index	7:0	RO	0	Current Jumbo Received BD Index requested by RBDI for BD fetching. Note that this index is different from MB producer index and also different from the index indicated by RBDC.

Jumbo Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C1C)

Table 46: Jumbo Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C1C)

Name	Bits	Access	Default Value	Description
Reserved	31:10	RO	0	–
BD Number	9:0	RW	0	Number of buffer descriptors indicated by the receive producer index for the DMA engine to initiate a transfer of buffer descriptors for replenishing the ring.

NIC Jumbo Receive BD Producer Index Register (offset: 0x3008)

Table 47: NIC Jumbo Receive BD Producer Index Register (offset: 0x3008)

Name	Bits	Access	Default Value	Description
Reserved	31:9	RO	0	–
NIC Jumbo Receive BD Producer Index	8:0	RW	–	–

NIC Receive BD Consumer Index Register (offset: 0x3C50–0x3C58)

These three registers are shared by the Receive BD Completion and the Receive Data and Receive BD Initiator state machines. They are used to keep track of the receive BDs that have been DMAed to the controller.

Table 48: NIC Receive BD Consumer Index Register (offset: 0x3C50–0x3C58)

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0	–
NIC Jumbo Receive BD Producer Index	7:0	RW	–	Current Jumbo Received BD have been fetched by RDMA module and are available for incoming RX packets.

NIC Diag Receive Return Ring BD 0 Index Register (offset: 0x3C80)

Table 49: NIC Diag Receive Return Ring BD 0 Index Register (offset: 0x3C80)

Name	Bits	Access	Default Value	Description
Local Diagnostic Receive Return Ring 0 index value	9:0	RW	0	Current Receive Return Ring 0 index value in HC module before applying RCB bit-mask value. The maximum value is 1023 for the BCM5725/ BCM5762/BCM57767. This value will be masked by the RCB bit-mask in WDMA module before be DMAed in status block.

Receive BD Jumbo Producer Ring Index Register (offset: 0x5870-5877)

Table 50: Receive BD Jumbo Producer Ring Index Register (offset: 0x5870-5877)

Name	Bits	Access	Default Value	Description
Received BD jumbo Producer Ring Index	7:0	RW	0	The Receive BD Extended Producer Ring Index register contains the index of the next buffer descriptor for the extended producer ring that will be produced in the host for the controller to DMA into controller memory. Host software writes this register whenever it updates the extended producer ring. This register must be initialized to 0.

RDI Mode Register (Offset: 0x2400)

Table 51: RDI Mode Register (Offset: 0x2400)

Name	Bits	Access	Default Value	Description
Large RX Ring Sizes	16	RW	0	When this bit is 1, following are the maximum allowable Receive Ring sizes: Standard Producer Ring == 2048 Jumbo Producer Ring == 1024 Receive Return Ring == 4096 When this bit is 0, following are the maximum allowable Receive Ring sizes: Standard Producer Ring == 512 Jumbo Producer Ring == 256 Receive Return Ring == 1024

NIC Ring Addresses

The NIC Ring Address field of each RCB has hardware default values. These values normally need not be programmed by the driver. However, the BCM5725/BCM5762/BCM57767 does require the driver to set the receive standard producer and the receive jumbo producer values as listed in the table below. Without the driver setting these values, the hardware does not operate properly.

Table 52: NIC Ring Addresses

RCB Type	NIC Ring Address Value
Send RCB0	Leave as hardware default
Receive Standard Producer RCB0	0x6000
Receive Jumbo Producer RCB0	0x7000

Summary of Register Settings to Support Jumbo Frames

- Standard RCB (Ring Control Block):

0x2450: Host address high [31:0]
0x2454: Host address low [31:0]
0x2458: [31:16]: Standard ring size (power of 2)
    [5:2]: std_max_packet_size = 0x5EE (1518 decimal)
    [1:0] = 0
0x245C: Standard NIC address = 0x6000 (do not use hardware default value as with legacy NetXtreme controllers)

The NIC address is a controller-internal memory address where the controller caches a portion of a ring to achieve faster, higher performance access to buffer descriptors.

- Jumbo RCB:

0x2440: Host address high [31:0]
0x2444: Host address low [31:0]
0x2448: [31:16]: Jumbo ring size (power of 2)
    [15:2] = 0
    [1:0] = 0
0x244C: Jumbo NIC address = 0x7000 (do not use hardware default value as with legacy NetXtreme controllers)

- Receive Return RCB:

Only the host address and ring size are applicable to the receive return RCB.

- Receive MBUF low water mark 0x4414 = 0x7E (program to this value only when jumbo enabled)
  Receive MBUF high water mark 0x4418 = 0xEA (program to this value only when jumbo enabled)
  Read DMA watermark register 0x4410 = 0x0
- Standard replenish threshold register 0x2C18 is typically 1/8 of total receive BDs in host memory.
  Jumbo replenish threshold register 0x2C1C is typically 1/8 of total Receive BDs in host memory.
- EMAC MTU register 0x43C: Program this register based on max packet size.

Scatter/Gather

Most often, the host software requests the NIC to transmit a frame that spans several physical fragments that are arbitrary in size and buffer alignment. This requires the Ethernet controller to gather all these fragments during a DMA process into a continuous data stream for transmission.

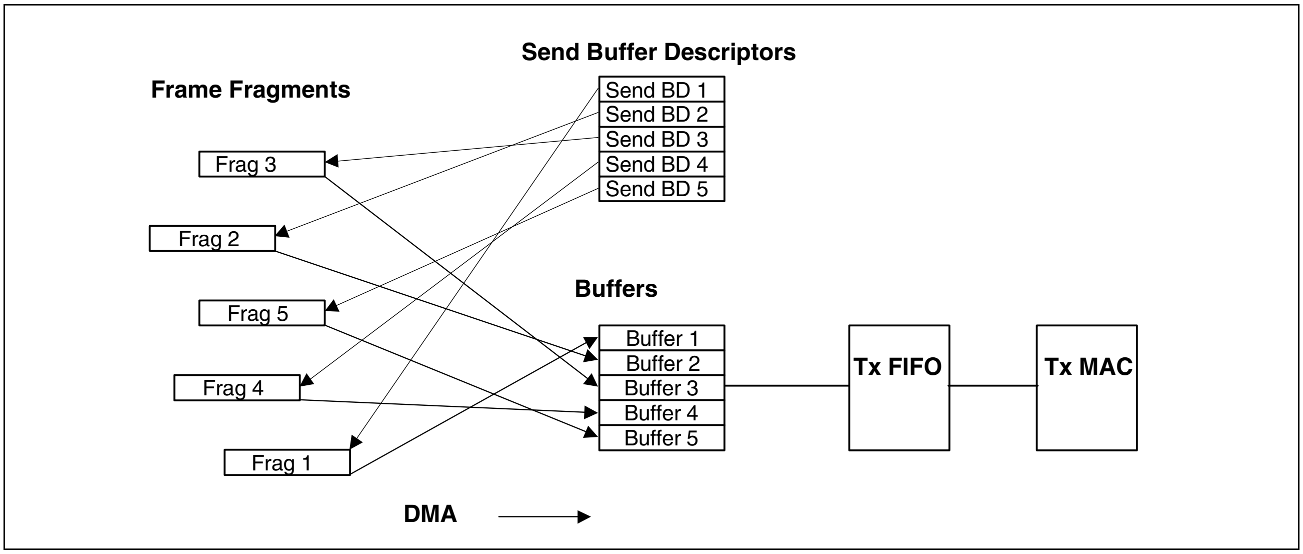
The ability to scatter/gather a frame lessens the restriction on the host software and increases overall system performance.

> ***Example:*** A TCP/IP protocol stack could preconstruct the MAC and IP headers in separate buffers that are combined with the payload to form a complete frame. Since the header data are fairly constant during a TCP or UDP session, the stack could use the same header buffers for the next frame.

The Ethernet controller uses a buffer descriptor for describing a physical fragment. There are two types of buffer descriptors; the Receive MAC processes receive buffer descriptors (Receive BD) and the Transmit MAC processes send buffer descriptors (Send BD).

Figure 30 illustrates the relationship between a frame consisting of multiple fragments and their corresponding send buffer descriptors.

Figure 30: Scatter Gather of Frame Fragments



To transmit a frame, the host software sets up consecutive buffer descriptors in a send ring. Each buffer descriptor describes a physical fragment of a frame. As an example, the above figure illustrates a frame consisting of five fragments that are scattered throughout host memory. Frag1, the first fragment, is at the start of the frame, and Frag5, the last fragment, is at the end of a frame. For each fragment, there is a corresponding buffer descriptor, SendBd1 through SendBd5. These buffer descriptors must be initialized in the send ring in a consecutive order, SendBd1 to SendBd5. The last send buffer descriptor of a frame must have the PACKET_END bit of Send BD Flags field set to indicate the end of a frame.

VLAN Tag Insertion

The Ethernet controller is capable of inserting 802.1Q-compliant VLAN tags into transmitted frames and extracting the VLAN tags from received frames. A frame containing the 802.1Q VLAN tag has the value TPID (Tag Protocol Identifier) value in the EtherType field followed by a 16-bit TCI (Tag Control Information) field, which is made up of one CFI bit, 3 802.1P priority bits, and a 12-bit VLAN ID. The original 16-bit EtherType/ Length field follows the TCI field.

Table 28 on page 101 shows the frame format with 802.1Q VLAN tag inserted.

The Ethernet controller allows the host software to enable or disable tag insertion on a per-packet basis. To send a frame with a VLAN tag, the host software must initialize the first send buffer descriptor of a packet with the VLAN tag value and set the VLAN_TAG bit of Send BD Flags field (see "Send Rings" on page 107).

TX Data Flow Diagram

Figure 31 on page 141 illustrates how a frame, consisting of several fragments, is sent from the host to the NIC and onto the network. For simplicity, the diagram depicts the operation of a single ring.

1. The host software calls a system API to retrieve the three physical fragments of the frame. It initializes the next three send buffer descriptors to point to each fragment. The send buffer descriptors reside in host memory. Internally, the host software maintains the ring's producer index. In this case, the producer index is incremented by three because there are three fragments.
2. The host software updates the send ring producer index by writing the producer index value to Send Ring Producer Index Mailbox at offset 0x300 for host standard and offset 0x5900 for indirect mode. The mailbox update triggers the Ethernet controller to process the send buffer descriptors.
3. The send buffer descriptors are DMAed to the ring's staging area in device memory as indicated in the RCB.
4. The Ethernet controller DMAs the frame (as described in the descriptors) to its internal memory for transmission.
5. Internally, the Ethernet controller maintains the send ring's consumer index, which is incremented as it processes the descriptors.
6. The new consumer index is written to the status block in NIC memory (see "Status Block" on page 82).
7. The status block is DMAed to host memory. This DMA is subject to host coalescing, and the NIC may generate an interrupt at this point.

Figure 31 and Figure 32 on page 142 show the basic driver flow to send a packet.

Figure 31: Transmit Data Flow

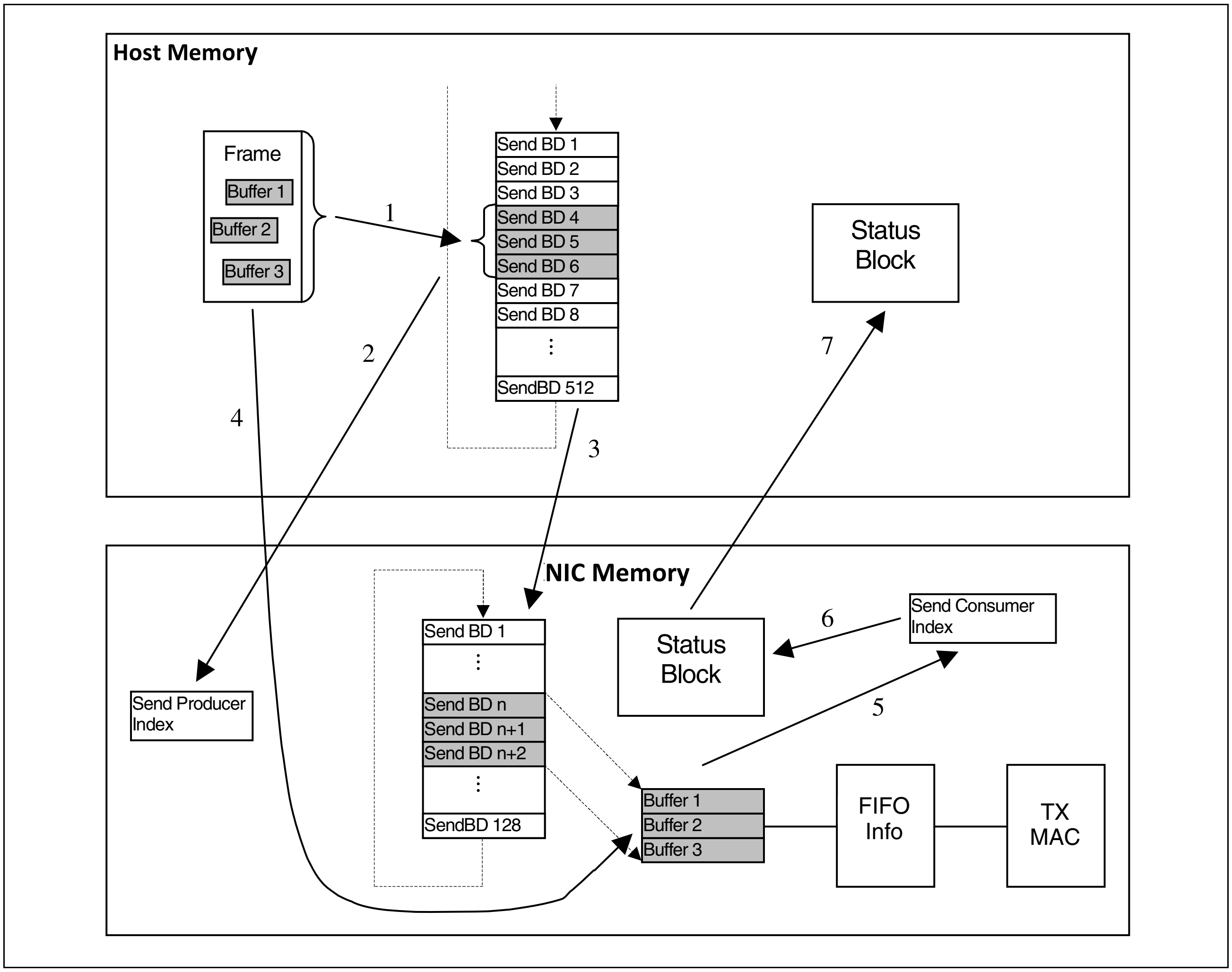
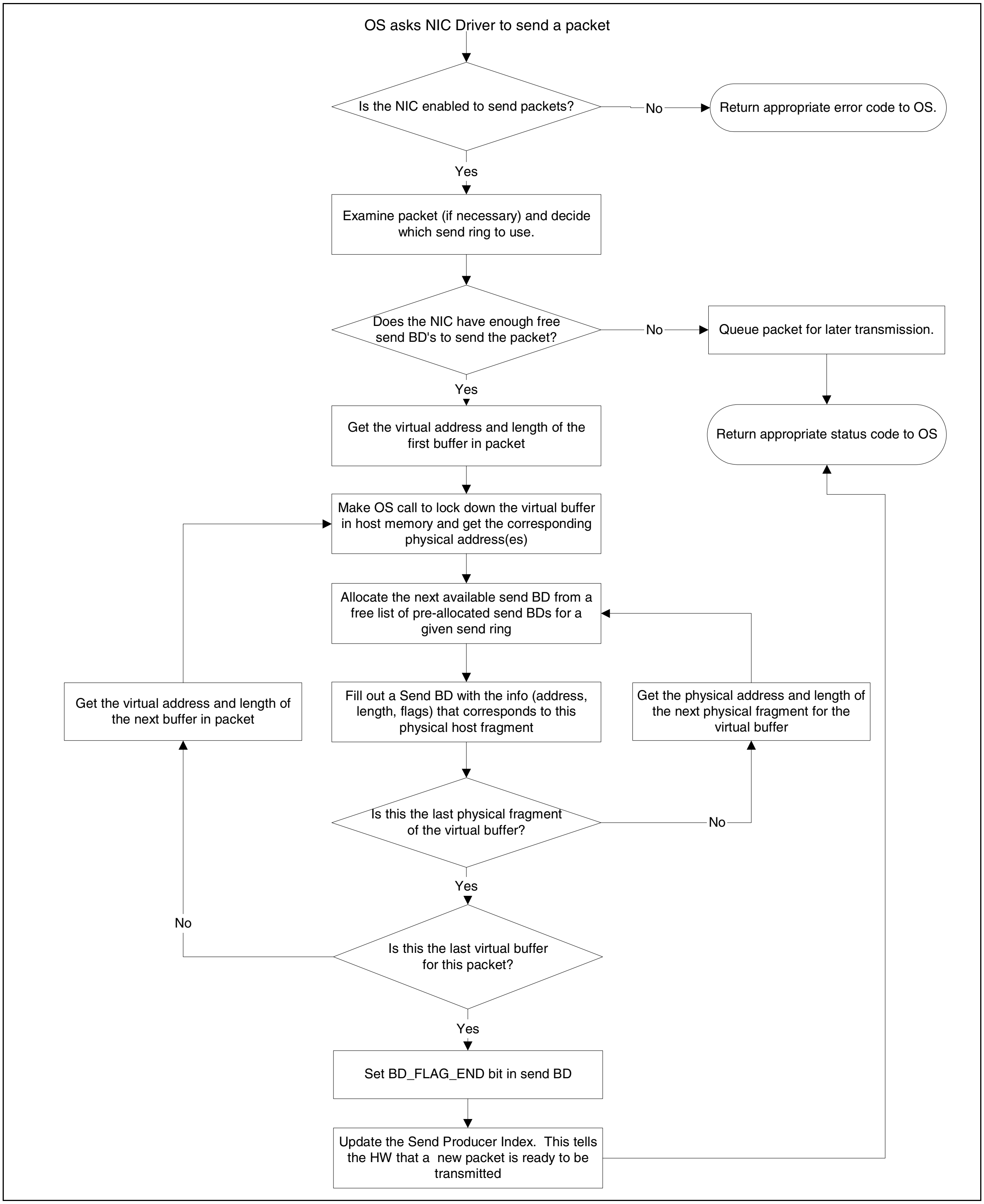


Figure 32: Basic Driver Flow to Send a Packet


OS asks NIC Driver to send a packet
Is the NIC enabled to send packets?
No
Return appropriate error code to OS.
Yes
Examine packet (if necessary) and decide which send ring to use.
Does the NIC have enough free send BD's to send the packet?
No
Queue packet for later transmission.
Yes
Return appropriate status code to OS
Get the virtual address and length of the first buffer in packet
Make OS call to lock down the virtual buffer in host memory and get the corresponding physical address(es)
Allocate the next available send BD from a free list of pre-allocated send BDs for a given send ring
Get the virtual address and length of the next buffer in packet
Fill out a Send BD with the info (address, length, flags) that corresponds to this physical host fragment
Get the physical address and length of the next physical fragment for the virtual buffer
Is this the last physical fragment of the virtual buffer?
No
Yes
No
Is this the last virtual buffer for this packet?
Yes
Set BD_FLAG_END bit in send BD
Update the Send Producer Index. This tells the HW that a new packet is ready to be transmitted

Reset

A hardware reset initiated by the PCI reset signal will initialize all PCI configuration registers and device MAC registers to their default values. The driver reset via the Core Clock Blocks Reset bit (see “Miscellaneous Configuration Register (offset: 0x6804)” on page 523) will also initialize all non-sticky registers to their default values. The content of the device internal memory remains unchanged after warm reset (any reset with the power supplied to the device).

At the end of the reset, the on-chip RX RISC executes a small on chip ROM code. This code loads an executable image contained in an attached NVRAM and referred to as the boot code. This boot code allows at least the following fields to be initialized to different values to support product variations (for additional details, see Section 3: “NVRAM Configuration,” on page 65).

- Vendor ID
- Device ID
- Subsystem Vendor ID
- Subsystem Device ID
- Possible PHY initialization

The boot code may have additional functionality such as PXE that must be acquiesced while the host software is running.

> ***Example:*** An NDIS driver issues a device reset via the Core Clock Blocks Reset bit (see “Miscellaneous Configuration Register (offset: 0x6804)” on page 523). After the reset is completed, the RX RISC begins executing the boot code as if the power was first applied to the device. However, the NDIS driver must have a mechanism to prevent the PXE driver from running and the boot code must be able to distinguish between a power-on reset and a reset initiated by the host software. The host software and the boot code could implement a reset handshake by using shared memory at offset 0x0b50 as a software mailbox (see “Firmware Mailbox” on page 208).

The BCM5725/BCM5762/BCM57767 Ethernet controller supports a boot code mechanism known as “self-boot”. For self-boot the boot code image is stored in internal ROM rather than in an external NVRAM. So there is no loading of a boot code image from external NVRAM when resetting in the self-boot scenario.

However, there may still be a very small external NVRAM device which may contain some configuration items and possibly boot code “patches” to be applied to the ROM'd self-boot boot code. Refer to the following Broadcom Application Notes for additional self-boot and general NVRAM access information:

- 5754X_5787X-AN10X-R “Self Boot Option”
- NetXtreme-AN40X-R “NetXtreme/NetLink Software Self-Boot NVRAM”
- NetXtreme-AN50X-R “NetXtreme®/NetLink® NVRAM Access”

MAC Address Setup/Configuration

The MAC address registers, starting at offset 0x0410, contain the MAC addresses of the NIC. These registers are usually initialized with a default MAC address extracted from the NIC NVRAM when it is first powered up. The host software may overwrite the default MAC address by writing to the MAC registers with a new MAC address. Table 53 illustrates the MAC register format.

The BCM5725/BCM5762/BCM57767 Ethernet controller allows a NIC to have up to four MAC addresses (offset 0x410–0x42F) that are used for hardware packet reception filtering. However, most host software will initialize the registers of the four MAC addresses to the same MAC address since a NIC usually has only one MAC address.

When flow control is enabled on the Ethernet controller, the MAC Address 0 is used as the source address for sending PAUSE frames (see “Pause Control Frame” on page 681).

Table 53: Mac Address Registers

Register Name	*Offset*	*31*	*24*	*23*	*16*	*15*	*8*	*7*	*0*
Mac_Address_0	0x0410	Unused				Octet 0		Octet 1	
	0x0414	Octet 2		Octet 3		Octet 4		Octet 5	
Mac_Address_1	0x0418	Unused				Octet 0		Octet 1	
	0x041c	Octet 2		Octet 3		Octet 4		Octet 5	
Mac_Address_2	0x0420	Unused				Octet 0		Octet 1	
	0x0424	Octet 2		Octet 3		Octet 4		Octet 5	
Mac_Address_3	0x0428	Unused				Octet 0		Octet 1	
	0x042c	Octet 2		Octet 3		Octet 4		Octet 5	

Packet Filtering

Multicast Hash Table Setup/Configuration

The MAC hash registers are used to help discard unwanted multicast packets as they are received from the external media. The destination address is fed into the normal CRC algorithm in order to generate a hash function. The most significant bits of the CRC are then used without any inversion in reverse order to index into a hash table, which is comprised of these MAC hash registers. If the CRC is calculated by shifting right, then the right-most bits of the CRC can be directly used with no additional inversion or bit swapping required. See “Ethernet CRC Calculation” for more details on the CRC algorithm.

All four MAC hash registers are used so that register 1 bit-32 is the most significant hash table entry and register 4 bit-0 is the least significant hash table entry. This follows the normal big-endian ordering used throughout the Ethernet controller. Since there are 128 hash table entries, 7 bits are used from the CRC. When hash table is extended to 256 entries, 8 bits from the CRC will be used as hash index.

The MAC hash registers are ignored if the receive MAC is in promiscuous mode.

Ethernet CRC Calculation

The Ethernet controller uses the standard 32-bit CRC required by the Ethernet specification as its FCS in all packets. The checksum is the 32-bit remainder of the polynomial division of the data taken as a bit stream of polynomial coefficients and a predefined constant, which also represents binary polynomial coefficients. The checksum is optionally appended most-significant bit first to a packet, which is to be sent down the wire. At the receiving side, the division is repeated on the entire packet including the CRC checksum. The remainder is compared to a known constant. For details on the mathematical basis for CRC checksums, see Tanenbaum's Computer Networks, Third Edition, c1996.

The 32-bit CRC polynomial divisor is shown below:

```
x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1
```

Generating CRC

The following steps describe a method to calculate the CRC with the resulting 32-bit quantity having reversed bit order (i.e., most significant bit x31 of the remainder is right-most bit). The data should be treated as a stream of bytes. Set remainder to 0xFFFFFFFF. For each bit of data starting with least-significant bit of each byte:

1. If right-most bit (bit-0) of the current remainder XORed with the data bit equal 1,
then remainder = (remainder shifted right one bit) XOR 0xEDB88320,
else remainder = (remainder shifted right one bit).

2. Invert remainder such that remainder = ~remainder.
Remainder is CRC checksum.
Right-most byte is the most significant and is to be sent first.
Swap bytes of CRC if big-endian byte ordering is desired.

Checking CRC

The following steps describe a method to check a stream of bytes, which has a CRC appended.

1. Set remainder to 0xFFFFFFFF.

2. For each bit of data starting with least-significant bit of each byte:
If right-most bit (bit-0) of the current remainder XORed with the data bit equal 1,
then remainder = (remainder shifted right one bit) XOR 0xEDB88320,
else remainder = (remainder shifted right one bit).

3. Remainder should equal magic value 0xDEBB20E3 if CRC is correct.

Initializing the MAC Hash Registers

The 128-bit multicast hash table is treated as a single object occupying four Ethernet controller registers starting at offset 0x0470 (see Table 54). The 128-bit value follows the big-endian ordering required by Ethernet controller. Thus, the most significant 32-bit of the 128-bit value resides in Mac_Hash_Register_0 at offset 0x0470 and the least significant 32-bit resides in Mac_Hash_Register_3 at offset 0x047c.

Host software can enable the reception of all multicast frames including broadcast frames by setting all four multicast hash registers to 0xFFFFFFFF.

Table 54: Multicast Hash Table Registers

Register Name	*Offset*	*Description*
Mac_Hash_Register_0	0x0470	Most significant 32-bit of the 128-bit hash table
Mac_Hash_Register_1	0x0474	Bits 64:93 of the 128-bit hash table
Mac_Hash_Register_2	0x0478	Bits 32:63 of the 128-bit hash table
Mac_Hash_Register_3	0x047c	Least significant 32-bit of the 128-bit hash table

The following C code fragment illustrates how to initialize the multicast hash table registers. The code fragment computes the indices into hash table from a given list of multicast addresses and initializes the multicast hash registers.

```
Unsigned long HashReg[4];
Unsigned long j, McEntryCnt;
Unsigned char McTable[32][6];    // List of multicast addresses to accept.
// Initialize the McTable here.
McEntryCnt = 32;
// Initialize the multicast table registers.
HashReg[0] = 0;                  // Mac_Hash_Regsiter_0 at offset 0x0470.
HashReg[1] = 0;                  // Mac_Hash_Register_1 at offset 0x0474.
HashReg[2] = 0;                  // Mac_Hash_Register_2 at offset 0x0478.
HashReg[3] = 0;                  // Mac_Hash_Register_3 at offset 0x047c.
for(j = 0; j < McEntryCnt; j++)
{
    unsigned long RegIndex;
    unsigned long Bitpos;
    unsigned long Crc32;
    Crc32 = ComputeCrc32(McTable[j], 6);
    // The most significant 7 bits of the CRC32 (no inversion),
    // are used to index into one of the possible 128 bit positions.
    Bitpos = ~Crc32 & 0x7f;
    // Hash register index.
    RegIndex = (Bitpos & 0x60) >> 5;
    // Bit to turn on within a hash register.
    Bitpos &= 0x1f;
    // Enable the multicast bit.
    HashReg[RegIndex] |= (1 << Bitpos);
}
```

The following C routine computes the Ethernet CRC32 value from a given byte stream. The routine is called from the above code fragment.

```
// Routine for generating CRC32.
unsigned long
ComputeCrc32(
    unsigned char *pBuffer, // Buffer containing the byte stream.
    unsigned long BufferSize) // Size of the buffer.
{
    unsigned long Reg;
    unsigned long Tmp;
    unsigned long j, k;
    Reg = 0xffffffff;
    for(j = 0; j < BufferSize; j++)
    {
        Reg ^= pBuffer[j];
        for(k = 0; k < 8; k++)
        {
            Tmp = Reg & 0x01;
            Reg >>= 1;
            if(Tmp)
            {
                Reg ^= 0xedb88320;
                }
            }
    }
    return ~Reg;
}
```

Promiscuous Mode Setup/Configuration

The host software may enable promiscuous mode by setting the Promiscuous_Mode bit (bit 8) of the Receive_MAC_Mode register (offset 0x468). The Promiscuous_Mode bit defaults to disabled after reset, and host software must explicitly set this bit for promiscuous mode. In promiscuous mode of operation, the Ethernet controller accepts all incoming frames that are not filtered by the active receive rules regardless of the destination MAC address. In other words, the Ethernet controller operating in promiscuous mode ignores multicast and MAC address filtering ("Multicast Hash Table Setup/Configuration" on page 144 and "MAC Address Setup/Configuration" on page 144), but applies Receive Rules.

Broadcast Setup/Configuration

The host software may configure the Ethernet controller to discard the received broadcast frames by using two receive rules as defined below. The Ethernet controller parses all incoming frames according to these receive rules and discards those frames that have a broadcast destination address (see "Receive Rules Setup and Frame Classification" on page 97 for more details on setting up the receive rules).

The following is a sample of the two receive rules for discarding broadcast frames.

```
Rule1 Control: 0xc2000000               Rule1 Mask/Value: 0xffffffff
Rule2 Control: 0x86000004               Rule2 Mask/Value: 0xffffffff
```

Section 7: Device Control

Initialization Procedure

This section describes the initialization procedure for the MAC portion of the NetXtreme family of devices.

1. Enable MAC memory space decode and bus mastering. If the device has not been initialized previously (power-on reset), the host software must enable these bits to be able to issue the core clock reset in Step 6. Set the Bus_Master and Memory_Space bits in the PCI Configuration Space Command register (see PCI "Status and Command Register (Offset: 0x04) — Function 0" on page 245).
2. Disable interrupts. Set the Mask_Interrupt bit in the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256).
3. Write the T3_MAGIC_NUMBER (0x4B657654 = "KevT") to the device memory at offset 0xB50 to notify the boot code that the following reset is a warm reset (driver initiated core_clocks reset).
4. Acquire the NVRAM lock by setting the REQ1 bit of the Software Arbitration register (see "Software Arbitration Register (Offset 0x7020)" on page 559) and then waiting for the ARB_WON1 bit to be set.
5. Clear the Fast Boot Program Counter register (Offset 0x6894) and enable the Memory Arbiter as specified in Step 10. Also initialize the Misc Host Control register as specified in Step 11.
6. Reset the core clocks. Set the CORE_Clock_Blocks-Reset bit in the General Control Miscellaneous Configuration register (see "Miscellaneous Configuration Register (offset: 0x6804)" on page 523). The GPHY_Power_Down_Override bit (bit 26) and the Disable_GRC_Reset_on_PCI-E_Block bit (bit 29) should also be set to 1.
7. Wait for core-clock reset to complete. The core clock reset disables indirect mode and flat/standard modes. Software cannot poll the core-clock reset bit to deassert since the local memory interface is disabled by the reset. Driver should delay minimum of 1 millisecond at this point.
8. Disable interrupts. Set the Mask_PCI_Interrupt_Output bit in the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). The bit was reset after the core_clock reset and interrupts must be masked off again.
9. Enable MAC memory space decode and bus mastering. Set the Bus_Master and Memory_Space bits in the PCI Configuration Space Command register (see "Status and Command Register (Offset: 0x04) — Function 0" on page 245).
10. Enable the MAC memory arbiter. Set the Enable bit in the Memory Arbiter Mode register (see "Memory Arbiter Mode Register (offset: 0x4000)" on page 451).
11. Initialize the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256).
    a. Set Endian Word Swap (optional). When the host processor architecture is big-endian, the MAC may word swap data when acting as a PCI target device. Set the Enable_Endian_Word_Swap bit in the Miscellaneous Host Control register.
    b. Set Endian Byte Swap (optional). When the host processor architecture is big-endian, the MAC may byte swap data when acting as a PCI target device. Set the Enable_Endian_Byte_Swap bit in the Miscellaneous Host Control register.

c. Enable the indirect register pairs (see "Indirect Mode" on page 164). Set the Enable_Indirect_Access bit in the Miscellaneous Host Control register.

d. Enable the PCI State register to allow the device driver read/write access by setting the Enable_PCI_State_Register bit in the Miscellaneous Host Control register.

12. Set Byte_Swap_Non_Frame_Data and Byte_Swap_Data in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521).

13. Set Word_Swap_Data and Word_Swap_Non_Frame_Data (optional). When the host processor architecture is little-endian, set these additional bits in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521).

14. Configure the Port Mode bits of the Ethernet MAC Mode register (see "EMAC Mode Register (offset: 0x400)" on page 321) to GMII for all devices supporting copper Ethernet Media. Wait for 40 ms.

15. Poll for bootcode completion. The device driver should poll the general communication memory at 0xB50 for the one's complement of the T3_MAGIC_NUMBER (i.e. 0xB49A89AB). The bootcode should complete initialization within 1000 ms for flash devices and 10000 ms for SEEPROM devices.

16. Enable/disable any required bug fixes. Refer to the applicable errata document for information on any errata that should be worked around by enabling/disabling the control bits of chip bug fixes if any are applicable.

17. Enable Tagged Status Mode (optional) by setting the Enable_Tagged_Status_Mode bit of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). For additional information on Tagged Status mode see "Interrupt Processing" on page 219.

18. Clear the driver status block memory region. Write zeros to the host memory region where the status block will be DMAed (see "Status Block" on page 82).

19. Configure the DMA Write Water Mark in the DMA Read/Write Control register (see "DMA Read/Write Control Register (offset: 0x6C) — Function 0" on page 258). If the Max Payload Size of PCIe Device Control register is 128 bytes, set the DMA write water mark bits (bits19-21) of DMA Read/Write Control register to 011b (for a water mark of 128 bytes). Otherwise (max payload size is 256 bytes or more), set the DMA write water mark bits (bits 19–21) of DMA Read/Write Control register to 111b (for a watermark of 256 bytes).

Note: For the BCM5725/BCM5762/BCM57767-PG106, do not touch DMA read/write control reg (0x6C), and let the boot code have full control of it.

20. Set DMA byte swapping (optional). If the host processor architecture is big-endian, the MAC may byte swap both control and frame data when acting as a PCI DMA master. Set the Byte_Swap_Non-Frame_Data, Byte_Swap_Data and Word_Swap_Data bits in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521).

21. Configure the host-based send ring. Set the Host_Send_BDs bit in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521).

22. Indicate Driver is ready to receive traffic. Set the Host_Stack_Up bit in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521).

23. Configure TCP/UDP pseudo header checksum offloading. This step is relevant when TCP/UDP checksum calculations are offloaded to the device. The device driver may optionally disable receive and transmit pseudo header checksum calculations by the device by setting the Receive_No_PseudoHeader_Checksum and Send_No_PseudoHeader_Checksum bits in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 521). If the Send_No_PsuedoHeader_Checksum bit is set, the host software should make sure of seeding the correct pseudo header checksum value in TCP/UDP checksum field. Similarly, if the Receive_No_PsuedoHeader_Checksum bit is set, the device driver should calculate the pseudo header checksum and add it to the TCP/UDP checksum field of the received packet.

24. Configure MAC Mbuf memory pool watermarks ("DMA MBUF Low Watermark Register (offset: 0x4414)" on page 455, and "MBUF High Watermark Register (offset: 0x4418)" on page 455). Broadcom has run hardware simulations on the Mbuf usage and strongly recommends the settings shown in the table below. These settings/values establish proper operation for 10/100/1000 speeds.

Table 55: Recommended BCM57XX Ethernet Controller Memory Pool Watermark Settings

Register	*Standard Ethernet Frames*
MAC RX Mbuf Low Watermark (0x4414)	0x2A
Mbuf High Watermark (0x4418)	0xA0

Note: The Low WaterMark Max Receive Frames register (0x504) specifies the number of good frames to receive after RxMbuf Low Watermark has been reached. The driver software should make sure that the MAC RxMbuf Low WaterMark is greater than the number of Mbufs required for receiving the number of frames as specified in 0x504. The first Mbuf in the Mbuf chain of a frame has 80 bytes of packet data while each of the subsequent Mbufs [except the last Mbuf] has 120 bytes for packet data. The last Mbuf in the chain has the rest of the packet data, which can be up to 120 bytes.

25. Configure flow control behavior when the RX Mbuf low watermark level has been reached (see table below for "Low Watermark Maximum Receive Frame Register (offset: 0x504)" on page 337.

Table 56: Recommended BCM57XX Ethernet Controller Low Watermark Maximum Receive Frames Settings

Register	*Recommended Value*
Low Water Mark Maximum Receive Frames (0x504)	1

26. Enable the buffer manager. The buffer manager handles the internal allocation of memory resources for send and receive traffic. The Enable and Attn_Enable bits should be set in the Buffer Manager Mode register (see "Buffer Manager Mode Register (offset: 0x4400)" on page 453.

27. Set the BD ring replenish threshold for the RX producer ring. The threshold values indicate the number of buffer descriptors that must be indicated by the host software before a DMA is initiated to fetch additional receive descriptors used to replenish used receive descriptors. The recommended configuration value for the Standard Receive BD Ring Replenish Threshold (see "Standard Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C18)" on page 382) is 0x19.

28. Initialize the standard receive buffer ring. Host software should write the ring control block structure (see "Ring Control Blocks" on page 124) to the Standard Receive BD Ring RCB register (see "Standard Receive BD Ring RCB Registers" on page 379). Host software should be careful to initialize the host physical memory address based on allocation routines specific to the OS.

29. Initialize the Max_len/Flags Receive Ring RCB register (0x2458). Note that beginning with BCM5725/ BCM5762/BCM57767 and BCM5718 families of controllers, field 15:2 no specifies the maximum expected size of a receive frame (it was formerly "Reserved").

30. Initialize the Receive Producer Ring NIC Address register (offset: 0x245C) to a value of 0x6000 for the BCM5725/BCM5762/BCM57767.

31. Initialize Receive BD Standard Producer Ring Index (0x26C).

32. Initialize the Standard Ring Replenish Watermark register (offset: 0x2d00). The recommended value is 0x20. If supporting Jumbo frames, then also initialize 0x2d04 (recommended value is 0x10). See section "Jumbo Frames" on page 119 for more detail regarding supporting Jumbo frames.

33. Initialize the Send Ring Producer Index registers. Clear (i.e., set to zero) the send BD ring host producer index (see "Send BD Ring Host Producer Index (High Priority Mailbox) Register (offset: 0x300-0x307)" on page 320).

34. Initialize the send rings. The Send Ring Control Blocks (RCBs) are located in the miscellaneous memory region from 0x100 to 0x1FF. Host software should be careful to initialize the host physical memory address based on allocation routines specific to the OS. The MAC caches ¼ of the available send BDs in NIC local memory, so the host driver must set up the NIC local address. The following formula should be used to calculate the NIC Send Ring address: `NIC Ring Address = 0x4000 + (Ring_Number * sizeof(Send_Buffer_Descriptor) * NO_BDS_IN_RING) / 4`

35. Disable unused receive return rings. Host software should write the RCB_FLAG_RING_DISABLED bit to the flags field of the ring control blocks of all unused receive return rings.

36. Initialize receive return rings. The receive return ring RCBs are located in the miscellaneous memory region from 0x200 to 0x2FF. Host software should be careful to initialize the host physical memory address based on allocation routines specific to the OS. The Max_Len field indicates the ring size and it can be configured to either 32 or 64 or 128 or 256 or 512. The NIC RingAddress field of the RCB has a hardware power-on default value that is invalid for receive return rings, and the driver should set NIC Ring Address to 0x6000.

37. Initialize the Receive Producer Ring mailbox registers. The driver should write the value 0x00000000 (clear) to the low 32 bits of the receive BD standard producer ring index mailbox (see "Receive BD Standard Producer Ring Index (High Priority Mailbox) Register (offset: 0x268-0x26f)" on page 319).

Note: Host software must insure that on systems that support more than 4 GB of physical memory, send rings, receive return rings, producer rings, and packet buffers are not allocated across the 4 GB memory boundary. For example, if the starting memory address of the standard receive buffer ring is below 4 GB and the ending address is above 4 GB, a read DMA PCI host address overflow error may be generated (see "Read DMA Status Register (offset: 0x4804)" on page 471.

Note: The standard RX producer threshold value should be set very low. Some OSs may run short of memory resources and the number of BDs that are made available decrease proportionally.

Note: The maximum number of send BDs for a single packet is (0.75)*(ring size).

38. Configure the MAC unicast address. See "MAC Address Setup/Configuration" on page 144 for a full description of unicast MAC address initialization.

39. Configure random backoff seed for transmit. See the Ethernet Transmit Random Backoff register (see "Ethernet Transmit Random Backoff Register (offset: 0x438)" on page 327). Broadcom recommends using the following algorithm: `Seed = (MAC_ADDR[0] + MAC_ADDR[1] + MAC_ADDR[2] + MAC_ADDR[3] + MAC_ADDR[4] + MAC_ADDR[5]) & 0x3FF`

40. Configure the message transfer unit (MTU) size. The MTU sets the upper boundary on RX packet size; packets larger than the MTU are marked oversized and discarded by the RX MAC. The MTU bit field in the Receive MTU Size register (see "Receive MTU Size Register (offset: 0x43C)" on page 327) must be configured before RX traffic is accepted. Host software should account for the following variables when calculating the MTU:

- VLAN TAG
- CRC
- Jumbo frames enabled

41. Configure the Inter-Packet Gap (IPG) for transmit. The Transmit MAC Lengths register (see "Transmit MAC Lengths Register (offset: 0x464) " on page 332) contains three bit fields: IPG_CRS_Length, IPG_Length, and Slot_Time_Length. The value the 0x2620 should be written into this register.

Note: An incorrectly configured IPG introduces far-end receive errors on the MAC's link partner.

42. Configure default RX return ring for nonmatched packets. The MAC has a rules checker, and packets do not always have a positive match. For this situation, host software must specify a default ring where RX packet should be placed. The bit field is located in the Receive Rules Configuration register (see "Receive Rules Configuration Register (offset: 0x500)" on page 336).

43. Configure the number of receive Lists. The Receive List Placement Configuration register (see "Receive List Placement Configuration Register (offset: 0x2010)" on page 375) allows host software to initialize QOS rules checking. For example, a value of 0x181 (as used by Broadcom drivers) breaks down as follows:

- One interrupt distribution list
- Sixteen active lists
- One bad frames class

44. Write the Receive List Placement Statistics mask. Broadcom drivers write a value of 0x7BFFFF (24 bits) to the Receive List Placement Stats Enable Mask register (see "Receive List Placement Statistics Enable Mask Register (offset: 0x2018)" on page 376).

45. Enable RX statistics. Assert the Statistics_Enable bit in the Receive List Placement Control register (see "Receive List Placement Statistics Control Register (offset: 0x2014)" on page 375).

46. Enable the Send Data Initiator mask. Write 0xFFFFFF (24 bits) to the Send Data Initiator Enable Mask register (see "Send Data Initiator Statistics Mask Register (offset: 0xC0C)" on page 358).

47. Enable TX statistics. Assert the Statistics_Enable and Faster_Statistics_Update bits in the Send Data Initiator Control register (see "Send Data Initiator Statistics Control Register (offset: 0xC08)" on page 357.

48. Disable the host coalescing engine. Software needs to disable the host coalescing engine before configuring its parameters. Write 0x0000 to the Host Coalescing Mode register (see "Host Coalescing Mode Register (offset: 0x3C00)" on page 443).

49. Poll 20 ms for the host coalescing engine to stop. Read the Host Coalescing Mode register (see "Host Coalescing Mode Register (offset: 0x3C00)" on page 443) and poll for 0x0000. The engine was stopped in the previous step.

50. Configure the host coalescing tick count. The Receive Coalescing Ticks and Send Coalescing Ticks registers (see "Receive Coalescing Ticks Register (offset: 0x3C08)" on page 444 and "Send Coalescing Ticks Register (offset: 0x3C0C)" on page 445) specify the number of clock ticks elapsed before an interrupt is driven. The clock begins ticking after RX/TX activity. Broadcom recommends the settings shown in the table below.

Table 57: Recommended BCM57XX Ethernet Controller Host Coalescing Tick Counter Settings

Register	*Recommended Value*
Receive Coalescing Ticks (0x3C08)	0x48
Send Coalescing Ticks (0x3C0C)	0x14

51. Configure the host coalescing BD count. The Receive Max Coalesced BD and Send Max Coalesced BD registers (see "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446 and "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 446 specify the number of frames processed before an interrupt is driven. Broadcom recommends the settings shown in the table below.

Table 58: Recommended BCM57XX Ethernet Controller Host Coalescing Frame Counter Settings

Register	*Recommended Value*
Receive Max Coalesced Frames (0x3C10)	0x05
Send Max Coalesced Frames (0x3C14)	0x35

52. Configure the max-coalesced frames during interrupt counter. While host software processes interrupts, this value is used. See the Receive Max Coalesced Frames During Interrupt and Send Max Coalesced Frames During Interrupt registers (see "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446 and "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 446. Broadcom recommends the settings shown in the table below.

Table 59: Recommended BCM57XX Ethernet Controller Max Coalesced Frames During Interrupt Counter Settings

Register	*Recommended Value*
Receive Max Coalesced Frames During Interrupt (0x3C20)	0x05
Send Max Coalesced Frames During Interrupt (0x3C24)	0x05

53. Initialize host status block address. Host software must write a physical address to the Status Block Host Address register (see "Status Block Host Address Register (offset: 0x3C38)" on page 447, which is the location where the MAC must DMA status data. This register accepts a 64-bit value in register 0x3C38 (high order 32 bits) and 0x3C3C (low order 32 bits).
54. Enable the host coalescing engine (0x3C00 bit 1).
55. Enable the receive BD completion functional block. Set the Enable and Attn_Enable bits in the Receive BD Completion Mode register (see "Receive BD Completion Mode Register (offset: 0x3000)" on page 384).
56. Enable the receive list placement functional block. Set the Enable bit in the Receive List Placement Mode register (see "Receive List Placement Mode Register (offset: 0x2000)" on page 374).
57. Enable DMA engines. Set the Enable_FHDE, Enable_RDE, and Enable_TDE bits in the Ethernet Mac Mode register (see "EMAC Mode Register (offset: 0x400)" on page 321.
58. Enable and clear statistics. Set the Clear_TX_Statistics, Enable_TX_Statistics, Clear_RX_Statistics, and Enable_TX_Statistics bits in the Ethernet Mac Mode register (see "EMAC Mode Register (offset: 0x400)" on page 321).
59. Delay 40 microseconds.
60. Configure the General Miscellaneous Local Control register (see "Miscellaneous Local Control Register (offset: 0x6808)" on page 524). Set the Interrupt_On_Attention bit for MAC to assert an interrupt whenever any of the attention bits in the CPU event register are asserted.
61. Delay 100 microseconds.
62. Configure the Write DMA Mode register (see "Write DMA Mode Register (offset: 0x4C00)" on page 503). The following bits are to be asserted:
    - Enable-starts the functional block
    - Write_DMA_PCI_Target_Abort_Attention_Enable
    - Write_DMA_PCI_Master_Abort_Attention_Enable
    - Write_DMA_PCI_Parity_Attention_Enable
    - Write_DMA_PCI_Host_Address_Overflow_Attention_Enable
    - Write_DMA_PCI_FIFO_Overerrun_Attention_Enable
    - Write_DMA_PCI_FIFO_Underrun_Attention_Enable
    - Write_DMA_PCI_FIFO_Overwrite_Attention_Enable
    - Write_DMA_Local_Memory_Read_Longer_Than_DMA_Length
63. Set bit-29 of the Write DMA Mode register (see "Write DMA Mode Register (offset: 0x4C00)" on page 503) to enable the host coalescence block fix that configures the device to send out status block update before the interrupt message.

Note: For the BCM5725/BCM5762/BCM57767-PG106, there is no need to touch this bit; keep the default value for this bit.

64. Delay 40 microseconds.
65. Configure the Read DMA Mode register (see "Read DMA Programmable IPv6 Extension Header Register (offset: 0x4808)" on page 472). The following bits are asserted:
    - Enable-start functional block

- Read_DMA_PCI_Target_Abort
- Read_DMA_PCI_Master_Abort
- Read_DMA_PCI_Parity_Error
- Read_DMA_PCI_Host_Overflow_Error
- Read_DMA_PCI_FIFO_Overrun_Error
- Read_DMA_PCI_FIFO_Underrun_Error
- Read_DMA_PCI_FIFO_Overread_Error
- Read_DMA_Local_Memory_Write_Longer_Than_DMA_Length

66. Delay 40 microseconds.

67. Enable the receive data completion functional block. Set the Enable and Attn_Enable bits in the Receive Data Completion Mode register (see "Receive Data Completion Mode Register (offset: 0x2800)" on page 381).

68. Enable the send data completion functional block. Set the Enable bit in the Send Data Completion Mode register (see "Send Data Completion Mode Register (offset: 0x1000)" on page 365).

69. Enable the send BD completion functional block. Set the Enable and Attn_Enable bits in the Send BD Completion Mode register (see "Send BD Completion Mode Register (offset: 0x1C00)" on page 372).

70. Enable the Receive BD Initiator Functional Block. Set the Enable and Receive_BDs_Available_On_Receive_BD_Ring in the Receive BD Initiator Mode register (see "Receive BD Initiator Mode Register (offset: 0x2C00)" on page 381).

71. Enable the receive data and BD initiator functional block. Set the Enable and Illegal_Return_Ring_Size bits in the Receive Data and Receive BD Initiator Mode register (see "Receive Data and Receive BD Initiator Mode Register (offset: 0x2400)" on page 378).

72. Enable the send data initiator functional block. Set the Enable bit in the Send Data Initiator Mode register (see "Send Data Initiator Mode Register (offset: 0xC00)" on page 357).

73. Enable the send BD initiator functional block. Set the Enable and Attn_Enable bits in the Send BD Initiator Mode register (see "Send BD Initiator Mode Register (offset: 0x1800)" on page 370).

74. Enable the send BD selector functional block. Set the Enable and Attn_Enable bits in the Send BD Selector Mode register (see "Send BD Ring Selector Mode Register (offset: 0x1400)" on page 367).

75. Replenish the receive BD producer ring with the receive BDs.

76. Enable the transmit MAC. Set the Enable bit and the Enable_Bad_TxMBUF_Lockup_fix bit in the Transmit MAC Mode register (see "Transmit MAC Mode Register (offset: 0x45C)" on page 330). Optionally, software may set the Enable_Flow_Control to enable 802.3x flow control.

77. Delay 100 microseconds.

78. Enable the receive MAC. Set the Enable bit in the Receive MAC Mode register (see "Receive MAC Mode Register (offset: 0x468)" on page 332). Optionally, software may set the following bits:

- Enable_Flow_Control-enable 802.3x flow control
- Accept_oversized-ignore RX MTU up to 64K maximum size
- Promiscuous_Mode-accept all packets regardless of destination address
- No_CRC_Check-RX MAC does not check Ethernet CRC
- Various Hash enable bits if using RSS mode (receive-side scaling)

79. Delay 10 microseconds.

80. Set up the LED Control register (0x40C). The Broadcom driver uses a value of 0x800 when initializing this register.

81. Activate link and enable MAC functional blocks. Set the Link_Status bit in the MII Status register (see "MII Status Register (offset: 0x450)" on page 328) to generate a link attention.

82. Disable auto-polling on the management interface (optional) by writing 0xC0000 to the MI Mode register (see "MII Mode Register (offset: 0x454) " on page 329).

83. Set Low Watermark Maximum Receive Frame register (offset: 0x504) to a value of 1 for the BCM5725/ BCM5762/BCM57767 controller.

84. Configure D0 power state in PMSCR. See "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253. Optional-the PMCSR register is reset to 0x00 after chip reset. Software may optionally reconfigure this register if the device is being moved from D3 hot/cold.

85. Set up the physical layer and restart auto-negotiation. For details on PHY auto-negotiation, refer to the applicable PHY data sheet.

86. Set up multicast filters. Refer to "Packet Filtering" on page 144 for details on multicast filter setup.

Energy Efficient Ethernet™

The BCM5725/BCM5762/BCM57767 controller supports the IEEE specification for Energy Efficient Ethernet (EEE) (IEEE 802.3az-2010). The algorithm below describes how to initialize and enable EEE mode in the BCM5725/BCM5762/BCM57767 controller.

```
/*
 * Controller EEE initialization
 */

// Disable LPI requests
val = reg_read(0x36B0);
val &= 0xFFFFFF7F;
reg_write(0x36B0, val);

// Setup PHY DSP for EEE
mii_write(0x18, 0x0C00);
mii_write(0x17, 0x4022);
mii_write(0x15, 0x017B);
mii_write(0x18, 0x0400);

if (enable EEE advertisement)
{
    // Enable EEE advertisement for 100Base-TX and 1000Base-T modes
    mii_write(0x0D, 0x0007);
    mii_write(0x0E, 0x003C);
    mii_write(0x0D, 0x4007);
    mii_write(0x0E, 0x0006);
    val = mii_read(0x0E);

    //Enable MAC control of LPI
    reg_write(0x36BC,0x01000004);
    reg_write(0x36D0,0x000001F8);
    reg_write(0x36B0,0x00100348);

    // Set EEE timer debounce values
    reg_write(0x36B4,0x07ff07ff);
    reg_write(0x36B8,0x07ff07ff);
}
else
{
    // Disable EEE advertisement for 100Base-TX and 1000Base-T modes
    mii_write(0x0D, 0x0007);
    mii_write(0x0E, 0x003C);
    mii_write(0x0D, 0x4007);
    mii_write(0x0E, 0x0000);
    val = mii_read(0x0E);

    // Disable MAC control of LPI
    reg_write(0x36B0,reg_read(0x36B0) &= 0x00100000);
}


/*
 * Link status interrupt handler
 */

// Check for PHY link status
if ((mii_read(0x11) & 0x100) == 0x100)
{

    // Check for 1000mb link
    if ((mii_read(0x19) & 0x700) == 0x700)
    {
        // Set EEE LPI exit timing for 1000mb link speed
        reg_write(0x36d0, 0x19d);
    }
    // Check for 100mb link
    elseif ((mii_read(0x19) & 0x500) == 0x500)
    {
        // Set EEE LPI exit timing for 100mb link speed
        reg_write(0x36d0, 0x384);
    }

    //delay 1000 milliseconds
    ms_delay(1000);

    // Read PHY’s EEE negotiation status
    mii_write(0x0d, 7);
    mii_write(0x0e, 0x803e);
    mii_write(0x0d, 0x4007);
    val = mii_read(0x0e);

    // Enable EEE LPI request if EEE negotiated
    if (4 == val || 2 == val)
    {
       reg_write(0x0x36b0, reg_read(0x36b0) | 0x80);
    }

    break;
}
else
{
    // Disable LPI requests
    val = reg_read(0x36B0);
    val &= 0xFFFFFF7F;
    reg_write(0x36B0, val);

}
```

After the controller is fully initialized, the following algorithm may be used to verify EEE link status:

```
i=0
while(i < 100)
{
    if ((mii_read(0x11) & 0x100) == 0x100)
    {
        if ((mii_read(0x19) & 0x700) == 0x700)
        {
            if (((mii_read(0xA) & 0x7000) == 0x7000)  ||
               ((mii_read(0xA) & 0x3000) == 0x3000)))
              {
                  //link negotiated to gigabit master or slave
                  reg_write(0x36d0, 0x19d);
              }
        }
        elseif ((mii_read(0x19) & 0x500) == 0x500)
        {
            //link negotiated to 100Mbps
            reg_write(0x36d0, 0x384);
        }

        ms_delay(1000);//delay 1000 milliseconds

        //Assert LPI
        reg_write(0x36b0,(reg_read(0x36b0) | 0x80))
        break;
    }
    else
    {
        ms_delay(500);//delay 500 milliseconds
        i++;
    }
}

if (i >= 100)
{
    Link_not_detected();
}
```

Section 8: PCI

Configuration Space

Description

PCI, PCI-X, and PCIe devices must implement sixteen 32-bit PCI registers. These registers are required for a device to have PCI compliance. The format and layout of these registers is defined in the PCI 2.2 specification. Capability registers provide system BIOS and Operating Systems visibility into a set of optional features, which devices may implement. Although the capability registers are not required, the structure and mechanism for chaining auxiliary capabilities is defined in the PCI specification. Both software and BIOS must implement algorithms to fetch and program capabilities fields accordingly. Refer to section 6.7 of the PCI SIIG 2.2 specification. Additional PCI configuration space may be used for device-specific registers. However, device-specific registers are not exposed to system software, according to a specification/standard. System software cannot probe device specific registers without a predetermined understanding of the device and its functionality. In summary, three types of PCI configuration space registers may be exposed by any particular device:

- Required
- Optional capabilities
- Device specific

Note: The BCM5725/BCM5762/BCM57767 is PCIe v2.0 compliant.

Network devices implement large quantities of registers, and these registers could consume huge amounts of PCI configuration space. PCI configuration access is not very efficient, on a performance basis.

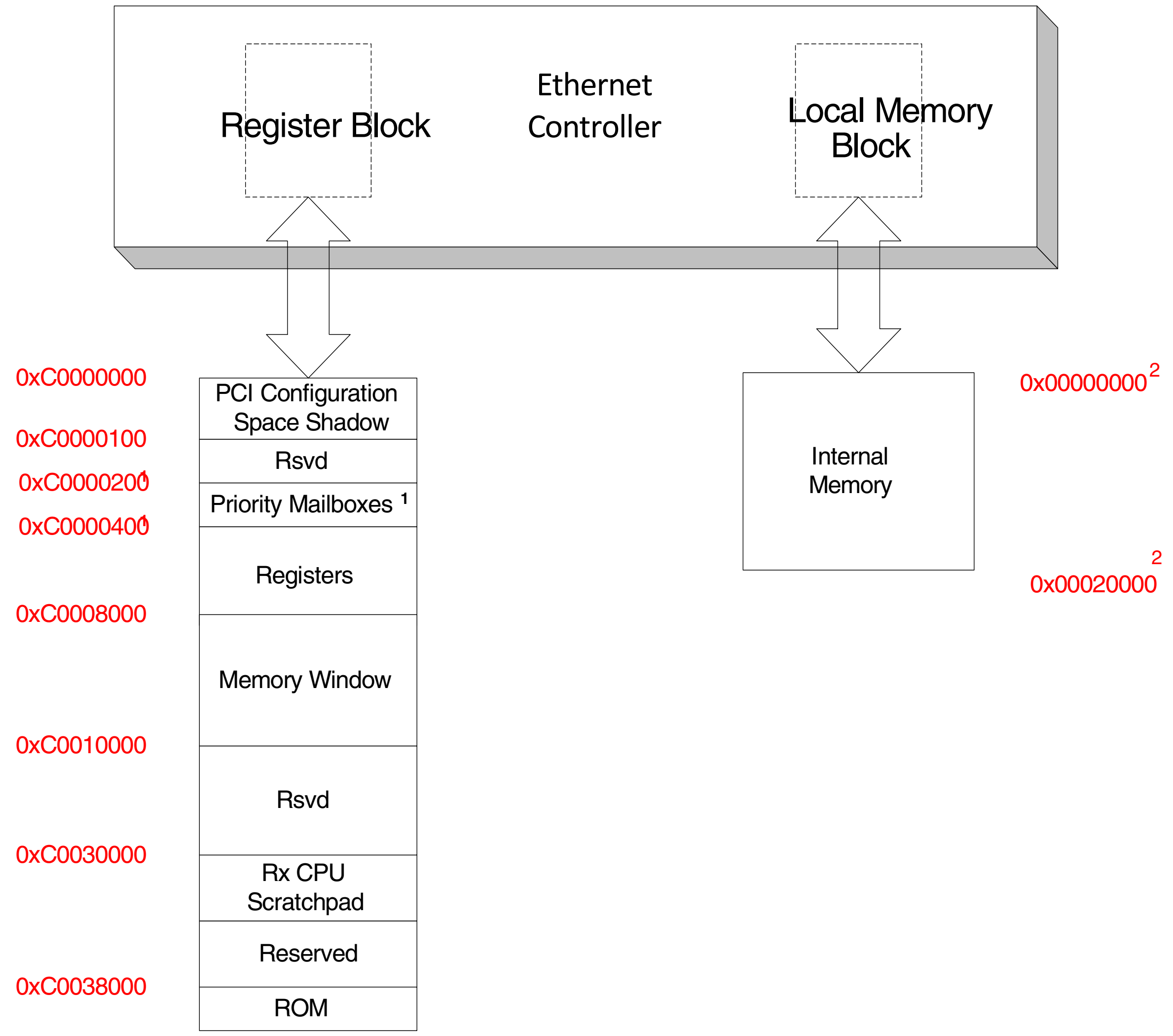
> ***Example:*** Intel x86 architectures use two I/O mapped I/O addresses 0xCF8 and 0xCFC for host-based access to PCI configuration space. Should a host device driver access these I/O addresses on every device read/write, CPU overhead would grow greatly. Generally, host device drivers should not use PCI configuration space for standard I/O and control programming. There is one special case—Universal Network Device Interface (UNDI) drivers. UNDI drivers may not have access to host memory mapped registers when operating in real-mode; thus, an indirect mode of access is necessary. The Ethernet controller implements a PCI indirect mode for memory, registers, and mailboxes access. A specific example of a device driver, which uses indirect mode, is the Preboot Execution (PXE) driver. PXE drivers may be stored in either option ROMs or directly in the system BIOS.

Most host device drivers use register blocks, which are mapped into host memory. Memory Mapped I/O is an efficient mechanism for PCI devices to use system resources. The type and extent of this memory mapping depends upon the MAC's configuration (see the operational characteristics subsection). A typical PCI device will decode a range of physical (bus) addresses, which do not conflict with physical memory or other PCI devices. Each device on the PCI bus will request a range of physical memory, and the PnP BIOS will assign mutually exclusive resources to that device. The size and range of resource is based upon each device's hardwired programming of the BAR. The Ethernet controller implements two modes of memory mapped I/O—Standard and Flat. I/O mapped I/O is not supported by the Ethernet controller, and there are no I/O space registers.

Note: The PCI BAR 0 register is only reset to 0 after a hard reset, otherwise it maintains its value over GRC and PCI resets.

Two programmable blocks expose Ethernet controller functionality to host software. The first is a register block. The second is a memory block. The register and memory blocks map into address spaces based on processor context. For example, the Ethernet controller has an on-chip RISC processor. This RISC processor will have an internal view of the register and memory blocks. This view is one large contiguous and addressable range, where the register block maps starting at offset 0xC0000000. Conversely, host processors have two entirely different views. When the Ethernet controller is configured in standard mode, the register block is mapped into a 64K host memory range. The host processor must use a memory window or indirect mode to access the memory block. It is fundamental to understand that the register and memory blocks are not necessarily tied together. The PCI mode and processor context all affect how software views both blocks (see Figure 33 on page 162).

Figure 33: Local Contexts



1. The high-priority mailboxes are at offset 0x200 through 0x3FF for host standard, and the low-priority mailboxes are at offset 0x5800 through 0x59FF for indirect mode.
2. The local memory addresses in this diagram apply to standard and internal views only. Refer to the section on memory maps and pool configuration.

The following components are involved in Ethernet controller configuration space mapping:

- Base Address registers
- Standard mode map mode
- Flat memory map mode
- Indirect access mode
- Configuration space header
- Host memory
- MAC registers

- MAC local memory

Functional Overview

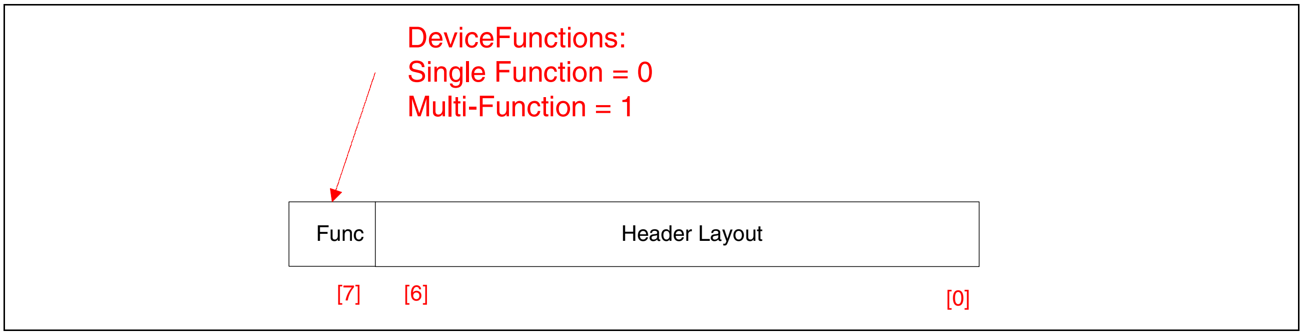
PCI Configuration Space Registers

The Ethernet controller configuration space can be broken into two regions: Header and Device Specific. Table 62 on page 178 shows the registers implemented to support PCI/PCI-X/PCIe functionality in the Ethernet controller. Reserved fields in PCI configuration registers will always return zero.

PCI Required Header Region

The bit-7 of the Header Type register (Offset 0x0E) in the PCI Required Header Region is used to identify whether the device is a single function device or multifunction device.

Figure 34: Header Type Register 0xE



Note: BIOS programmers should take special care to read bit_7 in PCI Header Type register (Offset 0x0E) before scanning the Ethernet controller PCI configuration space.

Single function PCI devices may decode access to non-implemented device functions in two ways, per Section 3.2.2.3.4 of the PCI 2.2 specification:

- A single function device may optionally respond to all function numbers as the same.
- May decode the function number field and respond only to function 0.

The Ethernet controller single function chips follow the stated technique #1— BIOS code scanning multifunctions get a target response from function(s) 1–7, but these functions are essentially shadows of function 0. Software that programs to function(s) 1–7 is remapped to function 0.

The header region is required by the PCI 2.2 specification. These registers must be implemented. The capabilities registers are optional; however, they must adhere to section 6.7 of the PCI SIIG 2.2 specification. Each capability has a unique ID, which is well-defined. The capabilities are chained using the Next Caps field, in the capability register. The last capability will have a Next Caps field, which is zeroed.

The Device-Specific registers are shown in Table 60.

Table 60: Device-Specific Registers

Register	*Cross Reference*
Miscellaneous Host Control	"Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256.
DMA Read/Write Control	"DMA Read/Write Control Register (offset: 0x6C) — Function 0" on page 258.
PCI State	"PCI State Register (offset: 0x70) — Function 0" on page 260.
Register Base Address	"Register Base Register (offset: 0x78) — Function 0" on page 261.
Memory Base Address	"Memory Base Register (offset: 0x7C) — Function 0" on page 261.
Register Data	"Register Data Register (offset: 0x80) — Function 0" on page 261.
Memory Window Data	"Memory Data Register (offset: 0x84) — Function 0" on page 261.
UNDI Receive BD Standard Producer Ring Producer Index Mailbox	"UNDI Receive BD Standard Producer Ring Producer Index Mailbox Register (offset: 0x98–0x9C)" on page 262.
UNDI Receive Return Ring Consumer Index Mailbox	"UNDI Receive Return Ring Consumer Index Register (offset: 0x88–0x8C)" on page 262.
UNDI Send BD Producer Index Mailbox	"UNDI Send BD Producer Index Mailbox Register (offset: 0x90–0x94)" on page 262.

Indirect Mode

Host software may use indirect mode to access the Ethernet controller resources, without using Memory Mapped I/O. Indirect mode shadows MAC resources to PCI configuration space registers. These shadow registers can be read/written by system software through PCI configuration space registers. The Ethernet controller indirect Mode registers expose the following MAC resources:

- Registers
- Local Memory
- Mailboxes

Indirect mode access can be used in conjunction with Standard Mode PCI access. Indirect mode has no interdependency on other PCI access modes and is a mode in itself.

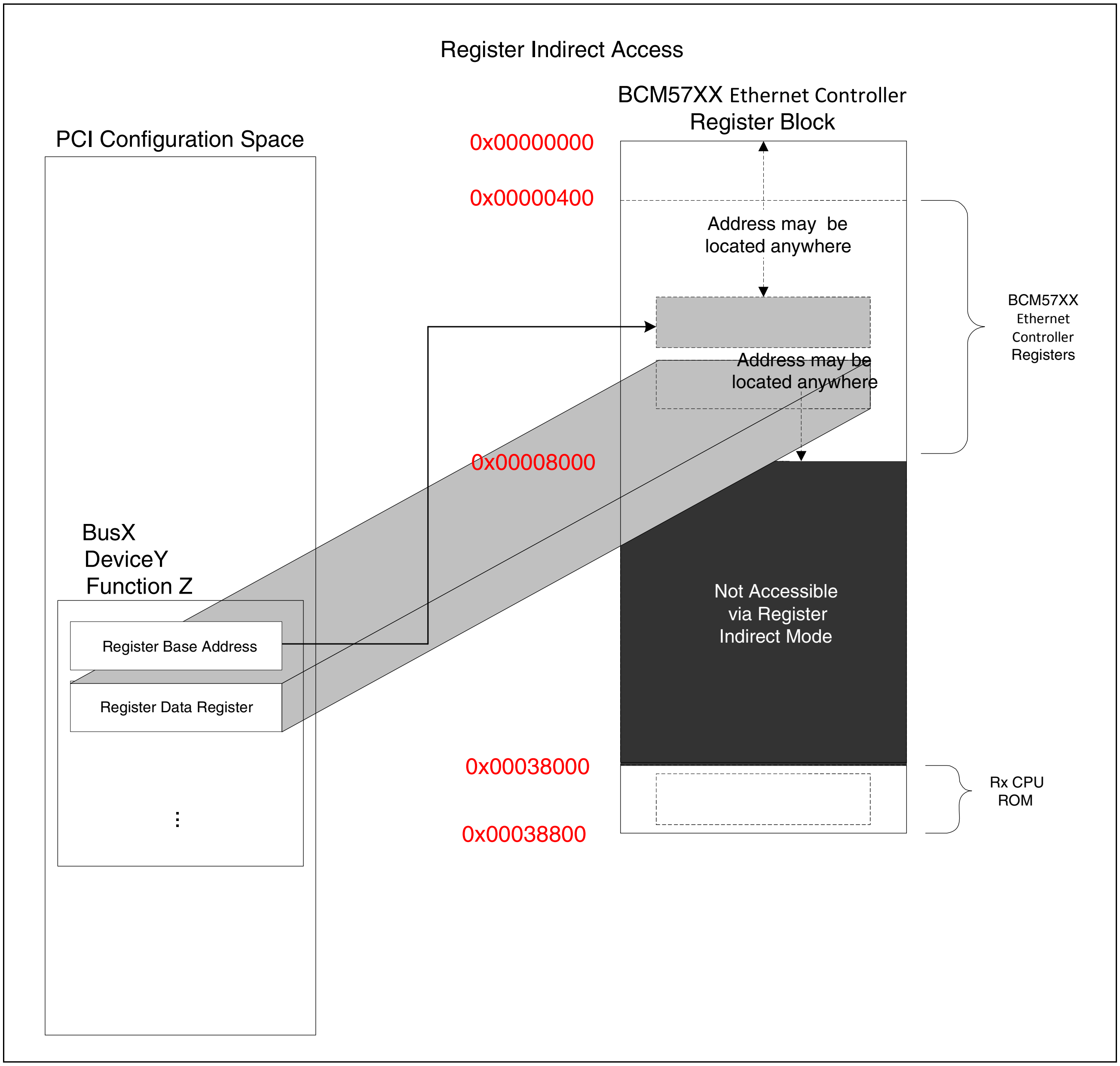
Note: Host software must assert the Indirect_Mode_Access bit in the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256) to enable indirect mode.

Indirect Register Access

Two PCI configuration space register pairs give host software access to the Ethernet controller register block. The Register_Base_Address register creates a position in the MAC register block. Valid positions range from 0x0000–0x8000 and 0x30000–0x38800 ranges. Access to the register block from 0x8000–0x30000, should be avoided and is not necessary. The Flat and Standard Modes do map a memory window into the 0x8000–0xFFFF ranges; however, the Memory Indirection register pair provides a more efficient mechanism to access the Ethernet controller memory block. The Register_Data register allows host software to read/write, from the indirection position. The Register_Base_Address register can be perceived as creating a cursor/pointer into the register block. The Register_Data register allows host software to read/write to the location, specified by the Register_Base_Address. This register pair accesses the Ethernet controller register block (see Figure 35 on page 166).

Note: If indirect register access is performed using memory write cycles (i.e., by accessing the Register_Base_Address and Register_Data registers through memory mapped by the PCI BAR register), as opposed to PCI configuration write cycles, the host software must insert a read command to the Register_Base_Address register between two consecutive writes to the Register_Base_Address and Register_Data registers.

Figure 35: Register Indirect Access

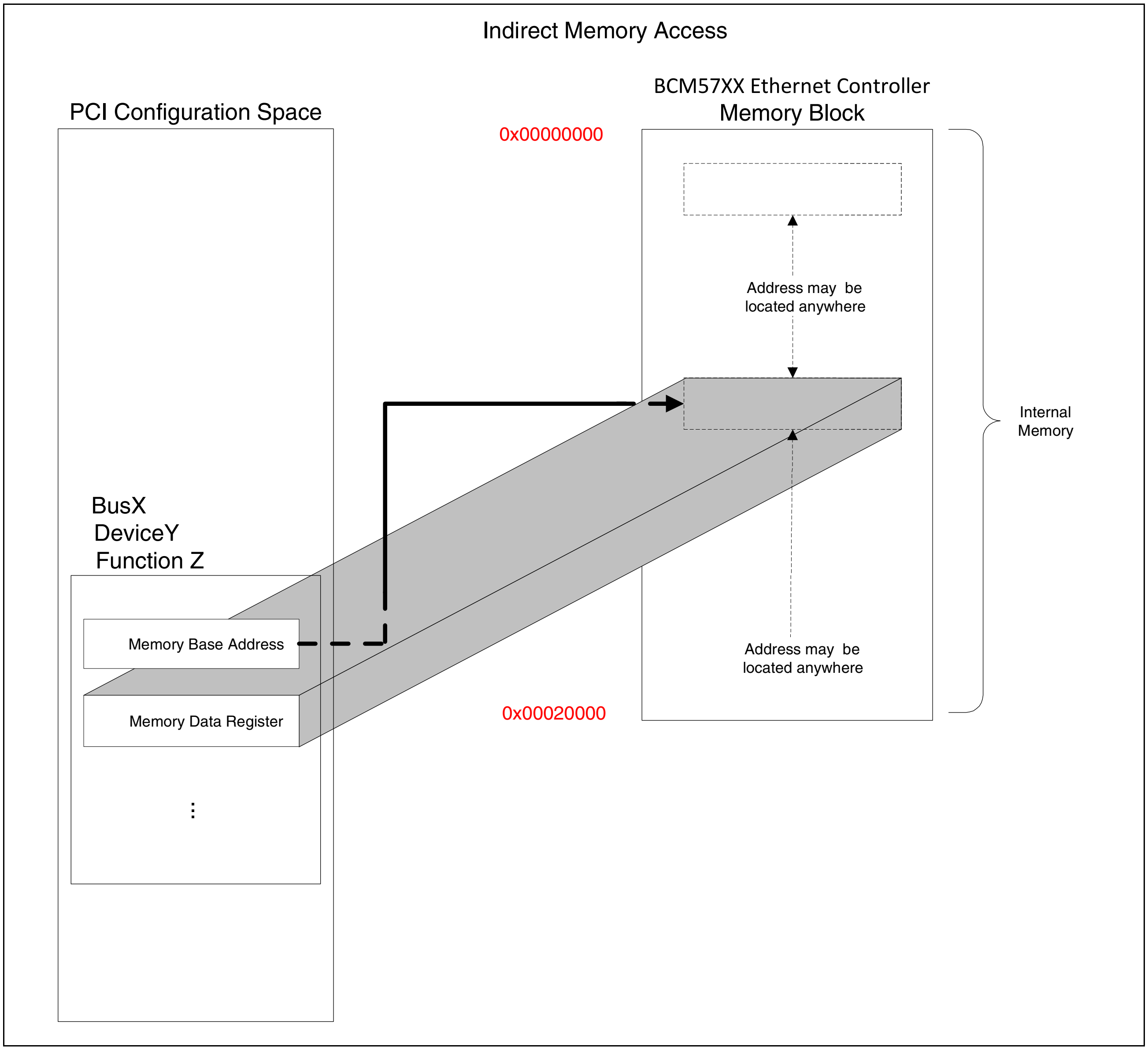


Indirect Memory Access

Memory indirect mode operates in the same fashion to register indirect mode. There is a PCI configuration space register pair, which is used to access the Ethernet controller memory block. The Memory_Window_Base_Address register positions a pointer/cursor in the local memory block. Unlike the Register_Base_Address register, the Memory_Window_Base_Address register may position at any valid offset. Access to ranges 0x00000–0x1FFFF is allowable. The Memory_Window_Data register is the read/write porthole for host software, using the previously positioned pointer/cursor. This register pair accesses the Ethernet controller local memory block (see Figure 36 on page 168).

Note: If Indirect Memory Access is performed using memory write cycles (i.e., by accessing the Memory_Window_Base_Address and Memory_Window_Data registers through memory mapped by the PCI BAR register), as opposed to PCI configuration write cycles, the host software must insert a read command to the Memory_Window_Base_Address register between two consecutive writes to the Memory_Window_Base_Address and Memory_Window_Data registers.

Figure 36: Indirect Memory Access



UNDI Mailbox Access

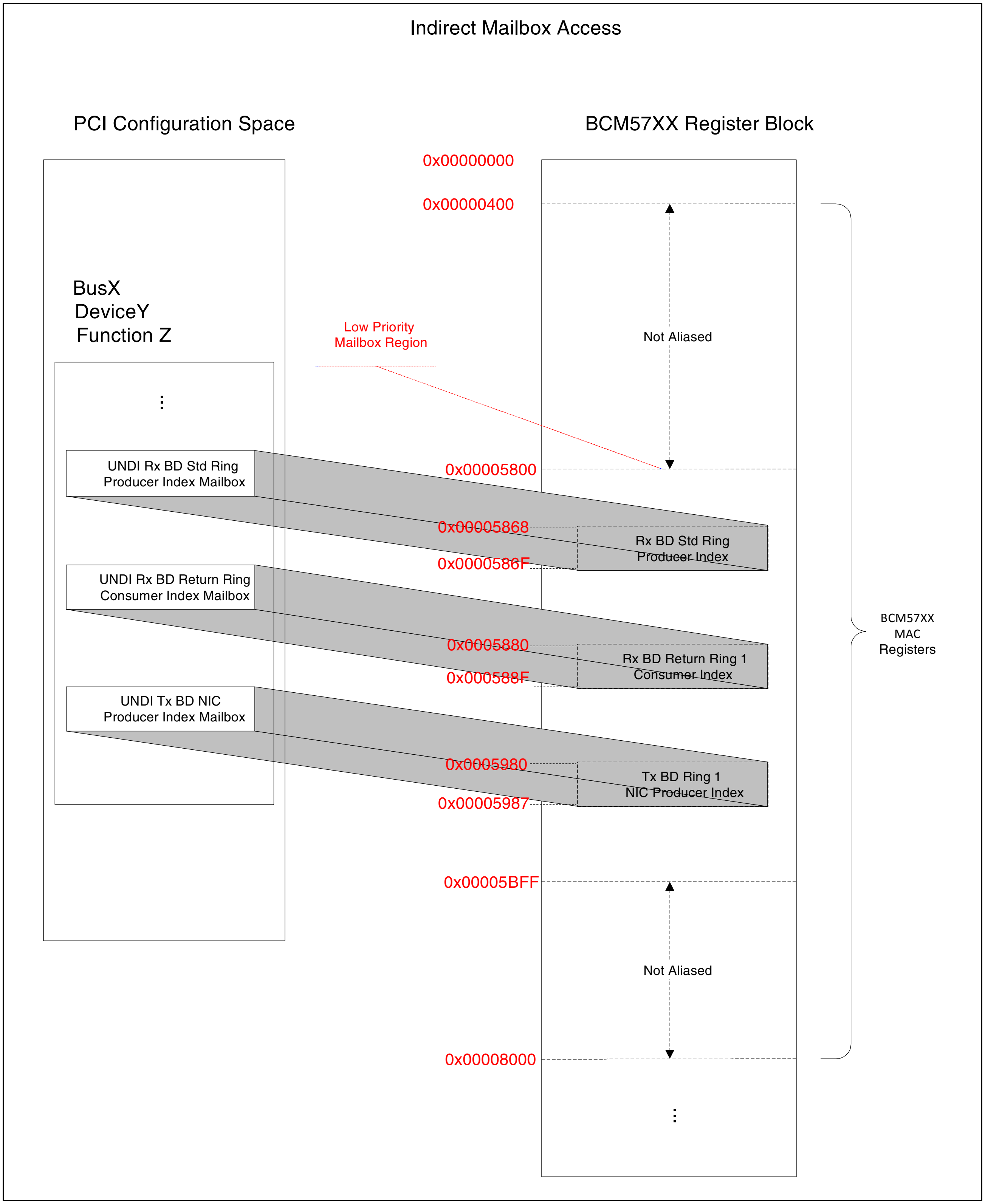
The UNDI mailboxes are shadows of Ethernet controller mailbox registers. All mailboxes reside in the Ethernet controller register block, not memory block. Unlike register and memory indirect access, the UNDI Mailboxes shadows are mapped 1:1 to a Ethernet controller register; these shadow registers do not have an address register.

- The UNDI_RX_BD_Standard_Ring_Producer_Index_Mailbox register shadows a mailbox located at offset 0x5868 (see "Receive BD Return Ring 3 Consumer Index (Low Priority Mailbox) Register (offset: 0x5898-0x589F)" on page 514), in the Ethernet controller register block. Any index update (write) to the UNDI_RX_BD_Standard_Ring_Producer_Index_Mailbox will advance the standard producer ring index; software signals hardware that an RX buffer descriptor is available.
- The UNDI_RX_BD_Return_Ring_Consumer_Index_Mailbox register corresponds to a mailbox located at offset 0x5880 (see "Receive BD Standard Producer Ring Index Register (offset: 0x5868)" on page 514). A update (write) to this register indicates that host software has consumed a RX buffer descriptor(s); return rings contain filled Enet frames, from the receive MAC.
- Finally, the UNDI_TX_BD_Host_Producer_Mailbox register maps to register offset 0x5900 () in the Ethernet controller register block. Host software writes to this register when Ethernet frame(s) are ready to be transmitted. Host software writes the index of buffer descriptor, which is ready for transmission.

Notice that all these UNDI shadows are the first or primary ring and not all the rings are shadowed into PCI configuration space. For example, Receive Return rings 2–16 do not have shadow registers. UNDI drivers only require a minimal set of registers to provide basic network connectivity. Functionality is the most important consideration. Fifteen additional receive return rings would extend the size of the Device Specific portion of the PCI Configuration Space registers.

The UNDI shadow registers alias three registers in the Ethernet controller register block (see Figure 37 on page 170).

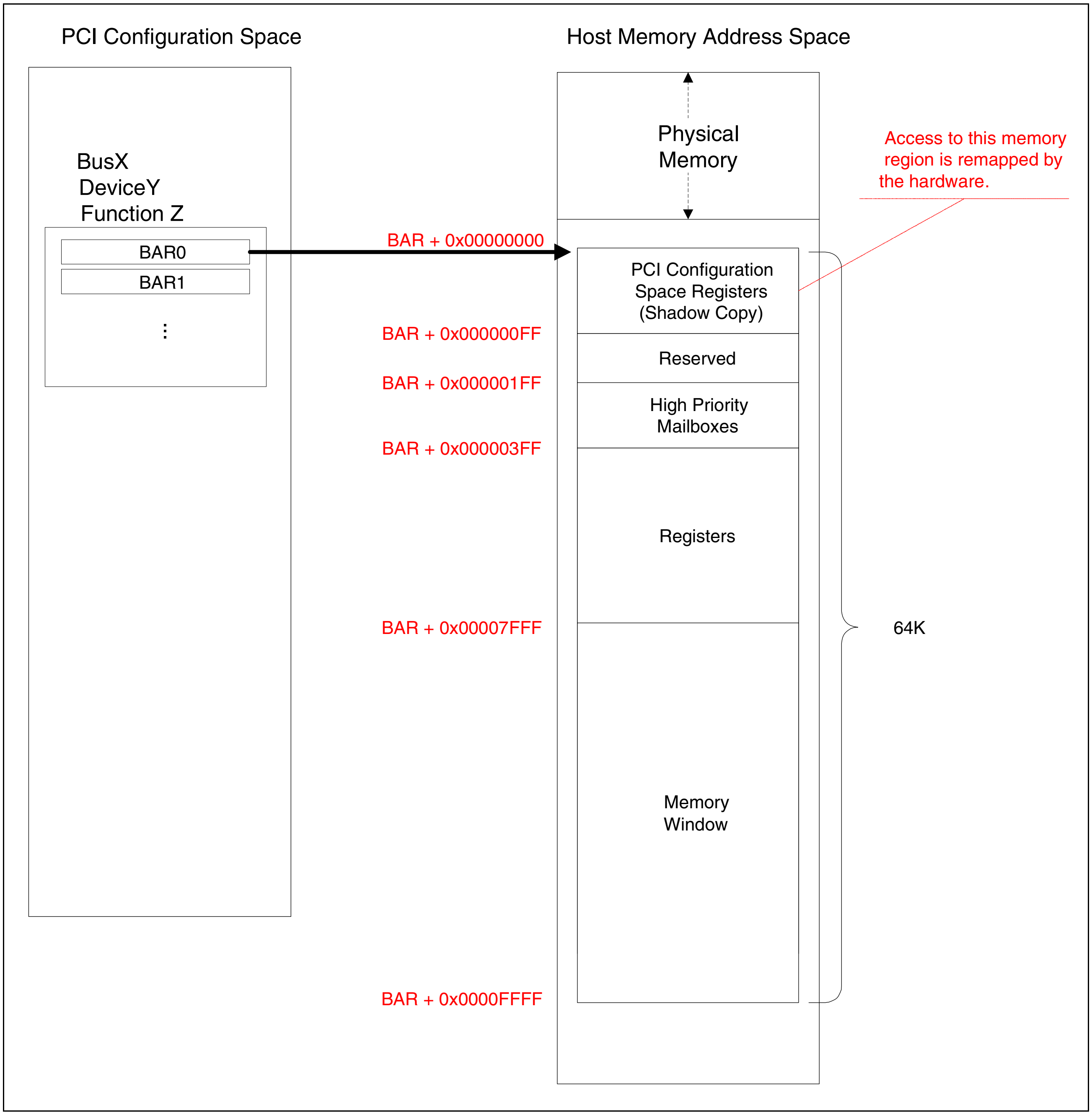
Figure 37: Low-Priority Mailbox Access for Indirect Mode


Indirect Mailbox Access
PCI Configuration Space
BCM57XX Register Block
BusX
DeviceY
Function Z
⋮
UNDI Rx BD Std Ring Producer Index Mailbox
UNDI Rx BD Return Ring Consumer Index Mailbox
UNDI Tx BD NIC Producer Index Mailbox
Low Priority Mailbox Region
0x00000000
0x00000400
Not Aliased
0x00005800
0x00005868
Rx BD Std Ring Producer Index
0x0000586F
0x0005880
Rx BD Return Ring 1 Consumer Index
0x000588F
0x0005980
Tx BD Ring 1 NIC Producer Index
0x00005987
0x00005BFF
Not Aliased
0x00008000
⋮
BCM57XX MAC Registers

Standard Mode

Standard mode is the most useful memory mapped I/O view provided by the Ethernet controller (see Figure 38). 64K of host memory space must be made available. The PnP BIOS or OS will program BAR0 and BAR1 with a base address where the 64K address region may be decoded. The BAR registers point to the beginning of the host memory mapped regions where Ethernet controller can be accessed.

Figure 38: Standard Memory Mapped I/O Mode



The Ethernet controller resources listed in the following are decoded in the 64K address block.

Table 61: PCI Address Map Standard View

Offset	*Name*	*Size*
0x00000000–0x000000ff	PCI Configuration space	256 bytes
0x00000200–0x000003ff	High-Priority Mailboxes	512 bytes
0x00000400–0x00007fff	Ethernet controller registers	31 KB
0x00008000–0x0000ffff	Memory Window	32 KB

32K is partitioned for MAC control registers and 32K available for a memory access window. Range 0x0000–0x00FF is a complete shadow of the PCI configuration space registers—host software can also read/write to the Ethernet controller's PCI configuration space registers via the host memory map. Host software may use the shadow registers to change PCI register contents and avoid PCI configuration cycles (transactions). Again, using the host memory map is slightly more efficient. The MAC's control/status registers are mapped from 0x0400–0x8000. See Section 12: "Ethernet Controller Register Definitions," on page 243 for complete register and bit definitions. Finally, the memory window range is 0x8000–0xFFFF. This 32K window is set in the PCI Configuration space using the Memory_Window_Base_Address register (see Figure 39). Bits 23:15 set the window aperture and bits 14:2 are effectively ignored/masked off. Bits 14–2 are relevant when host software uses memory indirection and the Memory_Window_Data register.

Figure 39: Memory Window Base Address Register

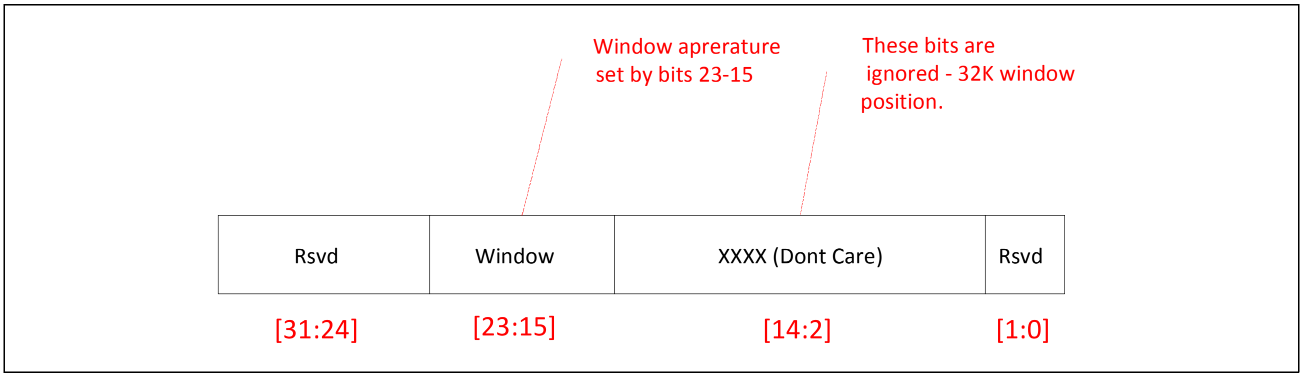


Figure 40 on page 173 shows how the 32K window can float in the Ethernet controller's local memory. The window aligns on 32K boundaries.

> ***Example:*** The memory window may start on the following addresses: 0x8000, 0x10000, and 0x18000. The window aperture may be positioned in the internal memory range 0x00000000 to 0x0001FFFF. When host software reads/writes to PCI_BAR + 32K + OFFSET in the host memory space, the Ethernet controller translates this read/write access to Memory_Window_Base_Address + OFFSET. Host software must not read/write from any address greater than PCI_BAR + 64K, since this memory space is not decoded by the Ethernet controller. Such an access may be decoded by another device, or simply go unclaimed on the PCI bus. Figure 40 on page 173 shows the relationship between the Memory_Window_Base_Address register and the Memory Window.

Figure 40: Standard Mode Memory Window

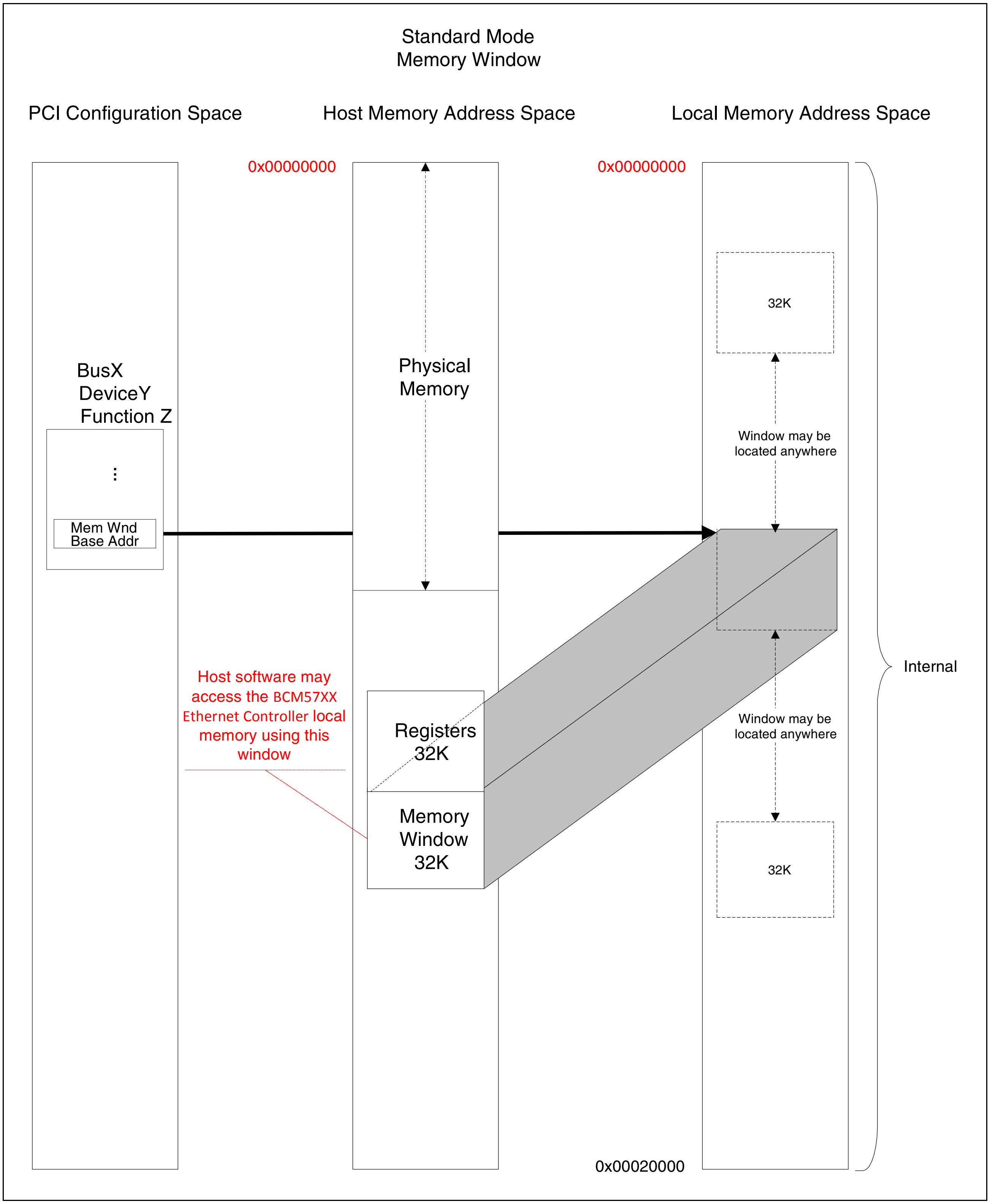


Figure 41: PCI Configuration Space/Host Memory Address Space

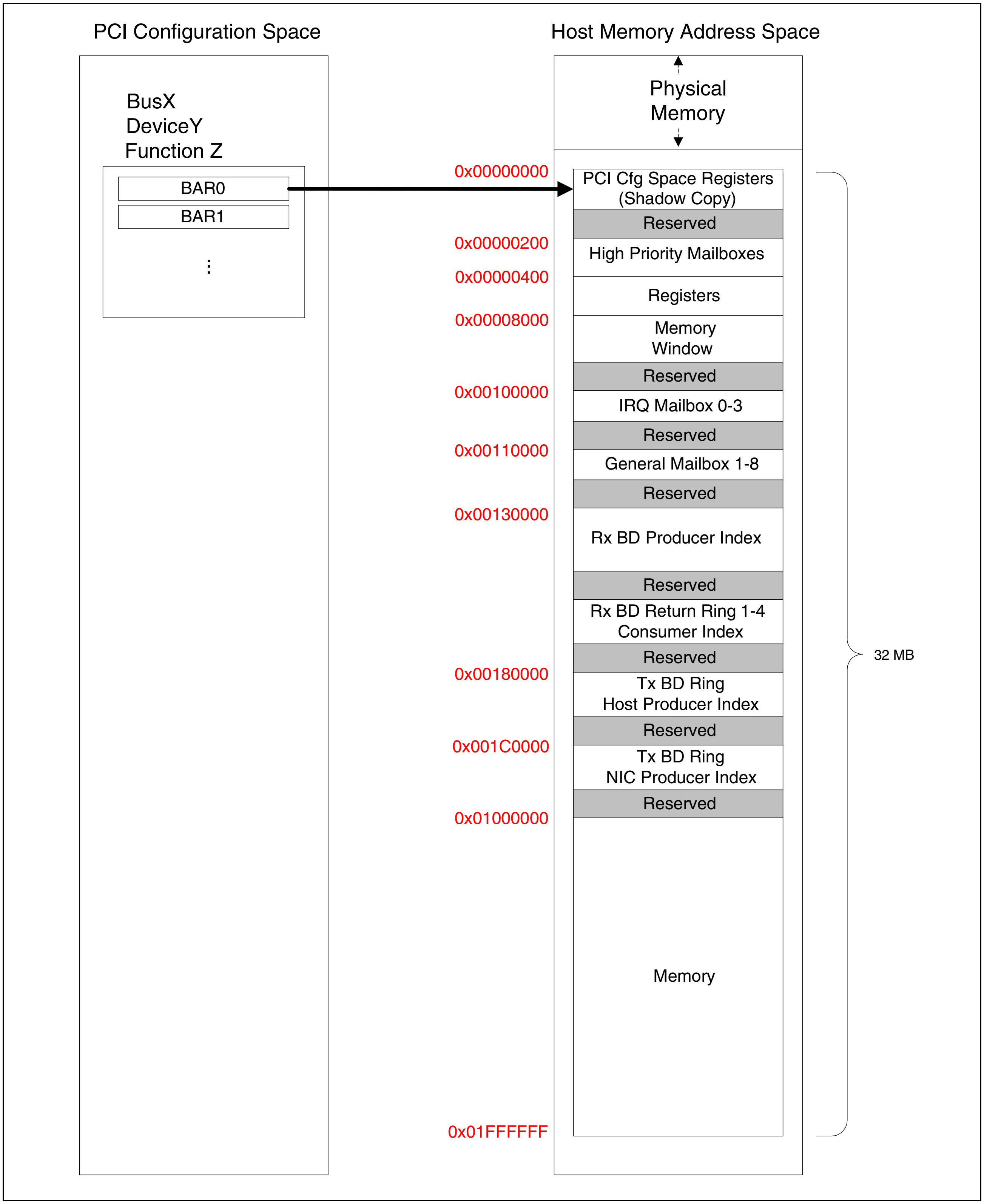
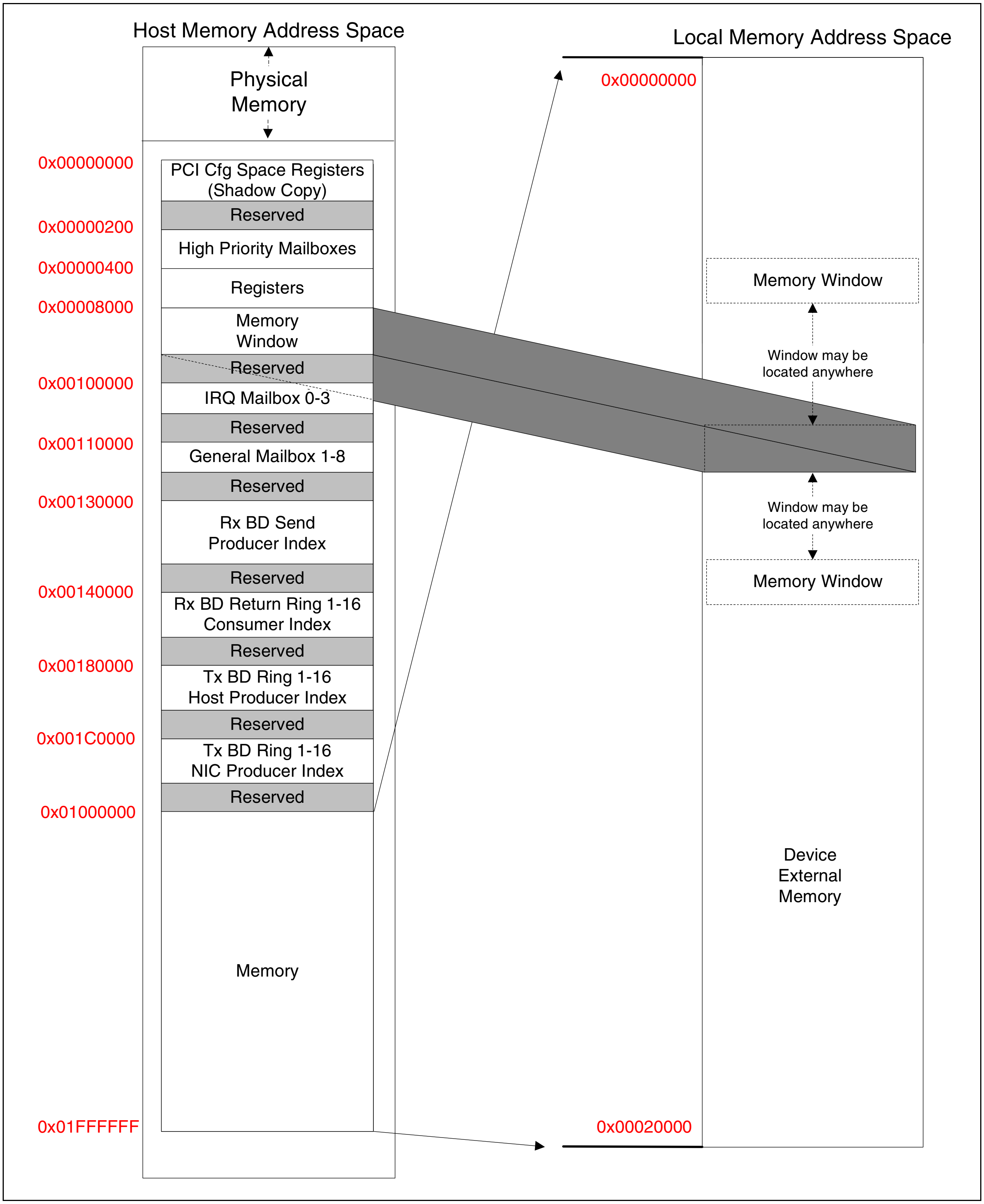


Figure 42: Techniques for Accessing Ethernet Controller Local Memory



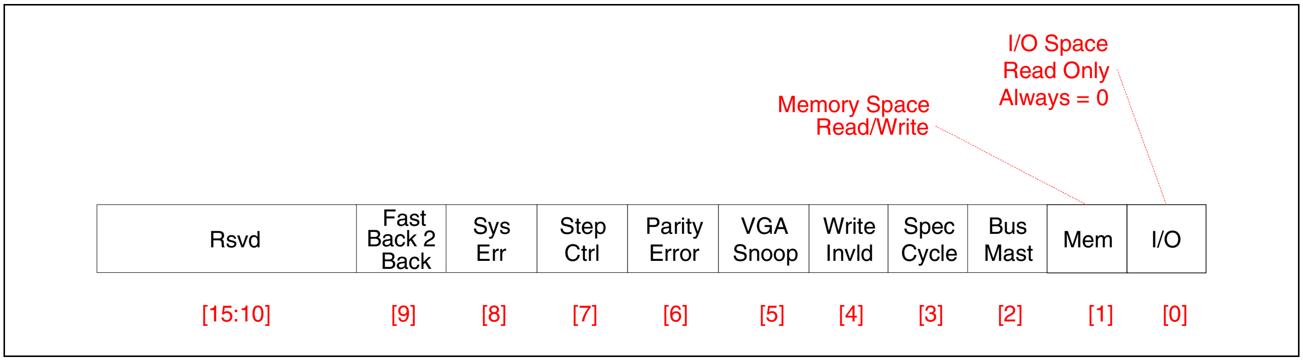
Memory Mapped I/O Registers

The following Ethernet controller registers are used in the mode configuration of the PCI memory-mapped I/O.

PCI Command Register

The PCI_Command register is 16-bits wide (see Figure 43). The Ethernet controller does not have I/O mapped I/O. The I/O_Space bit is de-asserted by hardware. The Ethernet controller does support Memory_Mapped_Memory and hardware will assert the Memory_Space bit. Both these bits are read-only and are usually read by the PnP BIOS/OS. The BIOS/OS examines these bits to assign non-conflicting resources to PCI devices.

Figure 43: PCI Command Register



PCI State Register

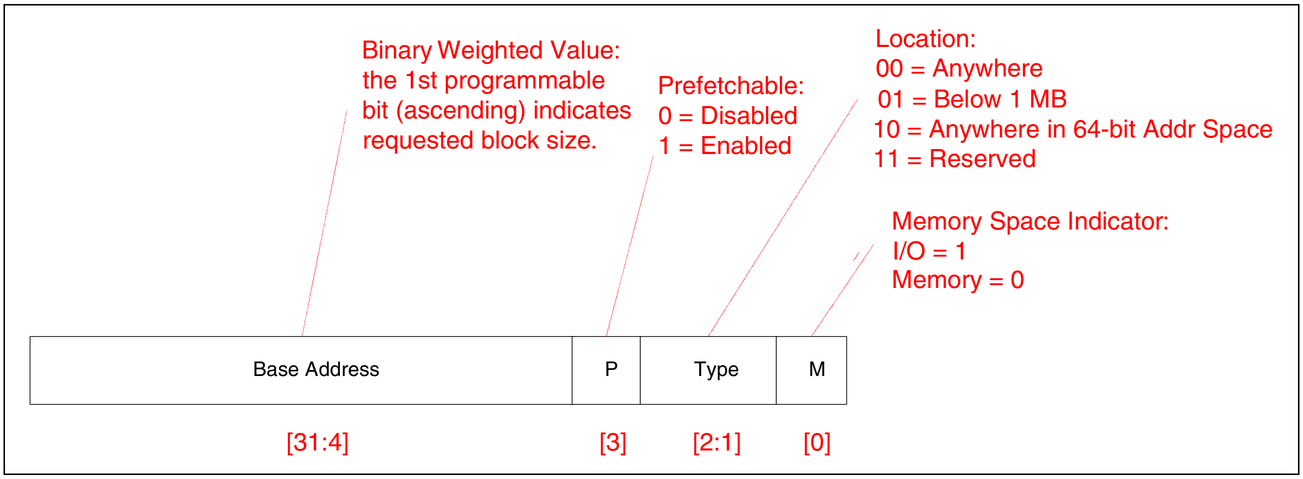
The PCI_State register is 32-bits wide. Operating mode is set with the Flat_View bit in the PCI_State register. When the Flat_View bit is asserted, the Ethernet controller decodes a 32M of block host memory. When the Flat_View bit is de-asserted, the Ethernet controller decodes a 64K block of host memory.

PCI Base Address Register

The PCI_Base_Address Register (BAR) specifies the location of a Ethernet controller memory mapped I/O block. The Ethernet controller mode configuration (Flat vs. Standard) affects how the BAR is setup (see Figure 44).

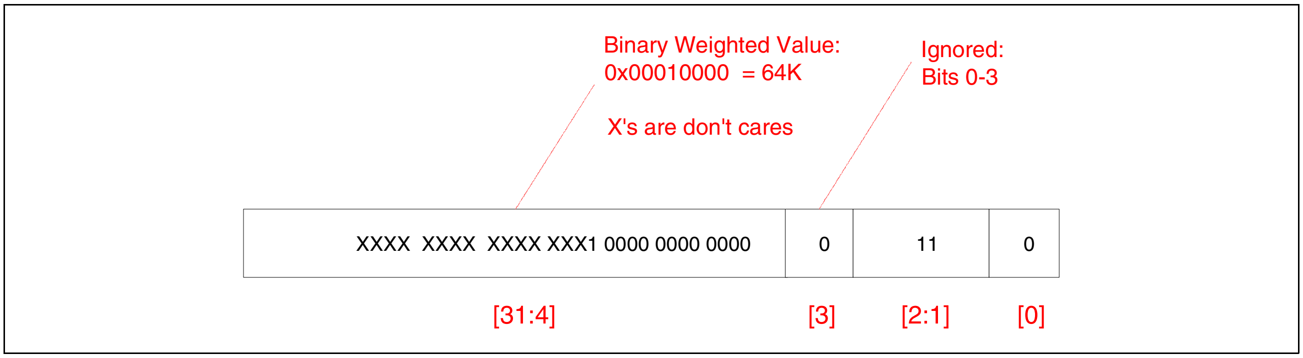
- Bits 4–31 in the PCI_Base_Address register are selectively programmable based on the amount of host memory requested. The PnP BIOS/OS will use an algorithm to test the BAR bits and determine the amount of physical memory requested.
- The Memory_Space_Indicator bit designates whether the BAR is memory or I/O mapped. The Ethernet controller hard codes the Memory_Space_Indicator bit to zero (de-asserted).
- The Location/Type bits specify locations in host memory space where a device can decode physical addresses. The Ethernet controller memory mapped I/O range may be placed anywhere in 64-bit address space (Type = 10).
- The Ethernet controller deasserts the Prefetchable bit to indicate that the memory range should not be cached.

Figure 44: PCI Base Address Register



The Ethernet controller 64K memory mapped I/O block is determined by the first programmable bit in the BAR. When the MAC is configured in standard mode, the mask 0xFFFF0000 identifies the BAR bits, which are programmable. Bit 16 is the first bit encountered in the scan upward, which is programmable; bits 0–3 are ignored. Host software will read zero values from bits 4–16. Figure 45 shows the BAR register and the bits returned to the OS/BIOS during resource allocation.

Figure 45: PCI Base Address Register Bits Read in Standard Mode



Register Quick Cross Reference

Device Family

The Ethernet controller PCI registers are listed in Table 62.

Table 62: PCI Registers

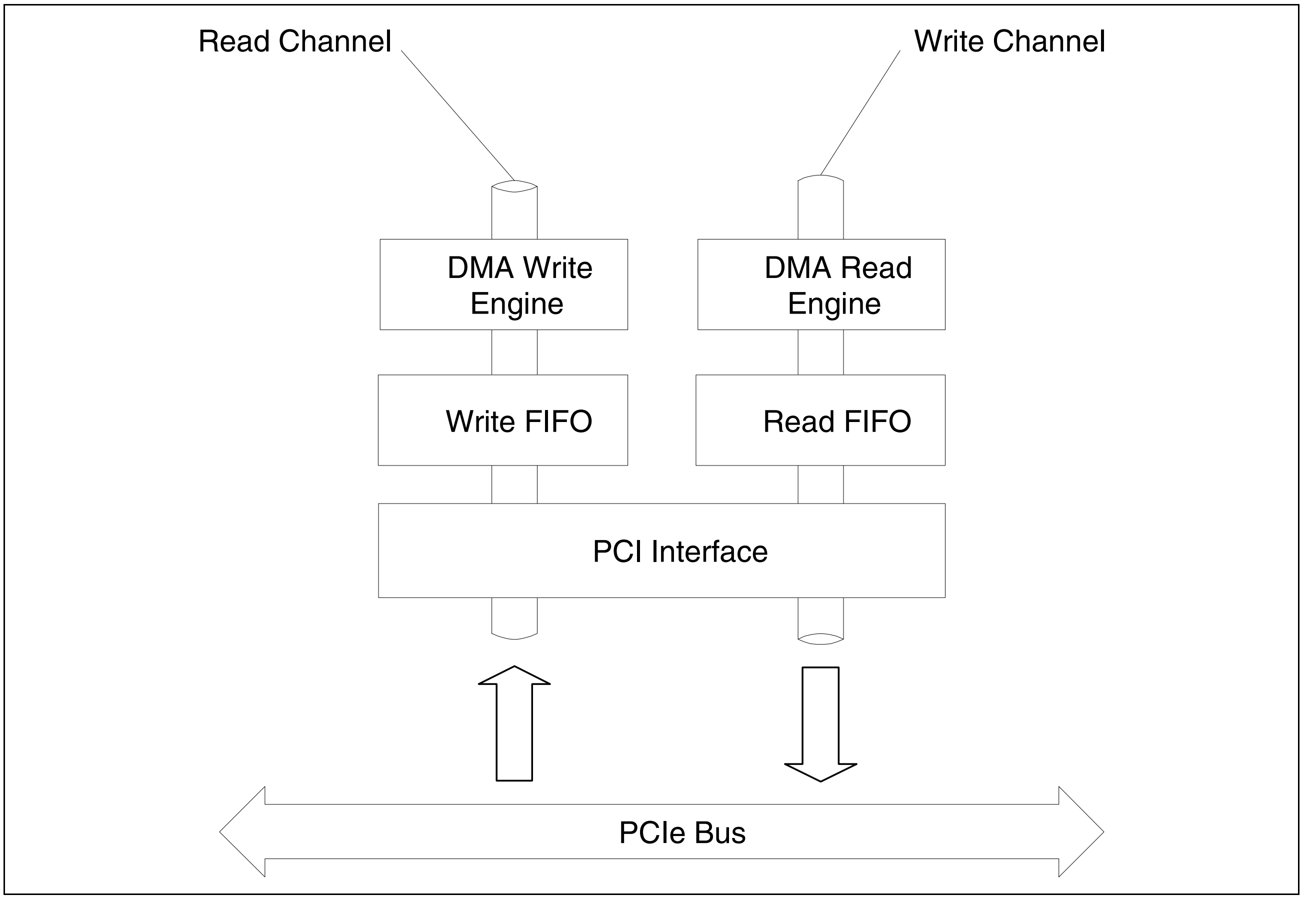
Register	*Bit*	*Cross Reference*
PCI Command	Memory_Space	"Status and Command Register (Offset: 0x04) — Function 0" on page 245.
PCI Command	IO_Space	"Status and Command Register (Offset: 0x04) — Function 0" on page 245.
PCI State	Flat_View	"PCI State Register (offset: 0x70) — Function 0" on page 260.
PCI Base Address 1	All	"Base Address Register 1 (offset: 0x10) — Function 0" on page 249.
PCI Base Address 2	All	"Base Address Register 2 (offset: 0x14) — Function 0" on page 250.

Bus Interface

Description

The read/write DMA engines both drive the PCIe interface. Typically, each DMA engine alternates bursts to the PCIe bus, and both interfaces may have outstanding transactions on the PCI bus. The BCM5725/BCM5762/BCM57767 architecture identifies two channels—a read DMA channel and a write DMA channel. Each channel corresponds to the appropriate DMA engine (see Figure 46). The configuration of the DMA engines and the PCI interface is discussed in this section.

Figure 46: Read and Write Channels of DMA Engine



The following architectural components are involved in the configuration of the PCI/DMA interface:

- DMA read engine
- DMA write engine
- DMA read FIFO
- DMA write FIFO
- PCIe interface
- PCI state register
- DMA read/write register

Operational Characteristics

Read/Write DMA Engines

Software must enable the bus master DMA bit for the Ethernet controller. The Ethernet controller is a bus-mastering device and the PCI interface requires that the Bus_Master enable bit be set by either the BIOS or host device driver. The bus master is the PCI transaction initiator. A PCI target will claim the transaction driven by the bus master. The Bus_Master enable bit is located in the PCI configuration space Command register (see "Status and Command Register (Offset: 0x04) — Function 0" on page 245) and this bit is read/write. The bit defaults to cleared/disabled after device reset.

The read and write DMA channels use FIFOs to buffer small amounts of PCI bus data. The FIFOs provide elasticity for data movement between internal memory and the PCI interface. Host software may configure DMA watermarks—values where PCI activity is enabled/disabled.

When enqueued data is less than the watermark value, PCI bus transactions are inhibited. The DMA channel will wait until the FIFO fills above the threshold before initiating PCI transactions. Host software may configure the DMA_Write_Watermark bit fields to set the activity threshold in the write FIFO. The DMA_Write_Watermark bit field is read/write and is also located in the DMA Read/Write register. The write watermark registers default to zero after power-on reset.

Expansion ROM

Description

The expansion ROM on the Ethernet controller is intended for implementation of PXE (Preboot Execution Environment). The devices support expansion ROM of up to 16 MB.

Operational Characteristics

By default, the Expansion ROM is disabled and the firmware has to explicitly enable this feature by setting PCI_State.PCI_Expansion_ROM_Desired bit to one (see "PCI State Register (offset: 0x70) — Function 0" on page 260). Once this bit is enabled, the boot code firmware handles the Expansion ROM accesses of the device.

BIOS

The BIOS detects if a PCI device supports Expansion ROM or not by writing the value 0xFFFFFFFE to Expansion_ROM_Base_Register (register 0x30 of PCI configuration). The BIOS then reads back from this register. If the value is nonzero, then this PCI device supports Expansion ROM; otherwise, it does not. The Ethernet controller returns a non-zero value appropriate for the expansion ROM size selected in NVRAM (see Section 4: "Common Data Structures," on page 67) when Expansion ROM is enabled (PCI_State.PCI_Expansion_ROM_Desired bit is set to 1). On the other hand, if the PCI_Expansion_ROM_Desired bit cleared, then the Ethernet controller returns a value of 0x00000000. This indicates to the BIOS that no Expansion ROM is supported.

If a PCI device supports Expansion ROM, the BIOS will assign a Expansion Base address to the device. It then checks for a valid ROM header (0x55 0xAA as first two bytes, and so forth) and checksum. If the ROM header and image are valid, the BIOS will copy the Expansion ROM image to HOST's Upper Memory Block (UMB) and invoke the initializing entry point.

Preboot Execution Environment

Preboot Execution Environment (PXE) is implemented as an Expansion ROM in the NIC implementation. In the LOM implementation, PXE normally resides in the system BIOS. In the NIC implementation, PXE image is stored in the NVRAM. Upon power on reset of the Ethernet controller, the RX RISC will load the boot code from the NVRAM into RX RISC scratch pad and execute. This boot code will program the device with programmable manufacturing information (such as MAC address, PCI vendor ID/device ID, etc.). If PXE is enabled, the boot code responds to the Expansion ROM accesses of system BIOS.

Boot code is executed when the Ethernet controller is reset via PCI Reset or S/W device reset. PXE initialization should only be necessary after a PCI reset. The boot code differentiates PCI Reset and driver initiated software reset by checking content in Internal Memory at 0xb50. If the content is 0x4B657654, then the reset is due to driver initiated software reset. Therefore, the device driver has to initialize 0xb50 with 0x4B657654 before issuing a S/W device reset.

Power Management

Description

The Ethernet controller is compliant with the PCI v2.0 power management specification. The MAC is programmable to two ACPI states: D0 and D3. The D0 state is a full power, operational mode—all the MAC core functions run at the highest clocking frequency, and components are fully functional. The MAC may be either initialized or un-initialized in the D0 ACPI state. An un-initialized D0 state is entered through a device reset or PME event; the MAC functional blocks are not started and initialized. Host software must reset/initialize hardware blocks to transition the device to a D0 initialized (active) state. The D0 active state places the device into a full power/operational mode. Receive and transmit data paths are fully operational, and the PCI block is initialized for bus mastering DMA.

Host device drivers do not differentiate between D3 hot and D3 cold states. ACPI-compliant device drivers are unloaded and quiescent in the D3 state and PCI slot power state is transparent. When the MAC is in D3 hot state, PCI slot power (3.3V or 5.0V) is available to power the PCI I/O pins. The PCI configuration and memory space may be accessed in D3 hot state. The core clock must remain enabled, so the MAC can respond to PCI configuration and memory transactions. The Disable_Core_Clock bit, in the PCI Clock Control register enables/disables clocking in the core clock domain. A D3 cold state provides only the PCI Vaux supply—PCI slot power is not present. The MAC will consume a maximum of 375 mA in a D3 cold power management mode.

The following functional blocks are integral to MAC power management:

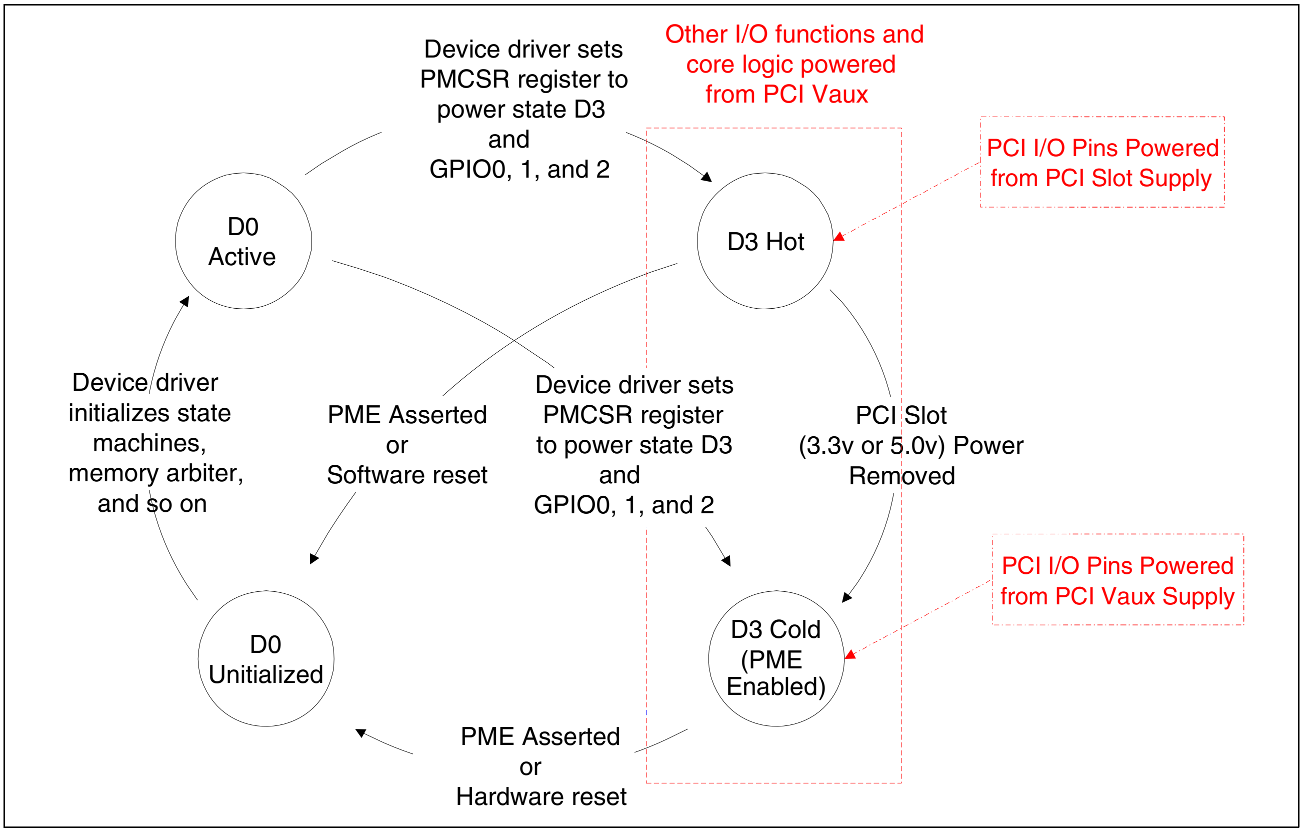
- PMSCR register
- PCI Clock Control register
- Miscellaneous Control register

- WOL
- PCI Vaux Supply
- PCI Slot Power Supply
- GPIO

Operational Characteristics

Figure 47 applies to the Ethernet controller reference designs. The MAC GPIO pins are available for application specific usage; however, Broadcom encourages both software and hardware engineers to follow the Broadcom design guidelines and application notes. NIC and LOM designs use external board level logic to switch power regulators for D3 ACPI mode.

Figure 47: Power State Transition Diagram


Device driver sets PMCSR register to power state D3 and GPIO0, 1, and 2
Other I/O functions and core logic powered from PCI Vaux
PCI I/O Pins Powered from PCI Slot Supply
D0 Active
D3 Hot
Device driver initializes state machines, memory arbiter, and so on
PME Asserted or Software reset
Device driver sets PMCSR register to power state D3 and GPIO0, 1, and 2
PCI Slot (3.3v or 5.0v) Power Removed
PCI I/O Pins Powered from PCI Vaux Supply
D0 Unitialized
D3 Cold (PME Enabled)
PME Asserted or Hardware reset

Device State D0 (Uninitialized)

The D0 state is entered after a PCI reset or device (software) reset. The assertion of PME causes the PCI bridge to drive RST. The MAC hardware blocks are not initialized in this state.

> ***Example:*** The RX engine, TX engine, multicast filter, and memory arbiter are all uninitialized. All the MAC functional blocks are powered.

Device State D0 (Active)

Host software has initialized the MAC hardware blocks. The RX and TX data paths are ready to send/receive Ethernet packets. The PCI block is available to DMA packets to host memory. This is a full power ACPI operational state. Host software/drivers must follow the initialization procedure (see "Initializing the MAC Hash Registers" on page 146) to move the MAC into a D0 active state.

When the BCM5725/BCM5762/BCM57767 NetXtreme device detects that main power is lost and it is still in the D0 state, it will reset itself to the D3 (Cold) state and then operate in 10/100 mode, like the OOB WOL state.

Device State D3 (Hot)

The MAC's configuration space and memory mapped I/O blocks are accessible in D3 hot state. The PCI I/O drivers are still powered by slot power in this state. However, host software has switched the MAC to use PCI Vaux for VDD_CORE and VDD_IO. GPIO pins 0, 1, and 2 are configured before the transition to this state. The RX RISC processor clock has been stopped in this state. The core clock remains active so PCI transactions may be processed by the MAC. This is a low-power state where some key components have been powered down. The physical layer auto-advertises 10 Mbps capability in this state, and link is set to 10 Mbps half-duplex or full-duplex. The PHY is configured for WOL mode. WOL pattern filters are initialized and active; the MAC will process Magic Packets™. The host chipset implements the power management policy for the PCI bus; the MAC driver does not influence the PCI Vaux or Slot power supply.

Note: The drivers should use configuration cycles (not the memory write cycles) to write to the PMCSR register at offset 0x4C for putting the device in D3 Hot state.

Device State D3 (Cold)

The MAC is completely powered by PCI Vaux in D3 cold. PCI configuration space and memory mapped I/O are not available. The only portion of the MAC active is the WOL pattern and Magic Packet filters[1]. The MAC will assert a PME in this state and indicate to the host bridge that a wake up event has occurred. The host bridge will normally provide PCI Slot power and then reset the device. GPIO pins 0, 1, and 2 are configured the same as D3 hot. Host software does not differentiate between D3 hot/cold. The MAC and PHY will not consume more than 375 mA in this mode. The integrated PHY must negotiate for 10 Mbps half/duplex speed. The PHY WOL mode is configured.

Note: The PCIe devices support the PCIe power management which is compatible with PCI bus power management.

Wake on LAN

See "Wake on LAN Mode/Low-Power" on page 202.

[1] Magic Packet™ is a registered trade mark of AMD.

GPIO

The use of GPIO pins for power management is design-specific, though Broadcom-delivered drivers use GPIO pins in the manner listed in Table 63. This usage is only applicable when the Ethernet controller is configured for a NIC design; it is not applicable to LAN-on-motherboard (LOM) designs. However, GPIO0 is used to control the NVRAM write-protect function in LOM designs (GPIO0 typically tied to WP pin of the BCM577XX NVRAM device).

Table 63: GPIO Usage for Power Management for Broadcom Drivers[a]

Function	Description	GPIO0	GPIO1	GPIO2
VAux	Sequence for switching to VAux	0	0	1
		1	1	1
		1	1	0
VMain	Sequence for switching to VMain	x	1	x
		x	0	x
		x	1	x

[a] x = Don't Care

Power Supply in D3 State

Table 64 shows the power supply to various power pins on the Ethernet controller, and it is assumed that host software has switched power regulators using GPIO pins 0, 1, and 2.

Table 64: Ethernet controller Power Pins

Pin	D3 Normal	D3 Hot	D3 Cold
VDD_CORE	PCI Slot	Vaux	Vaux
VDD_IO	PCI Slot	Vaux	Vaux
VDD_IO-PCI	PCI Slot	PCI Slot	No Power

Clock Control

Certain functional blocks in the MAC architecture should be powered down before a transition to D3 ACPI state. MAC clock generators/PLLs drive transistor level logic, which switch states on every clock pulse. Transistor level switching consumes power (mW). Software should selectively disable clocking to non-essential functional blocks. Software must set the Enable_Clock_Control_Register bit in the Miscellaneous Host Control register (see “Miscellaneous Host Control Register (offset: 0x68) — Function 0” on page 256); the assertion of this bit allows host software to configure the PCI clock control register. The following clock bits should be configured in the PCI Clock Control register:

- RX RISC clock disable
- Select alternate clock—the 133 MHz PLL is not used as reference clock.

Device ACPI Transitions

Host software must program the Power Management Control/Status (PMSCR) register to transition the device between D0 and D3 ACPI states. The Power State bit field in the PMSCR (see "Power Management" on page 181) may be programmed to D0, D1, D2, and D3 states.

Note: The D1 and D2 configurations are not supported in the Ethernet controller. The D1 and D2 bit configurations are available for applications, where D1 and D2 states are introduced for board level designs—the bits provide flexibility to the application. The Broadcom reference NIC/LOM designs do not use D1 and D2 states; therefore, host software should avoid setting these states. Before the Mac is moved into the D3 state, the clocks and GPIO must be configured (see above sections).

The PME signal is enabled in the PMSCR by asserting the PME_Enable bit. Device drivers/BIOS may also read the PME_Status bit to determine whether the event has been driven; PME_Status is a write to clear bit. The type and supported power management features for the Ethernet controller are reported in the Power Management Capabilities (PMC) register (see "Power Management Capability Register (offset: 0x48) — Functions 0" on page 252). System software and BIOS may read this register to enumerate and detect the power management features supported by the NIC/LOM. For example, the Ethernet controller can assert PME from both D3 hot and cold states. The PME_Support bit field in the PMC register will reflect this capability.

PCIe Active State Power Management

The BCM57765/BCM57765X PCIe interface supports the Active State Power Management (ASPM) functionality as specified in the PCIe 1.1 specification. The L0s-ASPM link state is supported. Both ASPM-L0s and ASPM-L1 link states are supported.

Although enabling the ASPM feature does not provide substantial power saving from the controller perspective, it does allow a system-level power saving. ASPM-L0s and ASPM-L1 support can be enabled (or disabled) by programming the ASPM-Control-Field. The device enters the corresponding ASPM state when there are no data transactions on the LAN and card reader.

Low Power (IDDQ) Mode

It is possible to optionally power down the device by asserting the LOW_PWR input pin or by setting the on-chip shared memory location 0xB50 (described below in "Disable Device Through BIOS"). The BCM57765/BCM57765X provides different firmware signatures to allow users to disable one or more PCIe functions associated with the device. The software method guarantees that the device is powered down to save power during an unexpected shutdown.

Disable Device Through BIOS

The Ethernet controllers can be disabled through BIOS by writing the value of DEADDEADh to shared memory location of B50h. This eliminates the need for BIOS to execute the device specific procedure for disabling the MAC device. The BIOS must do the following steps to disable the device.

1. Config cycle, write 88h to location 68h.
2. Config cycle, write 0B50h to location 7Ch.
3. Config cycle, write DEADDEADh to location 84h.

Note: The BIOS should first place the controller into the D3 power state prior to writing the 0xDEADDEAD signature value. See Table 62 on page 178 for power state control options (PCI Configuration register 0x4C bits 1:0).

The BCM5725/BCM5762/BCM57767 introduces a new power-down signature in addition to retaining the legacy 0xdeaddead signature functionality:

- 0xDEAD0111: Power-down card-reader but keep LAN function up

Endian Control (Byte and Word Swapping)

Background

There are two basic formats for storing data in memory—little-endian and big-endian. The endianess of a system is determined by how multibyte quantities are stored in memory. A big-endian architecture stores the most significant byte at the lowest address offset while little-endian architecture stores the least significant byte at the lowest address offset.

For example, the 32-bit hex value 0x12345678 would be stored in memory as shown in the following table.

Table 65: Endian Example

Address	***00***	***01***	***02***	***03***
Big Endian	12	34	56	78
Little Endian	78	56	34	12

Another method of viewing how this data would be stored is shown in the following tables.

Table 66: Storage of Big-Endian Data

Storage Byte	***00***	***01***	***02***	***03***
Data Contents	12	34	56	78

Table 67: Storage of Little-Endian Data

Storage Byte	***03***	***02***	***01***	***00***
Data Contents	12	34	56	78

Examples of big-endian platforms include SGI Irix, IBM RS6000, and SUN.

Examples of little-endian platforms include Intel x86 and DEC Alpha.

PCI assumes a little-endian memory model. PCI configuration registers are organized so that the least significant portion of the data is assigned to the lower address.

Architecture

The Ethernet controller is internally a big-endian machine, and its internal processors are big-endian devices. The Ethernet controller stores data internally in big-endian format using a 64-bit memory subsystem.

However, many hosts (e.g., x86 systems) use the little-endian format, and the PCI bus uses the little-endian format. Therefore the Ethernet controller has a number of byte swapping options that may be configured by software so that Little or Big Endian hosts can interface as seamlessly as possible with Ethernet controller over PCI. The Ethernet controller has the following bits that control byte and word swapping:

- Enable Endian Word Swap (bit 3, Miscellaneous Host Control register (offset 0x68 into PCI Config register, se "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). If 1, this register enables 32-bit word swapping when accessing the Ethernet controller via the PCI target interface.
- Enable Endian Byte Swap (bit 2, Miscellaneous Host Control register (offset 0x68 into PCI Config register, see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). If 1, this register enables byte swapping (within a 32-bit word) when accessing the Ethernet controller via the PCI target interface.
- Word Swap Data (bit 5, Mode Control register (offset 0x6800 into the Ethernet controller registers). If 1, this register enables word swapping of frame data when it comes across the bus.
- Byte Swap Data (bit 4, Mode Control register (offset 0x6800 into the Ethernet controller registers). If 1, this register enables byte swapping of frame data when it comes across the bus.
- Word Swap Non-Frame Data (bit 2, Mode Control register (offset 0x6800 into the Ethernet controller registers). If 1, this register enables word swapping of non frame data (i.e,. buffer descriptors, statistics, etc.) when it comes across the bus.
- Byte Swap Non-Frame Data (bit 1, Mode Control register (offset 0x6800 into the Ethernet controller registers). If 1, this register enables byte swapping of non frame data (i.e., buffer descriptors, statistics, etc.) when it comes across the *bus*.

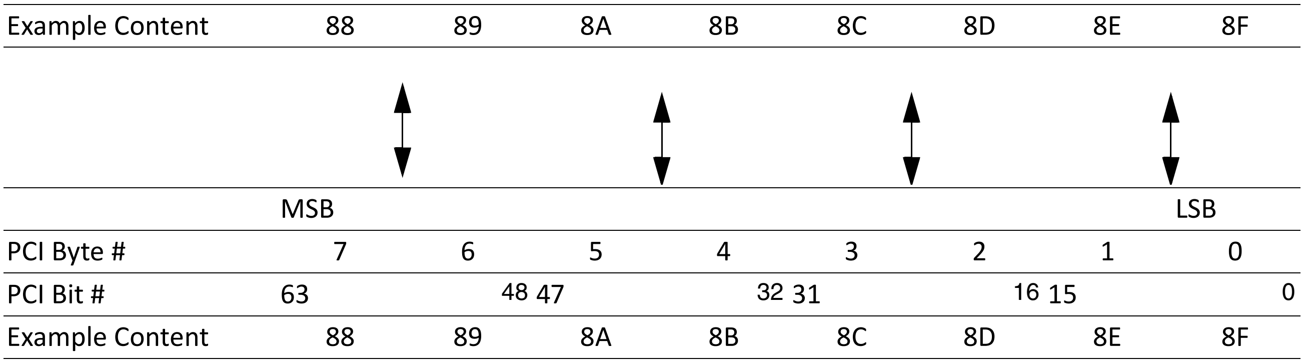
The setting of the above swapping bits will affect the order of how data is represented when it is transferred across PCI. Since byte swapping is a confusing subject, examples will be shown that reflect how each byte swapping bit works

Enable Endian Word Swap and Enable Endian Byte Swap Bits

The Enable Endian Word Swap, and Enable Endian Byte Swap bits affect whether words or bytes are swapped during target PCI accesses. Thus, these bits affect the byte order when the host is directly reading/writing to registers or control structures that are physically located on the Ethernet controller. These bits do not affect the byte ordering of packet data or other structures that are mastered (DMAed) by the Ethernet controller.

When the Ethernet controller is accessed via PCI (which is little endian) as a PCI target, the Ethernet controller must implicitly map those accesses to internal structures that use a 64-bit Big Endian architecture. In the default case where no swap bits are set the Ethernet controller maps PCI data to internal structures shown in Figure 48 and Figure 49.

	MSB							LSB
Internal Byte #	0	1	2	3	4	5	6	7
Internal Bit #	63	48	47	32	31	16	15	0



Example Content	88	89	8A	8B	8C	8D	8E	8F

	MSB							LSB
PCI Byte #	7	6	5	4	3	2	1	0
PCI Bit #	63	48 47		32 31		16 15		0
Example Content	88	89	8A	8B	8C	8D	8E	8F

Figure 48: Default Translation (No Swapping) on 64-Bit PCI

Internal Byte OrderingPCI Byte Ordering

	31	16	15	0
0x00	88	89	8A	8B
0x04	8C	8D	8E	8F

31	16	15	0	
8C	8D	8E	8F	0x00
88	89	8A	8B	0x04

Figure 49: Default Translation (No Swapping) on 32-bit PCI

As illustrated above, because the Ethernet controller uses an internal 64-bit big endian architecture, it will map (by default) the most significant byte of an 8-byte (64-bit) internal quantity to the most significant byte on a 64-bit PCI bus. This works nicely for quantities (fields) that are 64 bits in size (e.g., a host physical address). However, this can be confusing for quantities that are 32 bits in size. Without Word Swapping enabled, the host could easily access the wrong 32-bit quantity when making a 32-bit access.

Take, for example, a Ring Control Block (RCB). RCBs are on-chip structures and read/written by the host via PCI target accesses. The table below shows the big-endian layout of an on-chip RCB.

Table 68: RCB (Big Endian 32-Bit Format)

Byte #	0	1	2	3	
Bit #	31	16	15	0	
	MSB Host Ring Address				0x00
				LSB	0x04
	MSB MAX_Len	LSB	Flags		0x08
	NIC Ring Address				0x0C

If Word Swapping is not enabled, and the host made a 32-bit read request to address 0x08, the four bytes of data returned on the PCI bus would actually be the NIC Ring Address rather than the Max_Len and Flags fields. This initially might seem counter-intuitive, but is explained in Figure 49 on page 189. Therefore, if a software driver running on an x86 host (Little Endian) referenced on-chip data structures as they are defined in the Ethernet controller data sheet, the driver should set the Enable Endian Word Swap bit. By setting this bit, the translation would be as follows:

Internal Byte OrderingPCI Byte Ordering

31	16	15	0		31	16	15	0

0x00	88	89	8A	8B	88	89	8A	8B	0x00
0x04	8C	8D	8E	8F	8C	8D	8E	8F	0x04

Figure 50: Word Swap Enable Translation on 32-Bit PCI (No Byte Swap)

The only side effect for a little endian host that sets the Enable Endian Word Swap bit would be that the driver would have to perform an additional word swap on any 64-bit fields (e.g., a 64-bit physical address) that were given to the driver by the Network Operating System (NOS).

Little-endian hosts will not want to set the Enable Endian Byte Swap bit for target accesses. This bit is intended to be used by big endian systems that needed PCI data (little endian) translated back to big endian format.

Note: Some big endian systems automatically do this depending on the architecture of the host’s PCI to memory interface.

The following figures show the translation of data when the Enable Endian Byte Swap bit is set:

	Internal Byte Ordering				PCI Byte Ordering				
	31	16	15	0	31	16	15	0	
0x00	88	89	8A	8B	8F	8E	8D	8C	0x00
0x04	8C	8D	8E	8F	8B	8A	89	88	0x04

Figure 51: Byte Swap Enable Translation on 32-Bit PCI (No Word Swap)

	Internal Byte Ordering				PCI Byte Ordering				
	31	16	15	0	31	16	15	0	
0x00	88	89	8A	8B	8B	8A	89	88	0x00
0x04	8C	8D	8E	8F	8F	8E	8D	8C	0x04

Figure 52: Byte and Word Swap Enable Translation on 32-Bit PCI

Word Swap Data and Byte Swap Data Bits

The Word Swap Data, and Byte Swap Data bits effect how packet data is ordered on the PCI bus. These only affect how packet data is ordered, and do not affect non-frame data (i.e., buffer descriptors, statistics block, etc.). In other words, these bits effect how data is transferred to/from host send/receive buffers.

Example: If Ethernet controller were to receive a packet that had the following byte order:

01	*02*	*03*	*04*	*05*	*06*	*07*	*08*	*09*	*0A*	*0B*	*0C*	*0D*	*0E*	*0F*	*10*
D1	D2	D3	D4	D5	D6	S1	S2	S3	S4	S5	S6	T1	T2	IP1	IP2

Where:

- D1–D6 consists of the packet's destination address (Byte D0 is the first byte on the wire);
- S1–S6 is the source address;
- T1–T2 is the Ethernet type/length field;
- IP1–IP2 are the first two bytes of the IP header which immediately follow the type/length field.

The packet would be stored internally in big endian format:

Table 69: Big-Endian Internal Packet Data Format

B0	*B1*	*B2*	*B3*	*B4*	*B5*	*B6*	*B7*
63–56	*55–48*	*47–40*	*39–32*	*31–24*	*23–16*	*15–8*	*7–0*
D1	D2	D3	D4	D5	D6	S1	S2
S3	S4	S5	S6	T1	T2	IP1	IP2

However, when the data gets transferred across PCI, there could be confusion about the correct byte ordering because PCI is Little Endian whereas Ethernet controller is a Big Endian device. So, in order to provide flexibility for different host processor/memory architectures, Ethernet controller can order this data on PCI in four different ways depending on the settings of the Word Swap Data, and Byte Swap Data bits. The following figures illustrate how data would appear on the PCI AD[63:0] pins depending on the settings of those swap bits:

Word Swap Data = 0, and Byte Swap Data = 0

Table 70: 64-Bit PCI Bus (WSD = 0, BSD = 0)

B7	*7B6*	*B5*	*B4*	*B3*	*B2*	*B1*	*B0*
63–56	*55–48*	*47–40*	*39–32*	*31–24*	*23–16*	*15–8*	*7–0*
D1	D2	D3	D4	D5	D6	S1	S2
S3	S4	S5	S6	T1	T2	IP1	IP2

Table 71: 32-Bit PCI Bus (WSD = 0, BSD = 0)

B3	*B2*	*B1*	*B0*

Table 71: 32-Bit PCI Bus (WSD = 0, BSD = 0)

31–24	***23–16***	***15–8***	***7–0***
D5	D6	S1	S2
D1	D2	D3	D4
T1	T2	IP1	IP2
S3	S4	S5	S6

Word Swap Data = 0, and Byte Swap Data = 1

Table 72: 64-Bit PCI Bus (WSD = 0, BSD = 1)

B7	***B6***	***B5***	***B4***	***B3***	***B2***	***B1***	***B0***
63–56	***55–48***	***47–40***	***39–32***	***31–24***	***23–16***	***15–8***	***7–0***
D4	D3	D2	D1	S2	S1	D6	D5
S6	S5	S4	S3	IP2	IP1	T2	T1

Table 73: 32-Bit PCI Bus (WSD = 0, BSD = 1)

B3	***B2***	***B1***	***B0***
31–24	***23–16***	***15–8***	***7–0***
S2	S1	D6	D5
D4	D3	D2	D1
IP2	IP1	T2	T1
S6	S5	S4	S3

Word Swap Data = 1, and Byte Swap Data = 0

Table 74: 64-Bit PCI Bus (WSD = 1, BSD = 0)

B7	*B6*	*B5*	*B4*	*B3*	*B2*	*B1*	*B0*
63–56	*55–48*	*47–40*	*39–32*	*31–24*	*23–16*	*15–8*	*7–0*
D5	D6	S1	S2	D1	D2	D3	D4
T1	T2	IP1	IP2	S3	S4	S5	S6

Table 75: 32-Bit PCI Bus (WSD = 1, BSD = 0)

B3	*B2*	*B1*	*B0*
31–24	*23–16*	*15–8*	*7–0*
D1	D2	D3	D4
D5	D6	S1	S2
S3	S4	S5	S6
T1	T2	IP1	IP2

Word Swap Data = 1, and Byte Swap Data = 1

Table 76: 64-Bit PCI Bus (WSD = 1, BSD = 1)

B7	*B6*	*B5*	*B4*	*B3*	*B2*	*B1*	*B0*
63–56	*55–48*	*47–40*	*39–32*	*31–24*	*23–16*	*15–8*	*7–0*
S2	S1	D6	D5	D4	D3	D2	D1
IP2	IP1	T2	T1	S6	S5	S4	S3

Table 77: 32-Bit PCI Bus (WSD = 1, BSD = 1)

B3	*B2*	*B1*	*B0*
31–24	*23–16*	*15–8*	*7–0*
D4	D3	D2	D1
S2	S1	D6	D5
S6	S5	S4	S3
IP2	IP1	T2	T1

So, for a little-endian (e.g., x86) host, software should set both the Word Swap Data, and Byte Swap Data bits. This is because a little endian host will expect the first byte on the wire (byte D1) to be placed into memory at the least significant (starting) address of the packet data.

Word Swap Non-Frame Data and Byte Swap Non-Frame Data Bits

The Word Swap Non-Frame Data, and Byte Swap Non-Frame Data bits affect the byte ordering of certain shared memory data structures (buffer descriptors, statistics block, etc.) when those structures are transferred across PCI.

The following table shows as example of how a Send Buffer Descriptor is stored internally in the Ethernet controller.

Table 78: Send Buffer Descriptor (Big-Endian 64-Bit format)

Byte #	0 1	2 3	4 5	6 7	
Bit #	63 48	47 32	31 16	15 0	
	MSB Host Address LSB				0x00
	MSB Length LSB	Flags	Reserved	VLAN	0x08

Since the Ethernet controller uses a 64-bit memory subsystem, the above diagram is shown in 64-bit format. Furthermore, the table shows both the internal byte offset for each field and the bit position for each byte.

Note: This may seem confusing because big-endian notation normally has the bit positions incrementing from left to right. However, in this case, the bit positions are relevant because they correspond to the bit positions on PCI (AD[63:0]) if neither of the non-frame data swap bits are set. For clarification, the following table shows the same structure in 32-bit format.

Table 79: Send Buffer Descriptor (Big-Endian 32-Bit Format)

Byte #	0 1	2 3	
Bit #	31 16	15 0	
	MSB Host Address		0x00
		LSB	0x04
	MSB Length LSB	Flags	0x08
	Reserved	VLAN	0x0C

To provide flexibility for different host processor/memory architectures, the Ethernet controller can order the data in memory in four different ways depending on the settings of the Word Swap Non-Frame Data and Byte Swap Non-Frame Data bits. The following tables show how data will appear depending on the settings of those swap bits:

Word Swap Non-Frame Data = 0 and Byte Swap Non-Frame Data = 0

This would require the software to use the following little-endian data structure on the host:

Table 80: Send Buffer Descriptor (Little-Endian 32-Bit Format) with No Swapping

Byte #	3	2	1	0	
Bit #	31	16	15	0	
	Host Address			LSB	0x00
	MSB				0x04
	Reserved		VLAN		0x08
	MSB Length LSB		Flags		0x0C

In this case, the data structure takes on a slightly new format because the words have been swapped.

Word Swap Non-Frame Data = 1 and Byte Swap Non-Frame Data = 0

This requires the software to use the following little-endian data structure on the host:

Table 81: Send Buffer Descriptor (Little-Endian 32-Bit format) with Word Swapping

Byte #	3	2	1	0	
Bit #	31	16	15	0	
	MSB Host Address				0x00
				LSB	0x04
	MSB Length LSB		Flags		0x08
	Reserved		VLAN		0x0C

The disadvantage of this approach is if the host operating system supported a 64-bit data type for a physical address, the host device driver would have to swap the two 32-bit words that comprise the 64-bit address that the host operating system used.

Word Swap Non-Frame Data = 0 and Byte Swap Non-Frame Data = 1

This requires the software to use the following big-endian data structure on the host:

Table 82: Send Buffer Descriptor (Big-Endian 32-bit format) with Byte Swapping

Byte #	0	1	2	3	
Bit #	31	16	15	0	
	MSB Host Address				0x00
				LSB	0x04
	Reserved		VLAN		0x08
	MSB Length LSB		Flags		0x0C

Word Swap Non-Frame Data = 1 and Byte Swap Non-Frame Data = 1

This requires the software to use the following big-endian data structure on the host:

Table 83: Send Buffer Descriptor (Big-Endian 32-bit format) with Word and Byte Swapping

Byte #	0	1	2	3	
Bit #	31	16	15	0	
	MSB Host Address				0x00
				LSB	0x04
	MSB Length LSB		Flags		0x08
	Reserved		VLAN		0x0C

Section 9: Ethernet Link Configuration

Overview

The Ethernet controller supports multiple link operating modes. It can operate at multiple link speeds: 10 Mbps, 100 Mbps, or 1000 Mbps. It can also operate at half-duplex (IEE 802.3 CSMA/CD) or full-duplex. The MAC is compliant with IEEE 802.3, IEEE 802.3u, IEEE 802.3x, and IEEE 802.3z specifications.

GMII/MII

The Gigabit Media Independent Interface (GMII) is normally used to interface the controller to a transceiver that supports Gigabit Ethernet over copper wiring (1000BASE-T). The Media Independent Interface (MII) is used to interface the controller to a transceiver that is capable of 10/100 Mbps Ethernet. Depending upon the link speed, driver software will need to configure the Ethernet controller to operate in either GMII mode or MII mode.

Note: The integrated PHY transceiver of the BCM5725/BCM5762/BCM57767 has a has a fixed PHY address value of 1.

Configuring the Ethernet Controller for GMII and MII Modes

Configuring the Ethernet controller to operate in GMII or MII mode is simple. During initialization, software should configure the Ethernet_MAC_Mode.Port_Mode bits to a value that corresponds to the correct interface speed (01b for MII, 10b for GMII).

Configuring How MAC Detects Link Up/Down

The Ethernet controller has two different methods that it can use to determine if the Ethernet link is up or down. The link will be down if the Ethernet cable is not properly attached at both ends of the network. Link will be up only if the cable is properly attached and the devices at both ends of the cable recognize that link has been established. The device cannot successfully transfer packets on the link unless it determines that it has a valid link up.

The first method is called auto-polling. Software can enable auto-polling by setting the MII_Mode.Port_Polling bit. If enabled, the Ethernet controller periodically generates MDIO cycles to read the PHY’s Link_Status bit in MII Status register. The link status from the auto-polling operation is then reported in the MII_Status_Register.Link_Status and Transmit_MAC_Status.Link_Up (see “Transmit MAC Status Register (offset: 0x460) ” on page 331) bits.

The second method involves using the Ethernet controller's LNKRDY input signal. This method allows the Ethernet controller to determine the link status based on the link status output from integrated PHY connected to LNKRDY input of MAC. The Transmit_MAC_Status.Link Up bit (see "Transmit MAC Status Register (offset: 0x460) " on page 331) will reflect the link status input on LNKRDY signal to MAC from PHY. Host software can enable this method by clearing the MII_Mode.Port_Polling bit (bit-4 of offset 0x454).

The link state of the Ethernet controller can also be forced by disabling both the auto-polling function and the LNKRDY signal and forcing the link status by directly writing to the MII_Status.Link_Status bit.

Link Status Change Indications

It is advantageous for host software to know when the status of the Ethernet link has changed. To generate an interrupt to the host when link status changes, software should set the Ethernet_MAC_Event_Enable.Link_State_Changed bit (see "EMAC Event Enable Register (offset: 0x408)" on page 323) and the Mode_Control.Interrupt_on_MAC_Attention bit (see "Mode Control Register (offset: 0x6800)" on page 521). With this configuration, the Ethernet_MAC_Status.Link_State_Changed bit and Link_State_Changed bit in the status block (see "Status Block" on page 82) will be set when the link has changed state.

Configuring the GMII/MII PHY

GMII/MII transceivers (PHYs) contain registers that a software driver can manipulate to change parameters in the PHY. These parameters include the link speed or duplex that the PHY is currently running at, or the speed/ duplex options that the PHY advertises during the auto-negotiation process. NIC device drivers will typically access PHY registers during the driver initialization process to configure the PHYs speed/duplex or to examine the results of the auto-negotiation process.

The integrated PHY registers are accessed via a process called MDIO. The integrated PHY is connected to the Ethernet controller through an internal MDIO bus (MDIO and MDC pins). Software accesses PHY's registers via MDIO through the Ethernet controller's MII_Communication register. The following example code describes accessing the PHY registers through the MII_Communication registers of the Ethernet controller.

Reading a PHY Register

```
// If auto-polling is enabled, temporarily disable it
If (AutoPolling_Enabled == TRUE) Then
Begin
    Mi_Mode.PortPolling = 0
End
// Setup the value that we are going to write to MII Communication register
// Set bit 27 to indicate a PHY read.
// Set bit 29 to indicate the start of a MDIO transaction
Value32 = ((PhyAddress << 21) | (PhyRegOffset << 16) | 0x28000000)
// Write value to MII communication register
Mi_Communication_Register = Value32
// Now read back MII Communication register until the start bit
// has been cleared or we have timed out (>5000 reads)
Loopcount = 5000
While (LoopCount > 0)
Begin
    Value32 = Mi_Communication_Register
    If (!(Value32 | 0x20000000)) then BREAK loop
    Else Loopcount--
End
// Print message if error
If (Value32 | 0x20000000) then
Begin
    // It a debug case – cannot read PHY
    Procedure (Print Error Message)
    Value32 = 0
End
// If auto-polling is enabled, turn it back on
If (AutoPolling_Enabled == TRUE) then
Begin
    Mi_Mode.PortPolling = 1
End
// Now return the value that we read (lower 16 bits of reg)
Return (Value32 & 0xffff)
```

Writing a PHY Register

```
// If auto-polling is enabled, temporarily disable it
If (AutoPolling_Enabled == TRUE) Then
Begin
    Mi_Mode.PortPolling = 0
End
// Setup the value that we are going to write to MII Communication register
// Set bit 26 set to indicate a PHY write.
// Set bit 29 to indicate the start of a MDIO transaction
// The lower 16 bits equal the value we want to write to the PHY register
Value32 = ((PhyAddress << 21) | (PhyRegOffset << 16) | RegValue | 0x24000000)
// Write value to MII communication register
Mi_Communication_Register = Value32
// Now read back MII Communication register until the start bit
// has been cleared or we have timed out (>5000 reads)
Loopcount = 5000
While (LoopCount > 0)
Begin
    Value32 = Mi_Communication_Register
    If (!(Value32 | 0x20000000)) then BREAK loop
    Else Loopcount--
End
// Print message if error
If (Value32 | 0x20000000) then
Begin
    // It a debug case – can't write PHY
    Procedure (Print Error Message)
    Value32 = 0
End
// If auto-polling is enabled, turn it back on
If (AutoPolling_Enabled == TRUE) Then
Begin
    Mi_Mode.PortPolling = 1
End
```

MDI Register Access

Configuring physical devices and querying the status of physical devices are done via the MDIO interface (MDC and MDIO).

Note: This procedure is PHY-independent. The MAC access to the PHY is the same for the entire NetXtreme family.

There are two modes in which the internal MII Management interface signals (MDC/MDIO) can be controlled for communication with the internal transceiver registers. These modes are as follows:

- Autopolling Mode. Enabled by setting the Enable bit in the MAC Ethernet MII Mode register. The device will poll for the link status bit in the transceiver.
- Command Control. Writing to the MII Communications register directly to either read or write the transceiver registers.

Autopolling Mode has the lowest priority and it will be stalled any time there is an active operation through the MII Communications register.

Operational Characteristics

The interface between the MAC and physical devices is with the two signals of:

- MDIO clock (MDC)
- Bidirectional serial data (MDIO)

The details of the MDIO interface can be found in the IEEE 802.3 specification.

Access Method

The MAC provides the auto-access method to access the Physical Device register via the MDIO interface.

Auto-Access Method

The Ethernet controller has a built-in interface to access physical device registers without having to control MDC and MDIO pins by software/firmware. It provides an easy way to access the physical device register.

To use this mode, MDI_Control_Register.MDI_Select has to be cleared to 0. The MII_Communication_Register is used to access physical device.

Note: Programmers must be careful to wait for the start _busy bit to clear. Writing to the MII Communication register prior to the completion of a previous MDI access will yield unpredictable MDI data. The previous access will not complete successfully.

For example, to read a 16-bit PHY register at offset 0x2 of a PHY device which is strapped to PHY address 1, perform the following steps:

1. MII_Communication_Register.Register_Address is set to 0x2.
2. MII_Communication_Register.PHY_Addr is set to 1.
3. MII_Communication_Register.Command is set to 0x2.
4. MII_Communication_Register.Start_Busy is set to 1.
5. Poll Until MII_Communication_Register.Start_Busy is cleared to 0.
6. MII_Communication_Register.Transaction_Data contains 16-bit data of the PHY register.

See "Configuring the GMII/MII PHY" on page 198 for example code.

To write a value of 0x1000 into 16-bit PHY register at offset 0x0 of a PHY device which is strapped to PHY address 1, perform the following steps:

1. MII_Communication_Register.Register_Address is set to 0x0.
2. MII_Communication_Register.PHY_Addr is set to 1.
3. MII_Communication_Register.Command is set to 0x1.
4. MII_Communication_Register.Transaction_Data is set to 0x1000
5. MII_Communication_Register.Start_Busy is set to 1.
6. Poll Until MII_Communication_Register.Start_Busy is cleared to 0.

See "Configuring the GMII/MII PHY" on page 198 for example code.

Wake on LAN Mode/Low-Power

Description

The Ethernet controller uses the ACPI D3 hot/cold (low-power) state to conserve energy. The OS power management policy notifies device drivers to initiate power management transitions. The device driver should move the MAC into the D3 hot/cold power state—a response to the power management request. While the Ethernet controller is in a D3 state, the RX MAC will filter incoming packets. The RX MAC compares incoming traffic for Interesting Packet pattern matches. The Ethernet controller asserts the PCI PME signal, when a positive WOL packet comparison is made. The PME signal notifies the Operating System and host device driver to transition the MAC into the D0 (high power) state.

WOL mode is a combination of PHY and MAC configurations. Both the PHY and MAC must be configured correctly to enable Broadcom's WOL technology. The Ethernet controller provides WOL pattern filters for 10/100 wire speeds.

The Ethernet controller supports both Interesting Packet pattern matching the AMD Magic Packet proprietary technology for WOL. The WOL support for the AMD Magic Packet format does not require host software to configure a pattern filter. The Magic Packet comparison is made in hardware and is enabled through a register interface. The AMD Magic Packet can be either broadcast or directed, and must contain the receiver's MAC address at least six times (repeating) in the packet. The Magic Packet wake-up is configured different from pattern match wake-up.

The following components are involved in WOL operation:

- Internal memory
- WOL Pattern Pointer register
- WOL Pattern Configuration register
- WOL streams
- Pattern data structure
- GPIO
- Firmware mailbox
- PHY auto-negotiation
- Ethernet controller power management

Functional Overview

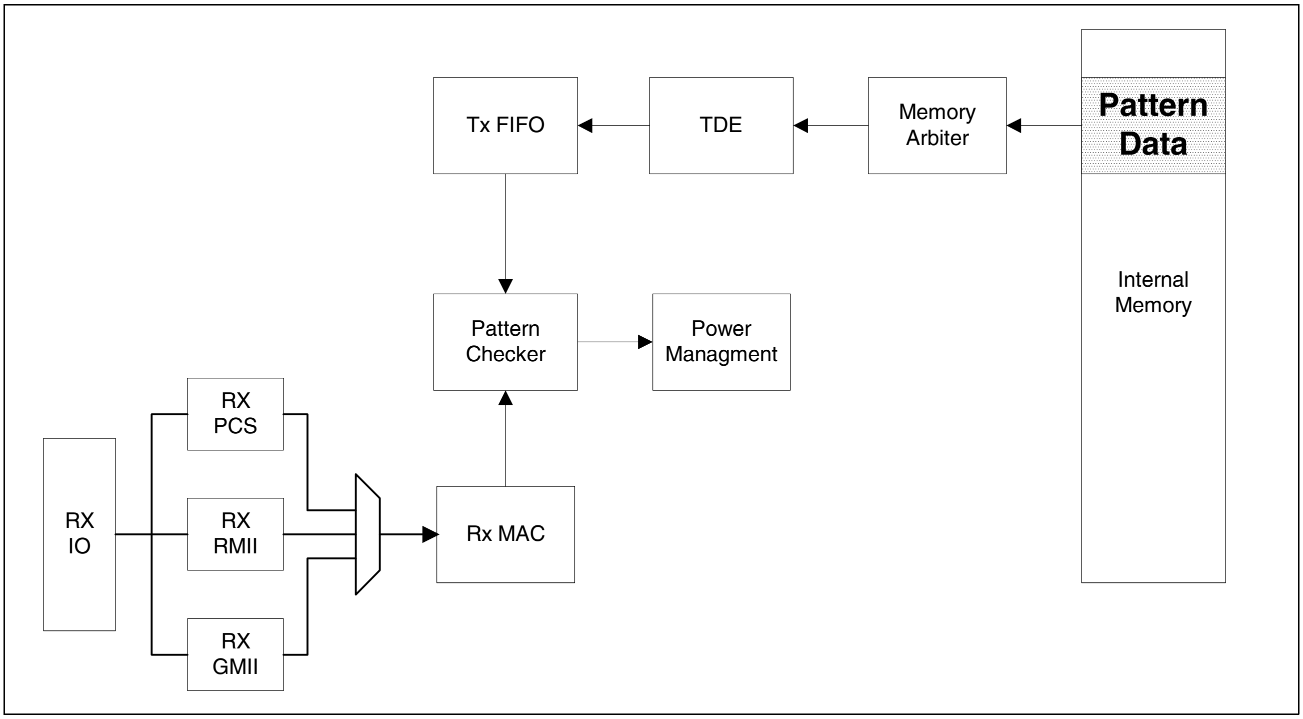
The Ethernet controller is capable of WOL in 10/100 Mbps for copper-based controllers.

Note: When configured for WOL in 1000 Mbps mode, the Ethernet controller draws more than the 375 mA allowed by the PCI specification.

The Ethernet controller uses the TX FIFO to store pattern data (see Figure 53). During WOL operation, the transmit engine is disabled and its FIFO is free for use. The TDE fetches data from the memory arbiter starting at a location specified in the WOL_Pattern_Pointer register. The WOL pattern checker pulls data off the TX FIFO for packet comparisons. The RX MAC will move incoming frame(s) to the pattern checker, and the remaining RX data path is not utilized. A state machine controls the Magic Packet comparisons. The WOL state machine will move out of an Idle state, when ACPI power management is enabled. The WOL state machine will clear the TX FIFO and Match register. The Match register indicates a positive Magic Packet comparison(s) on a stream.

In 10/100 Mbit mode, data is received once every four clock cycles. The pattern checker compares the first three patterns in the first cycle, the second three patterns in the second cycle, and the third three patterns in the last cycle. It is idle during the fourth cycle. In gigabit mode, the pattern checker gets three pattern words from the FIFO at one time.

Figure 53: WOL Functional Block Diagram



Operational Characteristics

Internal Memory

The WOL pattern must be stored in the Ethernet controller miscellaneous memory region. All memory locations require the host software to reinitialize the WOL pattern before each D0 to D3 transition. The RX/TX MAC places packets into this internal memory and the WOL pattern is overwritten during normal operation. When the Ethernet controller operates in D0 state, internal data structures use the same memory location as the WOL pattern. Host software should reinitialize the WOL pattern before each WOL sleep transition.

Table 84 shows the required memory regions for the WOL pattern.

Table 84: Required Memory Regions for WOL Pattern

Internal Address Range	*Size*	*Name*
0x8000–0x8FFF	8 KB	Miscellaneous Memory Region

WOL Pattern Pointer Register

The WOL_Pattern_Pointer specifies a location within Ethernet controller address space where the pattern buffers reside (see "WOL Pattern Pointer Register (offset: 0x430)" on page 326 for the register definition). The internal memory subsection discusses how host programmers can choose an address range. The WOL_Pattern_Pointer register uses a pointer value, not an internal memory location. The pointer value is calculated by dividing an internal memory location by the value 8. Do not program the WOL_Pattern_Pointer register with the actual internal memory location. Rather, host software must first convert the base address to a pointer value. Here are example conversion from memory base to pointer values:

- 0x0000 (Misc Memory)/8 = 0x00 (required value)
- 0x400 (base addr)/8 = 0x80 (pointer value)
- 0x8000 (base addr)/8 = 0x1000 (pointer value)
- 0xF000 (base addr)/8 = 0x1E00 (pointer value)

WOL Pattern Configuration Register

The WOL_Pattern_Configuration register contains two programmable data fields. Both fields use different units of measurement, so the host programmer should be careful (see for the register definition). This register is used to position and extract data from RX Ethernet frames.

- Offset Field—The Offset field in the WOL_Pattern_Configuration register specifies a position in RX Ethernet frame(s), where comparisons for WOL patterns should begin. This register uses a unit of measurement specified in terms of 2-byte chunks. Software should not program this field with a byte value, but should first normalize to a 2-byte unit. Hardware cannot begin WOL comparisons on odd byte alignments (i.e., 3,5,7,9 offsets). Host software must begin all pattern matching on even byte boundaries (i.e., 2,4,6,8 offsets). The 2 bytes per unit forces even byte alignment. For example:
  - 0x14 (byte offset)/2 = 0x0A (register ready)
  - 0x28 (byte offset)/2 = 0x14 (register ready)
  - 0xFC (byte offset)/2 = 0x7E (register ready)

- Length Field—The Length field in the WOL_Pattern_Configuration register specifies the number of clock cycles required to compare a variable number of bytes, in the RX stream. The Length field uses a unit of measurement specified in terms of memory arbiter clock cycles. Software should not program this field with a byte value. The Length field should be programmed with the maximum number of clocks required to compare the largest pattern size for the nine streams (10/100 mode only).

  Note: The Ethernet controller only supports one pattern stream at gigabit wire speed, so the length field will always be the largest pattern size.

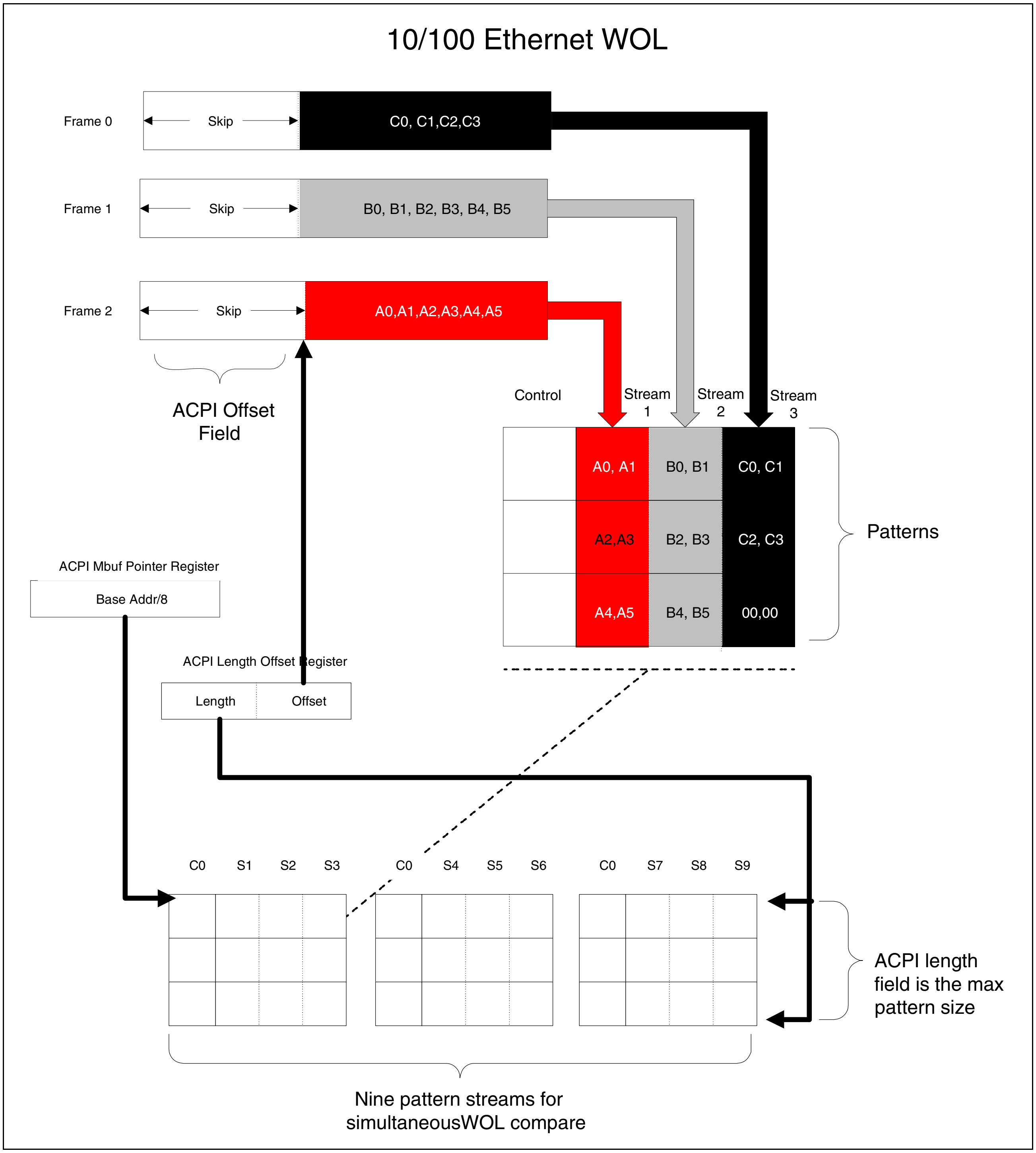
  The programmer must use the following equation to calculate the number of clock cycles required to match patterns at 10/100 wire-speed: (Length/2) * 3 MA clocks. The equation breaks down as follows:

  - Determine the number of bytes in the RX Ethernet frame to compare. This value is a byte length.
  - The WOL pattern checker can compare two bytes simultaneously. Divide length by two bytes and round up to nearest integer value.
  - The Ethernet controller compares 2 bytes every three memory arbiter (MA) clock cycles. Multiply (Length/2) by three clock cycles.
- The following are example clock cycle calculations:
  - Data stream length = 25 bytes
  - 25 bytes/2 = 12.5 byte-pairs
  - Round(12.5) = 13 byte-pairs
  - 13 byte-pairs * 3 clocks/byte-pairs = 39 clocks (register ready)
  - Data stream length = 83 bytes
  - 83 bytes/2 = 41.5 byte-pairs
  - Round(41.5) = 42 byte-pairs
  - 42 byte-pairs * 3 clocks/byte-pair = 126 clocks (register ready)

WOL Streams

A stream is a comparison operation on RX frame(s). When the MAC is running at 10/100 Mbps wire speed, nine different patterns can be compared against the RX frame(s). The Ethernet controller moves RX frame(s) into nine parallel comparators, and the frame is matched simultaneously. The MAC is capable of filtering nine different patterns in 10/100 modes. The WOL pattern checker breaks frames into 2-byte pairs, so all nine comparators can begin matching data. In Figure 54 on page 206, three Ethernet frames are compared against the nine available patterns.

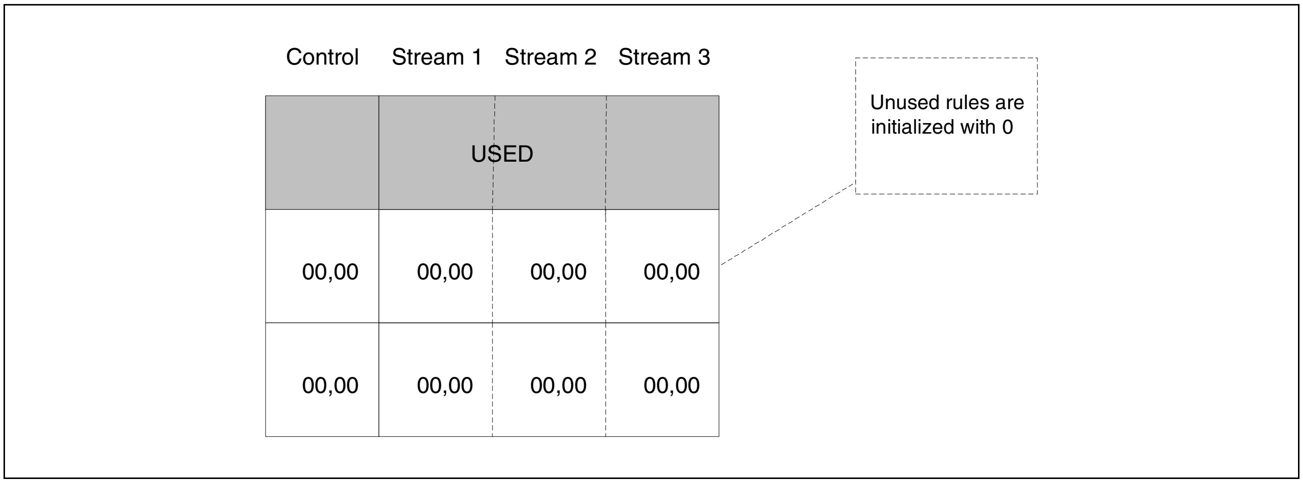
Figure 54: Comparing Ethernet Frames Against Available Patterns (10/100 Ethernet WOL)



Pattern Data Structure

The maximum number of entries in either 10/100 or 1000 mode is 128. The Ethernet controller cannot process a pattern that requires more than 128 entries. The size of an entry will vary based on 10/100 or 1000 Mbps mode. Additionally, all unused rows must be initialized with zeros. The WOL hardware cannot process an entry unless unused rows and rules have been zeroed out (see Figure 55).

Figure 55: Unused Rows and Rules Must Be Initialized with Zeros



Frame patterns are stored as data structures in memory. A control word is always present in a 64 bit entry/row. The control word describes proceeding data fields in the entry.

In 10/100 Mbps mode, one WOL entry requires three 64-bit wide rows (see Table 85). The total length of an entry is 192 bits. Each 64-bit row contains a 16-bit control word, which identifies byte enables (see Table 86). The remaining 48-bits contains 2-byte rules. The 2-byte rules are distributed across three streams: S, S+1, and S+2. The next row's 2-byte rules will correspond to three more streams: S+3, S+4, and S+5. Both Table 85 and Table 86 use Sx notation to denote separate comparison streams. The D0 notation indicates the first 2 bytes in the packet stream are compared.

Table 85: 10/100 Mbps Mode Frame Patterns Memory

63 48	47 32	31 16	15 0
CTRL012	S0D0	S1D0	S2D0
CTRL345	S3D0	S4D0	S5D0
CTRL678	S6D0	S7D0	S8D0

Table 86: Frame Control Field for 10/100 Mbps Mode

Bits	Field	Description	Access
63:62	Reserved		
61	S0 High Byte Enable	Enable S0 higher byte for comparison	R/W
60	S0 Low Byte Enable	Enable S0 lower byte for comparison	R/W

Table 86: Frame Control Field for 10/100 Mbps Mode (Cont.)

Bits	Field	Description	Access
59	S1 High Byte Enable	Enable S1 higher byte for comparison	R/W
58	S1 Low Byte Enable	Enable S1 lower byte for comparison	R/W
57	S2 High Byte Enable	Enable S2 higher byte for comparison	R/W
56	S2 Low Byte Enable	Enable S2 lower byte for comparison	R/W
55:51	Reserved	–	–
50	S0 Done	End of S0 Stream	R/W
49	S1 Done	End of S1 Stream	R/W
48	S2 Done	End of S2 Stream	R/W

Table 87 shows an example of how 10/100 Mbps frame data is split up in the pattern data structure. Eight streams are compared simultaneously with three 64-bit rows comprising one WOL entry. Rows 0–2 compare frame data0 against eight rules. Rows 3–5 compare frame data1 against the next eight rules. Rows 6–9 compare data2 against the final eight rules. The eight rules may be uniquely defined for all three WOL entries.

Table 87: Example of Splitting 10/100 Mbps Frame Data in Pattern Data Structure

Data[63:48]	Data[47:32]	Data[31:16]	Data[15:0]
Control Bits	Stream 0 data 0	Stream 1 data 0	Stream 2 data 0
Control Bits	Stream 3 data 0	Stream 4 data 0	Stream 5 data 0
Control Bits	Stream 6 data 0	Stream 7 data 0	Stream 8 data 0
Control Bits	Stream 0 data 1	Stream 1 data 1	Stream 2 data 1
Control Bits	Stream 3 data 1	Stream 4 data 1	Stream 5 data 1
Control Bits	Stream 6 data 1	Stream 7 data 1	Stream 8 data 1
Control Bits	Stream 0 data 2	Stream 1 data 2	Stream 2 data 2
Control Bits	Stream 3 data 2	Stream 4 data 2	Stream 5 data 2
Control Bits	Stream 6 data 2	Stream 7 data 2	Stream 8 data 2

Firmware Mailbox

When the Ethernet controller initializes (the firmware boot code is loaded from NVRAM when the chip powers on or when reset completes), the boot code checks the T3_FIRMWARE_MAILBOX in shared memory. When the T3_MAGIC_NUM signature (0x4B657654) is present, the boot code does not issue a hard reset to the PHY. This is especially important in WOL mode since the PHY should not be reset.

Before the host software issues a reset to the Ethernet controller, it must write the T3_MAGIC_NUM to the shared memory address T3_FIRMWARE_MAILBOX (0xb50). This address is a software mailbox, which boot code polls before it resets the PHY. The boot code will acknowledge the signature by writing the one's complement of the T3_MAGIC_NUM back into the T3_FIRMWARE_MAILBOX. If the T3_MAGIC_NUM is present, the boot code will not reset the PHY. After resetting the Ethernet controller, host software should poll for the one's complement of the T3_MAGIC_NUM before it proceeds, otherwise, boot code initialization may interfere with the host software initialization.

If the host software will be controlling the WOL configuration, it should write the DRV_WOL_SIGNATURE (0x474c0000) to the shared memory address DRV_WOL_MAILBOX (0xd30) so that the boot code will not take over the WOL initialization. If the DRV_WOL_SIGNATURE is not present, and WOL has been enabled, the boot code will assume that the host software is a legacy driver and skip the WOL initialization. If WOL is disabled, the boot code will take over the WOL initialization based on the NVRAM configuration.

Table 88: Firmware Mailbox Initialization

Name	Address	Recommended Value
T3_FIRMWARE_MAILBOX	0x0B50	0x4B657654
DRV_WOL_MAILBOX	0xd30	0x474c0000

PHY Auto-Negotiation

The integrated PHY should be configured to auto-negotiate for a 10 Mbps connection (see Table 89). This step is required if the NIC must be placed into a D3 cold state. Half- or full-duplex operation is acceptable. Software must modify auto-advertise configurations in the PHY's MDI registers. The link partner will read advertisement settings to find a highest common capability. Since WOL requires 10 Mbps wire speed, the two PHYs will effectively auto-negotiate for half- or full-duplex connection.

Table 89: Recommended Settings for PHY Auto-Negotiation

Register	Bit	Recommended Value
Auto_Negotation_Advertisement	10_BASE_TX_Half_Duplex	Enable
Auto_Negotation_Advertisement	10_BASE_TX_Full_Duplex	Enable
Auto_Negotation_Advertisement	100_BASE_TX_Half_Duplex	Disable
Auto_Negotation_Advertisement	100_BASE_TX_Full_Duplex	Disable
1000BASE-T_Control	1000_BASE_TX_Half_Duplex	Disable
1000BASE-T_Control	1000_BASE_TX_Full_Duplex	Disable

Power Management

The clocking inputs need to be modified for WOL mode (see Table 90). The RX CPU is not required during WOL operation, so its clock can be disabled. The MAC has an internal phase-locked loop that clocks internal logic at 133 MHz. Software must select an alternate clocking source and then disable this PLL.

Table 90: WOL Mode Clock Inputs

Register	Bit	Recommended Value
PCI Clock_Control	RX RISC_Clock_Disable	Set the bit to 1
PCI Clock_Control	Select_Alternate_Clock	Set the bit to 1
PCI Clock_Control	PLL133	Set the bit to 1

The settings shown in Table 91 enable Magic Packet detection logic in the MAC. These setting also enable the MAC to assert PME on the PCI bus. The RX MAC should maintain the multicast and broadcast settings that were previously configured by the NOS. The Microsoft® power management specification states:

"Only a frame that passes the device's MAC, broadcast, or multicast address filter and matches on the previously loaded sample patterns will cause the wake-up signal to be asserted."

The ACPI_Power-on bit needs to be set for pattern match, but not for Magic Packet recognition. The Magic Packet detection mechanism is separate from the pattern match mechanism. Host software may configure WOL using four filter permutations:

- Pattern match WOL disabled. Magic Packet disabled.
- Pattern match WOL enabled. Magic Packet disabled.
- Pattern match WOL disabled. Magic Packet enabled.
- Pattern match WOL enabled. Magic Packet enabled.

Table 91: Magic Packet Detection Logic Enable

Register	Bit(s)	Recommended Value
PCI Power_Management_Control/Status	PME_Enable	Enable
PCI Power_Management_Control/Status	Power_State	0x03
Ethernet_MAC_Mode	ACPI_Power-On	See above
Ethernet_MAC_Mode	Magic_Packet_Detection	See above

Integrated MACs

Table 92 lists the WOL mode control registers in the Ethernet controllers.

Table 92: Integrated MAC WOL Mode Control Registers

Register	Bit(s) Name	Description	Cross Reference
WOL_Pattern_Pointer	All	This register points to an internal memory location. Programmers should calculate pointer value by dividing a base address by 8.	"WOL Pattern Pointer Register (offset: 0x430)" on page 326
WOL_Pattern_Configuration	Length	The number of memory arbiter clock cycles needed to read X bytes in the RX stream/frame.	"WOL Pattern Configuration Register (offset: 0x434)" on page 327
	Offset	The number of bytes into the RX stream/frame to begin the pattern comparison.	

Table 92: Integrated MAC WOL Mode Control Registers (Cont.)

Register	Bit(s) Name	Description	Cross Reference
Ethernet_MAC_Mode	Port_Mode	This bit field specifies the type of interface the Ethernet controller port is currently using: MII, GMII, or none.	"EMAC Mode Register (offset: 0x400)" on page 321.
	Magic_Packet_Detection	Enable WOL pattern filtering.	
	Promiscuous_mode	All frames are forwarded, without any filtering, when this bit is enabled.	
PCI Clock_Control	RX RISC_Clock_Disable	Disable the clock to the receive CPU.	"Clock Control Register (offset: 0x688C)" on page 536
	Alternate_Clock_Source	Use an alternate clock as a reference, rather than the PLL 133.	
	PLL133	Disable the 133 MHz phase-locked loop.	
Misc Local Control	Misc_Pin_0_Output	GPIO pin 0.	"Miscellaneous Local Control Register (offset: 0x6808)" on page 524.
	Misc_Pin_0_Output_Enable	When asserted, MAC drives pin output.	
	Misc_Pin_1_Output	GPIO pin 1.	
	Misc_Pin_1_Output_Enable	When asserted, MAC drives pin output.	
	Misc_Pin_2_Output	GPIO pin 2.	
	Misc_Pin_2_Output_Enable	When asserted, MAC drives pin output.	
Power Management Control/Status	PME_Enable	Enable the Ethernet controller to assert PME on PCI bus.	"Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253.
	Power_State	Set the ACPI power state: D0, D3.	

WOL Data Flow Diagram

The Ethernet controller and PHY are both configured for WOL mode. The process is as follows:

1. Clear the PME_Status bit in the "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253. This bit must be cleared, so the PME interrupt is not immediately generated once the NIC is moved to the D3 state. The bit could be asserted from a previous D3–D0 transition.
2. Set the Mask_PCI_Interrupt_Output bit in the Miscellaneous_Host_Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). This bit should be set, so the Ethernet controller does not generate interrupts during the WOL configuration of the PHY. The device driver's ISR may attempt to reset and reconfigure the PHY as part of an error recovery code path.

3. If host software must place the NIC into D3 cold state, the following step is necessary. Set the 10_Base_TX_Half_Duplex and 10_BASE_TX_Full_Duplex Capability bits, in the Auto-Negotiation Advertisement Register. Clear the 100_BASE_TX_Full_Half_Duplex and 100_BASE_TX_Full_Duplex Capability bits, in the Auto-Negotiation Advertisement Register. Clear the 1000_BASE_TX_Half_Duplex and 1000_BASE_TX_Full_Duplex Capability bits, in the 1000BASE-T Control Register. The link partner will now only be able to auto-negotiate for 10 Mbps speed full/half-duplex.
4. Set the Restart_Auto_Negotiation bit in the MII Control Register. The integrated PHY and link partner will now reconfigure for 10 Mbps wire speed. Essentially, 10 Mbps link must be auto-negotiated or forced.
5. Disable the FHDE, RDE, TDE bits of the "EMAC Mode Register (offset: 0x400)" on page 321", and on-chip RISCs.
6. Host software must write the signature 0x4B657654 to internal memory address 0x0B50. Check for one's complement of 0x4B657654.
7. Enable the Wake_On_LAN bit in the AUXILIARY Control Register.
8. For Interesting Packet WOL Only: Set up the Interesting Packet pattern in Ethernet controller local memory.
9. For Interesting Packet WOL Only: Write a pointer value to the "WOL Pattern Pointer Register (offset: 0x430)" on page 326. This register uses a normalized pointer value, not a device base address. The value written to this register is BCM5700_BASE_ADDR/8. The base address must be a specific location in local memory: 0x8000, 0xC000, or 0xD000. The choice of memory location depends upon other MAC configurations, and the selection is not arbitrary.
10. For Interesting Packet WOL Only: Write the Offset field in the "WOL Pattern Configuration Register (offset: 0x434)" on page 327. The WOL pattern checker will position into received frames on two-byte intervals. The pattern checker compares two bytes in parallel, so host software should program the offset field accordingly. Host software may perceive this unit as OFFSET_BYTE/2 units.
11. For Interesting Packet WOL Only: Write the Length field in the "WOL Pattern Configuration Register (offset: 0x434)" on page 327. The length value is specified in terms of Memory Arbiter clock cycles, not bytes/words/dwords. A comprehensive discussion of how the clock cycles are calculated will be presented.
12. Set the Port_Mode field in the "EMAC Mode Register (offset: 0x400)" on page 321 to GMII mode. These bits enable the GMII between the MAC and internal PHY.
13. For Interesting Packet WOL Only: Enable the ACPI_Power-On bit in the "EMAC Mode Register (offset: 0x400)" on page 321. This bit will enable logic for D3 hot/cold transitions to D0 ACPI state. The MAC will also be capable of asserting PME on the PCI bus.
14. For Interesting Packet WOL Only: Enable the Magic_Packet_Detection bit in the "EMAC Mode Register (offset: 0x400)" on page 321. The WOL logic will compare RX frames for Magic Packet patterns.
15. Set the RX RISC_Clock_Disable bit in the PCI Clock_Control register (see "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253). The receive CPU will be stopped, and the clocking circuitry disabled.
16. Set the Enable_Alternate_Clock bit in the PCI Clock_Control register (see "Clock Control Register (offset: 0x688C)" on page 536). The Ethernet controller's 133 MHz Phase Locked Loop (PLL) no longer clocks internal logic and an alternate clock reference is used. Set the PLL LowPowerClock bit while keeping the Enable_Alternate_Clock bit set. Wait at least 27 µs and then clear the Enable_Alternate_Clock bit. The Ethernet controller's PLL is then switched to its lower power consumption mode.
17. In NIC applications, switch from VMAIN to VAUX in order to prevent a GRC reset. Set the required GPIOs of Ethernet controller if any of them are used for switching the power from VMAIN to VAUX.

18. Enable the RX MAC by setting the Enable bit of "Receive MAC Mode Register (offset: 0x468)" on page 332" and put it in promiscuous mode by setting the Promiscuous Mode bit of "Receive MAC Mode Register (offset: 0x468)" on page 332.

19. Enable the PME bit in the PCI "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253. The Ethernet controller asserts PME to wake up the system. Set the Power_State bits to D3 in the "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 253.

Flow Control

Description

The Ethernet controller supports IEEE 802.3x flow control. Flow control is a switched Ethernet capability, where link partners may pause traffic. The 802.3x flow control specifies that a MAC sublayer may transmit pause frames. The pause frames instruct the MAC's link partner to wait a specified amount of time, before sending additional frames. This delay provides the MAC time to free packet buffers. Conversely, the MAC sublayer must also accept/receive pause frames. Flow control is used by switches and bridges to prevent clients of dissimilar speeds from exhausting switching packet buffers. Clients and servers may use flow control for similar reasons. A very important requirement is that both link partners must share a full-duplex connection for flow control to be enabled. IEEE 802.3x flow control does not operate on a half-duplex connection. More information on flow control can be found in Appendix A: "Flow Control," on page 676.

The following architectural blocks are integral to flow control:

- Transmit MAC
- Receive MAC
- Statistics Block
- PHY Auto-negotiation
- PHY Auto-Advertise

Operational Characteristics

The Ethernet controller implements pause functionality using Xon and Xoff states. The MAC will extract a pause quantum from a pause control frame. Then, the MAC will configure its internal timer with the pause_time specified by the link partner. Frames that are currently in the transmit engine will be completed before the transmit engine is inhibited. The MAC has moved flow control into a Xoff state once the transmit engine is inhibited. Note that the transmit engine is not completely disabled since the IEEE 802.3 specification stipulates that MAC control frames should not be paused.

One of the following conditions moves the Ethernet controller into an Xon state:

- Link partner sends a pause frame with pause_time = 0.
- Internal pause timer expires.

Transmit MAC

The transmit MAC is responsible for sending flow control frames. Software enables the transmit MAC to send flow control frames by setting the Enable_Flow_Control bit in the Transmit_MAC_Mode register (see "Transmit MAC Mode Register (offset: 0x45C)" on page 330). When software clears the Enable_Flow_Control bit, the transmit MAC will not generate flow control frames. The MAC_RX_MBUF_Low_Water_Mark register value triggers PAUSE frames to be transmitted when a threshold value is passed. Software may alter the watermark to tune system performance.

Table 93: Transmit MAC Watermark Recommendation

Register	*Recommended Value*
MAC_RX_MBUF_Low_Water_Mark	24

As soon as PAUSE frame is transmitted, any incoming packet can be dropped, and the ifInDiscard counter in statistics will increase. When packet size is small (64 bytes) with 1000 Mbps, more frames can be dropped. Even if the PAUSE frame is transmitted, Pause frames cannot inhibit MAC control frames.

Low Water Mark Maximum Receive Frames register (see "Low Watermark Maximum Receive Frame Register (offset: 0x504)" on page 337) control the number of good frames to receive after the RX MBUF Low Water Mark has been reached. After the RX MAC receives this number of frames, it will drop subsequent incoming frames until the MBUF High Water Mark is reached.

The IEEE 802.3 pause control frame contains a pause_time field. The Ethernet controller inserts a time quanta into the pause_time field. Software should set the Enable_Long_Pause bit in the Transmit_MAC_Mode register to configure long pause quanta. Clearing the Enable_Long_Pause bit will default the pause_time back to the shorter quanta. Table 94 shows the pause quanta based on the Enable_Long_Pause bit setting.

Table 94: Pause Quanta

Enable_Long_Pause Bit	*Pause_Time*
DISABLED (0)	0x1FFF
ENABLED (1)	0xFFFF

Receive MAC

The Ethernet controller receive MAC's link partner may want to inhibit frame transmission until upstream resources become available. The receive MAC must be configured to accept IEEE 802.3x pause frames (see Table 95). Software should set the Enable_Flow_Control bit in the Receive_MAC_Mode_Control register to enable automatic processing of flow control frames. If software clears the Enable_Flow_Control bit, IEEE 802.3x pause frames will be discarded. The Keep_Pause bit in the Receive_MAC_Mode_Control register will instruct the RX engine to forward pause frames to host memory. Software may be interested in setting this bit for debugging or promiscuous/sniffer configurations. Passing pause frames to the host will increase DMA and protocol processing and consume available host buffers. The receive MAC will filter pause control frames when the Keep_Pause bit is disabled.

Table 95: Keep_Pause Recommended Value

Register.Bit	*Recommended Value*
Receive_MAC_Mode_Control.Keep_Pause	DISABLED

Statistics Block

The statistic block shown in Table 96 is a common data structure. The relationships of flow control statistics are discussed in this section. Xon/Xoff statistical counters are related to internal Ethernet controller flow control states. Xon is associated to transmit enabled state and Xoff is associated to transmit disabled state. These Xon/ Xoff states are not part of the IEEE 802.3 specification; the Ethernet controller uses Xon/Xoff to manage flow control state and transitions. The Xon/Xoff statistics provide programmers with a high level of granularity for the measurement of Ethernet controller flow control performance in a LAN (see Appendix A: "Flow Control," on page 676).

Table 96: Statistic Block

Statistic	*Description*
xoffStateEntered	This counter is bumped under the following conditions: • IEEE 802.3 MAC flow control pause frame received with valid CRC. • (Pause_time > 0) The link partner requests transmission inhibit. The counter increments independently of the enabled/disabled state of Receive_MAC_Mode_Control.Flow_Enabled.
xonPauseFramesReceived	This counter is incremented under the following conditions: • IEEE 802.3 MAC flow control pause frame received with valid CRC. • (Pause_time == 0) The link partner no longer requires the device family to pause/wait/delay outgoing packets. The counter increments independently of the enabled/disabled state of Receive_MAC_Mode_Control.Flow_Enabled.
xoffPauseFramesReceived	This counter is incremented under the following conditions: • IEEE 802.3 MAC flow control pause frame received with valid CRC. • (Pause_time > 0) The link partner requires the BCM5725/BCM5762/ BCM57767 to pause/wait/delay outgoing packets. The counter increments independently of the enabled/disabled state of Receive_MAC_Mode_Control.Flow_Enabled.
outXon	This counter is incremented under the following conditions: • Transmit_MAC_Mode_Control.Flow_Enabled bit is set. • (MAC_RX_MBUF_Low_Water_Mark > Threshold Value MAC resources are available. • (pause_time == 0) 802.3 MAC flow control frame is sent.
outXoff	This counter is incremented under the following conditions: • Transmit_MAC_Mode_Control.Flow_Enabled bit is set. • (MAC_RX_MBUF_Low_Water_Mark < Threshold Value) MAC resources are running low and a pause is desired. • (pause_time > 0) IEEE 802.3 MAC flow control frame is sent.

PHY Auto-Negotiation

The PHY encodes flow control capability into Fast Link Pulse (FLPs) bursts. Link partners will extract encoded flow control capability from FLPs and then create a Link Code Word (LCW). The LCW is a message, which contains a selector and technology ability field. The technology ability field contains a bit called Pause_Operation_for Full_Duplex_Link (A5). Refer to Annex 28-B of the IEEE 802.3 specifications. The A5 bit signifies that a link partner has implemented pause functionality. If both link partners support auto-negotiation, they will further exchange data regarding flow control, using the next page bit in the LCW.

Auto-advertise is integrally tied to auto-negotiation. If link partner does not support pause functionality, the PHY Auto_Negotation_Link_Partner_Ability_Register does not set the Pause_Capable bit. The Ethernet controller should not send pause frames to this link partner since flow control is not implemented or disabled. The Ethernet controller can still accept pause frames, but sending a pause frame does not yield a preferred result.

Integrated MACs

Table 97 lists the flow control registers in the Ethernet controllers.

Table 97: Integrated MAC Flow Control Registers

Register	*Bit(s) Name*	*Description*	*Cross Reference*
Receive MAC Mode	Enable_Flow_Control	Enable automatic processing of IEEE 802.3 flow control frames.	See "Receive MAC Mode Register (offset: 0x468)" on page 332.
Transmit MAC Mode	Enable_Flow_Control	Enable automatic processing of IEEE 802.3 flow control frames.	See "Transmit MAC Mode Register (offset: 0x45C)" on page 330.
MAC_RX_MBUF_Low_Water_Mark	All 32 bits	The number of internal buffers that must be available before the RX engine can accept a frame from the wire. Threshold value for initiating flow control.	See "Low Watermark Maximum Receive Frame Register (offset: 0x504)" on page 337.

Flow Control Initialization Pseudocode

```
//Check the Link State
If (MII_Status_Reg.Link_Status == TRUE) Then
{
    //Check PHY status register for full-duplex configuration
    If (MII_Aux_Status_Reg.Auto_Neg_HCD ==
        (1000_FULL_DUPLEX Or 100_FULL_DUPLEX Or 10_FULL_DUPLEX) ) Then
    {
            //Check if USER has forced either auto-negotiation or auto-advertise
            If ( (Driver_Auto_Neg_Variable == ENABLED) And
                (Driver_Auto_Advertise_Variable != FORCED_SPEED_DUPLEX )) Then
            {
                // Probe Phy control registers for advertised flow control info
                // Expected abilities should match the configured abilities. Expected abilities
                // are based on the IEEE 803.3ab flow control subsection.
                If  ( (Auto_Neg_Advertise_Reg.Asymetric_Pause != 802.3ab_Table_28B-3 ) And
                    (Auto_Neg_Advertise_Reg.Pause_Capable != 802.3ab_Table_28B-3 ) ) Then
                {
                    //The current advertised state does not match 802.3 specifications
                    Driver_ Link__link_state = LINK_STATUS_DOWN
                }
                Else
                {
                    If  (Auto_Neg_Advertise_Reg.Pause_Capable == ENABLED)
                    {
                        If ( Auto_Neg_Advertise_Reg.Asymetric_Pause == ENABLED) ) Then
                        {
                            If  (Auto_Neg_Link_Partner_Ability_Reg.Pause_Capable == ENABLED) Then
                            {
                                Driver_Flow_Capability = FLOW_CONTROL_TRANSMIT_PAUSE \
                                    |  FLOW_CONTROL_RECEIVE_PAUSE
                            }
                            Else If (Auto_Neg_Link_Partner_Ability_Reg.Asymetric_Pause == \
                                ENABLED) Then
                            {
                                Driver_Flow_Capability = FLOW_CONTROL_RECEIVE_PAUSE
                            }
                            Else
                            {
                                Driver_Flow_Capability = NONE
                            }
                        }
                        //The local physical layer was not configured to advertise Asymmetric pause
                        Else
                        {
                            If  (Auto_Neg_Link_Partner_Ability_Reg.Pause_Capable == ENABLED) Then
                            {
                                Driver_Flow_Capability = FLOW_CONTROL_TRANSMIT_PAUSE \
                                    | FLOW_CONTROL_RECEIVE_PAUSE
                            }
                            Else
                            {
                                Driver_Flow_Capability = NONE
                            }
                        }
                    }
                    // The local physical layer was not configured to advertise Pause capability
                    Else If  (Auto_Neg_Advertise_Reg.Asymetric_Pause == ENABLED) Then
                    {
                        If  (Auto_Neg_Link_Partner_Ability_Reg.Pause_Capable == ENABLED) Then
                        {
                            Driver_Flow_Capability = FLOW_CONTROL_TRANSMIT_PAUSE
                        }
                        Else
                        {
                            Driver_Flow_Capability = NONE
                        }
                    }
            } //Link Status is up
        } // Auto negotiation was not disabled && Speed Duplex was not forced
        Else
        {
            // The use forced speed/duplex, so the partner's flow control capabilities are
            // indeterminate - software cannot use the Link_Partner_Abitity
            // registers.
            Driver_Flow_Capability= DISABLED
        }
    } //The current link is full-duplex at 10/100/1000 wire speeds
    Else
    {
            //Full-Duplex mode is not available or forced half-duplex
            //Flow control is not available in half-duplex mode.
            Driver_Flow_Capability = NONE
    }
    //Configure MAC Flow Control Registers
    if ( Driver_Flow_Capability & FLOW_CONTROL_RECEIVE_PAUSE )
    {
        Receive_MAC_Mode_Control_Register.Enable_Flow_Control = ENABLED
    }
    if ( Driver_Flow_Capability & FLOW_CONTROL_TRANSMIT_PAUSE ) Then
    {
        Transmit_MAC_Mode_Control_Register.Enable_Flow_Control = ENABLED
    }
}  // Link is up on the local PHY
```

Section 10: Interrupt Processing

Host Coalescing

Interrupt coalescing (or interrupt moderation) is a common technique used by NIC vendors to increase the performance of NICs. High-level descriptions of the benefits of interrupt coalescing can be found at:

- http://www.microsoft.com/HWDEV/devdes/optinic.htm
- http://support.microsoft.com/support/kb/articles/Q170/6/43.ASP
- http://msdn.microsoft.com/library/books/serverdg/networkadapterrequirements.htm

Description

The Ethernet controller supports the concept of host coalescing. Host coalescing controls when status information is returned to the host, and when interrupts are generated. The Ethernet controller provides a number of SW configurable registers that control when/how it updates the host with status information and how often it asserts an interrupt.

When the Ethernet controller has completed transmit or receive events, it updates a Status block in host memory. This status block contains information that tells the host which transmit buffers have been DMAed by the NIC, and which receive Buffer Descriptors (BDs) have been consumed by a newly arrived received packet. Normally, the host will check this status block when an interrupt is generated. In addition, the host could also poll the status block to determine whether or not it had been updated by the hardware since the last time the host had read the status block (this is called during interrupt processing).

When the NIC updates the status block, it will make a decision about whether to assert the interrupt line ($\overline{\text{INTA}}$) or not. The Ethernet controller has special interrupt avoidance mechanisms that allow the host to tell the NIC not to generate an interrupt when it writes a status block back to the host. In addition, there are also mechanisms that allow host SW to control when and how often the status block is updated.

> ***Example:*** The host could configure the NIC to only update status block after it receives two packets, as opposed to one packet. These mechanisms are documented in more detail to follow.

Operational Characteristics

The Ethernet controller DMAs the status block to host memory before a line interrupt or MSI is generated. The host ISR reads the update bit at the top of the status block and checks whether this bit is set to 1 or not. When set to 1, the updated bit of status block indicates the host that the status block has been refreshed by the MAC. The ISR must then write to clear/de-assert this bit to dirty the status block, and then the ISR may proceed to read the updated producer/consumer index pointers. This mechanism allows host system software to determine if the status block has been updated. Due to various asynchronous timing issues (dependent upon platform) the ISR may occasionally see stale data. The ISR may either spin and wait for the status block DMA to complete and explicitly flush the status block or just wait for the next line interrupt.

Registers

The Ethernet controller supports a variety of registers that affect status block updates and interrupt generation (see Table 98).

Table 98: Interrupt-Related Registers

Register	***Cross Reference***
Miscellaneous Host Control register. The two bits of this register that are related to interrupts are: • Mask PCI Interrupt Output (aka Mask Interrupt) bit • Clear Interrupt $\overline{\text{INTA}}$ bit	"Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256.
Miscellaneous Local Control register. The two bits of this register that are related to interrupts are: • Set Interrupt bit • Clear Interrupt bit	"Miscellaneous Local Control Register (offset: 0x6808)" on page 524.
Interrupt Mailbox 0 register	"Interrupt Mailbox 0 (High Priority Mailbox) Register (offset: 0x200-207)" on page 319 for host standard and "Interrupt Mailbox 0 Register (offset: 0x5800)" on page 513 for indirect mode.
Receive Coalescing Ticks register	"Receive Coalescing Ticks Register (offset: 0x3C08)" on page 444.
Send Coalescing Ticks register	"Send Coalescing Ticks Register (offset: 0x3C0C)" on page 445.
Receive Max Coalesced BD Count register	"Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446.
Send Max Coalesced BD Count register	"Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 446.

MSI

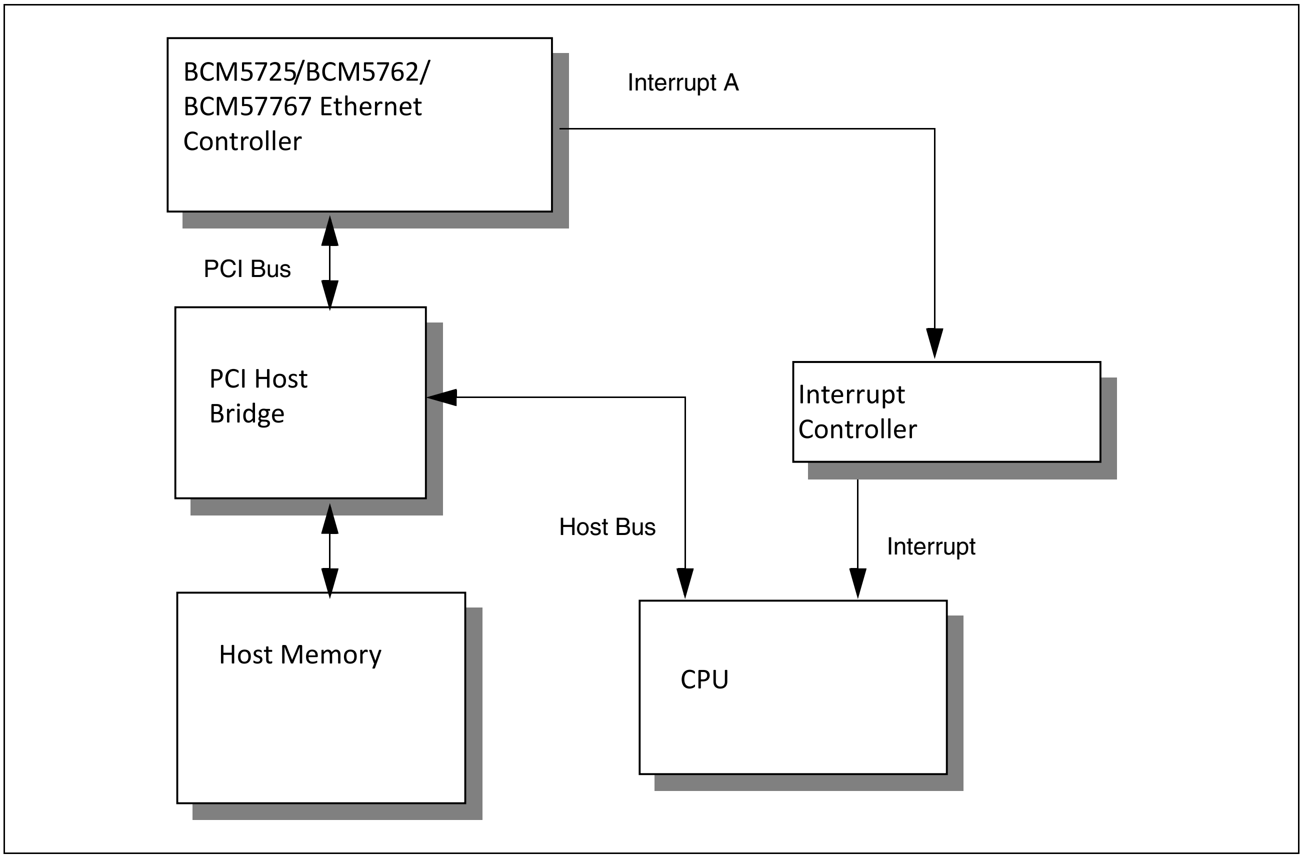
PCI Specification 2.2 defines a new mechanism for a device to request services by its device driver. It is called Message Signaled Interrupt (MSI). MSI will eventually deprecate the traditional interrupt mechanism. In MSI, device DMAs a specified DWORD data to a specified host address if it needs to request services by its device driver. The MSI state machine can be enabled/disabled by setting/resetting the Enable bit of MSI Mode register (offset 0x6000). By default, this bit is set to 1 indicating that the MSI state machine is enabled. The main advantages of MSI generation versus using a traditional interrupt are as follows:

- Eliminates the need for interrupt signal trace on the PCI device.
- Eliminates the need to perform a dummy read from the device by the device driver in its interrupt service routine. The dummy read is done at the beginning of ISR to force all posted memory writes to be flushed to the host memory.

Traditional Interrupt Scheme

A simplified block diagram showing traditional interrupt scheme is depicted in Figure 56.

Figure 56: Traditional Interrupt Scheme



To clarify second issue in traditional interrupt scheme, an example is given. The Ethernet controller receives one or more packets from the networks. The Ethernet controller does the following:

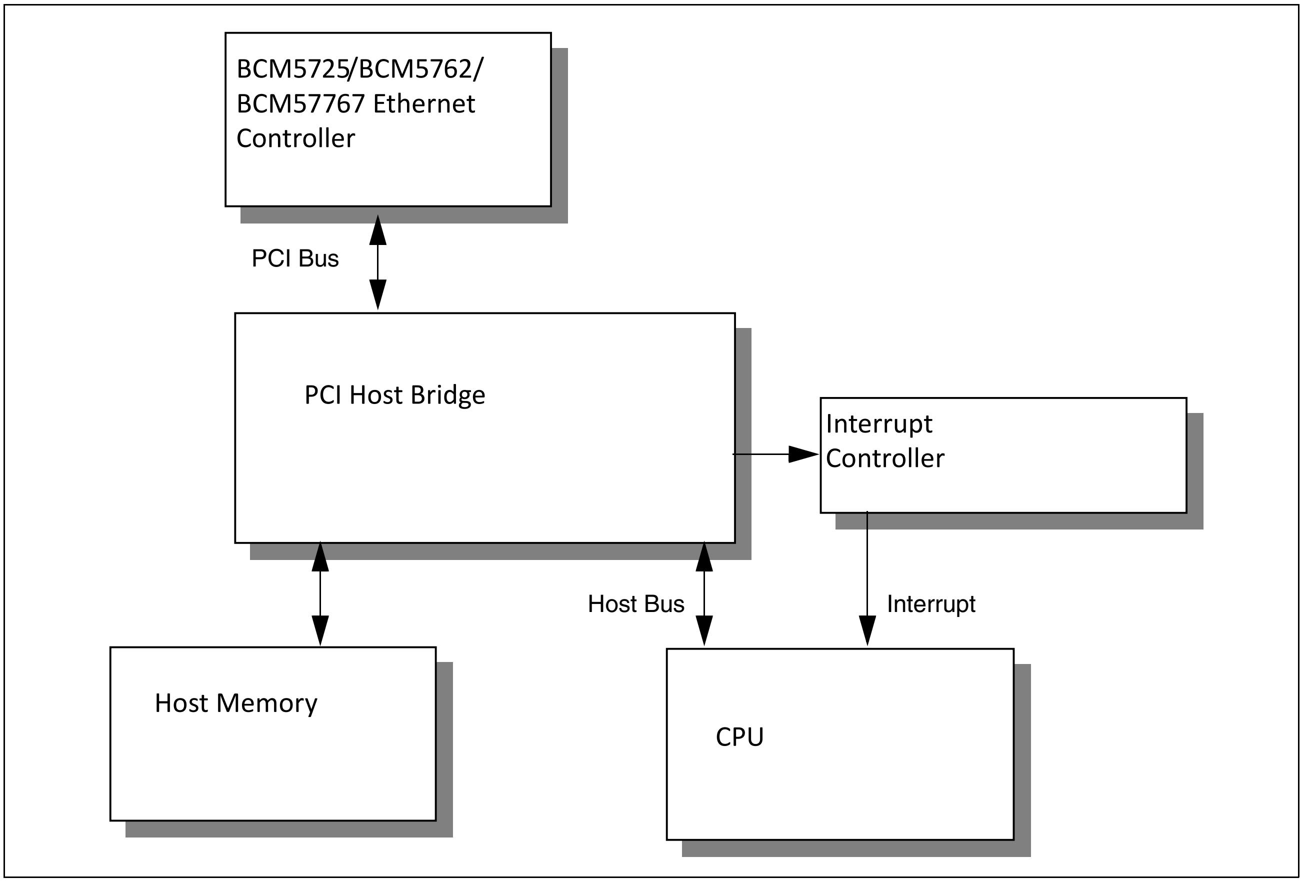
- DMAs data of received packets to the host.
- DMAs receive buffer descriptors to Receive Return Ring in the host memory.
- DMAs status block to the host memory.
- Generates an interrupt to request its device driver for processing.

The writes are posted and are actually performed at some later time by the PCI host bridge. When interrupt service routine of device driver is executed, the driver reads the status block from the host memory and finds that status block does not contain latest index information if the writes for status block are not performed by the PCI host bridge yet. The scheme to resolve this problem is to do a dummy read of the Ethernet controller in the beginning of the interrupt service routine. The dummy read has to traverse the same bridge that memory writes from the Ethernet controller have to traverse to get to the host memory. The ordering rules for bridges dictate that the bridge must flush its posted write buffers before permitting a read to traverse the bridge. As a result, writes for status block are flushed to the host memory by the bridge before dummy read cycle is completed.

Message Signaled Interrupt

A simplified block diagram showing a possible MSI scheme is depicted in Figure 57.

Figure 57: Message-Signaled Interrupt Scheme



Similar example in traditional interrupt scheme is used again here to illustrate MSI concept. The Ethernet controller receives one or more packets from the networks. The Ethernet controller does the following:

- DMAs data of received packets to the host.
- DMAs receive buffer descriptors to receive return ring in the host memory.
- DMAs status block to the host memory.
- Writes specified DWORD data to specified host address.

In this mode, the Ethernet controller writes DWORD data to specified host address instead of generating an interrupt. The specified data and address are configurable. The specified address is typically a memory-mapped IO port within the PCI host bridge. The PCI host bridge is the gateway to the main memory controller. This means that the DWORD data write (MSI message) to PCI host bridge is in the posted write buffers and was posted after the writes for the status block update. It is the rule that PCI host bridge must perform posted writes in the same order that they were received. This means that by the time MSI message arrives at the PCI host bridge, the status block has already been posted to the host memory. Upon receipt of the MSI message write, the PCI host bridge generates the interrupt request to the processor. Interrupt service routine of the device driver is invoked. It is not necessary to do a dummy read because updated status block is already in the host memory.

PCI Configuration Registers

Operating system/system software can configure the specified DWORD data and specified 64-bit host address for the device with MSI_DATA (Offset 0x64) and MSI_Address register (Offset 0x5c), respectively.

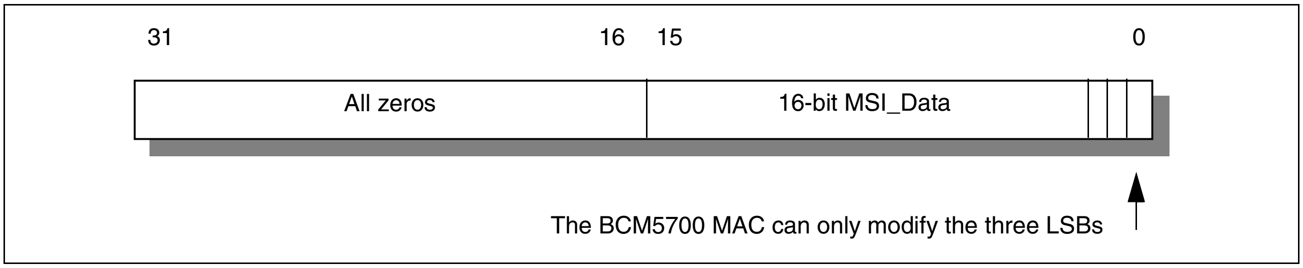
MSI Address

This is a 64-bit field. MSI address at offset 0x5c and 0x60 should be programmed with the low-order and high-order bits of the 64-bit physical address. If the host only supports 32-bit physical address, the high-order address should be programmed with zeros.

MSI Data

This is a 16-bit field. The least significant three bits can be modified by the Ethernet controller when it writes MSI message to host. The DWORD data for the MSI message is depicted as shown in Figure 58.

Figure 58: MSI Data FIeld



The Ethernet controller can support up to eight message types, and these MSI messages can be generated by either of the two sources of:

- Host coalescing engine
- Firmware

Host Coalescing Engine

After the host coalescing engine updates the status block on the host (due to receive indication, transmit completion, and so on), it either generates an interrupt or writes a MSI message if MSI is enabled. The least significant 3-bits of the MSI message originating from host coalescing block is configurable and can be configured by programming bits 4, 5, and 6 of the Host_Coalesing_Mode register. The default of these bits is zeros.

Firmware

The Ethernet controller provides a way for firmware executed by RX RISC to generate MSI messages. Firmware can generate MSI messages by using MSI_FIFO_Access register (Offset 0x6008). For example, if firmware wants to generate an MSI message with least significant 3-bit as 0x2, it will write 2 to MSI_FIFO_Access register. It also needs to verify that the MSI message is written successfully by reading back MSI_FIFO_Access Overflow. If this bit is zero, then the MSI message is encoded successfully and will be sent to HOST. Otherwise, the message is not encoded.

Note: Without any special firmware supporting multiple MSIs, the device can generate only 1 MSI message even though the device requests for 8 MSI messages through Multiple Message Capable field (bits 3:1) of Message Control register (offset 0x5A). The least significant 3-bits of the MSI message generated by the device are always taken from bits 6:4 of Host_Coalesing_Mode register (offset 0x3C00).

MSI-X

This section discusses the BCM5725/BCM5762/BCM57767 MSI-X implementation.

Note: Refer to the status-block format for various status-block formats when using MSI-X. The exact format of the status block changes depending upon which combination of MSI-X, RSS, Ethernet Audio Video (EAV) is being used.

MSI-X Vectoring

The BCM5725/BCM5762/BCM57767 introduces PCIe compliant MSI-X capability with support for a maximum of 6 vectors. This is accomplished without disturbing the legacy interrupt system model.

The motivation behind introducing MSI-X interrupt vectoring is to provide per receive queue indication/per transmit queue completion to the host based device driver.

The maximum number of MSI-X vectors is determined by certain chip modes of operation. Multiple Receive Queues (RX Return Rings) in the BCM5725/BCM5762/BCM57767 stems from the Receive Side Scaling (RSS) feature or EAV feature.

Note: RSS and EAV are mutually exclusive. Only one of these two features may be enabled at any time.

When both RSS Mode and EAV Mode are disabled all of the received packets are posted in the default receive queue or RX Return Ring 0 (At times, referred to simply as the Receive Return Ring). The number of Transmit Queues is also limited to one in this mode of operation.

In RSS enabled Mode (EAV disabled), receive traffic classification (sorting) is done purely based on the RSS hash-lookup table. There are four RSS Receive Queues and a single Transmit queue in this mode of operation. Therefore up to 5 MSI-X Vectors shall be offered in RSS Mode.

In EAV Mode (RSS disabled), receive traffic classification is performed by EAV Filters. Sorted RX traffic is routed amongst 4 Receive Queues (RXQs). The transmit side offers 2 transmit queues (one for normal "best effort" traffic and one for isochronous EAV traffic). Therefore, while in EAV Mode, there is an opportunity to map packet indication and completions into 6 MSI-X vectors.

The MSI-X specification allows a device to advertise the availability of a chosen number of vectors in its PCIe Capability list. However, it does not specify a mechanism for an OS to negotiate the number of vectors down to the number it would actually want to use. One way for an OS to allocate a smaller number of vectors than a device has asked for is simply not to fill out all of the Vector Table Entries advertised by a device. In this scenario the device would first need to identify the situation and then it would need to reassign or regroup the internal interrupt sources into the limited number of Table Entries allocated by the OS. Unfortunately, the device hardware itself would not possess all of the information necessary to accomplish either of these tasks. Hence device driver involvement is necessary.

In the case of the BCM5725/BCM5762/BCM57767 controller it has been assumed that in most cases an OS would allocate the requested number of vectors. A deviation from that would be considered as an exception, in which case, since support for MSI-X has already been advertised to the BIOS, there would be no way to fall back to INTX mode. The way to handle this situation would be to force everything into MSI-X Vector#0 and not use any other vectors, disregarding how ever many vectors the OS actually allocated.

The BCM5725/BCM5762/BCM57767 offers two vector modes within MSI-X mode:

- Single-vector mode
- Multivector mode

Each of these two modes in turn offers two submodes:

- Single-vector mode (restrict to Vector#0)
  - Single-vector RSS mode
  - Single-vector EAV mode
- Multivector Mode
  - Multivector RSS mode (6 vectors requested, only 5 used)
  - Multivector EAV mode (6 vectors requested)

The idea is that during boot-up, if either RSS or EAV Mode is desired to be enabled, the driver would program the chip into the appropriate Multivector submode first. Subsequently, if the OS does not allocate a requested number of MSI-X vectors (i.e., 5 in this case), the driver shall reprogram the chip to the appropriate Single Vector submode. All of this must be performed before the driver enables the EMAC to receive or transmit traffic. Once traffic is started the MSI-X Vector Mode must not be reprogrammed. The controller will behave unpredictably if that is done.

The Single-Vector Mode or the Multivector Mode may be chosen by the driver by programming register bit 0x6000[7]. Such submodes are derived by the chip from appropriate RSS and EAV mode settings. All permissible combinations are shown in the table below.

Table 99: MSI-X Vector Mode Selection

MSI-X Vector Mode	*Multivector Enable 0x6000[7]*	*RSS Mode Enable 0x468[23]*	*EAV Mode Enable 0x6800[18]*
Single Vector — RSS mode	Yes	0x0	0x1
Single Vector — EAV mode	Yes	0x0	0x0
Multivector — RSS mode	Yes	0x1	0x1
Multivector — EAV mode	Yes	0x1	0x0
INTx signaling — EAV or RSS mode	No	X	X

Vector allocation in each mode is shown below.

- Non-RSS + Non-EAV (Legacy) mode
  - Unified Status Block
    - RX Return Ring Indication
    - RX Standard/Jumbo Producer Ring Indication
    - Send Ring Completion
    - Link Status Change
    - Error/Attention
- Single-Vector mode
  - RSS Mode: Vector#0 — aggregate of the following:
    - RX Return Ring0 through Ring3 Indications
    - RX Standard/Jumbo Producer Ring Indication
    - Send Ring Completion
    - Link Status Change
    - Error/Attention
  - EAV Mode: Vector#0 — aggregate of the following:
    - RX Return Ring0 through Ring3 Indications
    - RX Standard/Jumbo Producer Ring Indication
    - Send Ring#1 Completion (normal best-effort Send Ring)
    - Send Ring#2 Completion (isochronous Send Ring)
    - Link Status Change

    - Error/Attention

- Multivector mode
  - RSS (6 Vector) Mode
    - Vector#0 — aggregate of the following:
      - Send Ring Completion
      - RX Standard/Jumbo Producer Ring Indication
      - Link Status Change
      - Error/Attention
    - Vector#1 — RSS Return Ring 0 Indication
    - Vector#2 — RSS Return Ring 1 Indication
    - Vector#3 — RSS Return Ring 2 Indication
    - Vector#4 — RSS Return Ring 3 Indication
    - Vector#5 — Unused
  - EAV (6 Vector) Mode
    - Vector#0 — Aggregate of the following:
      - Send Ring#1 Completion
      - RX Standard/Jumbo Producer Ring Indication
      - Link Status Change
      - Error/Attention
    - Vector#1 — RXQ Return Ring 0 Indication
    - Vector#2 — RXQ Return Ring 1 Indication
    - Vector#3 — RXQ Return Ring 2 Indication
    - Vector#4 — RXQ Return Ring 3 Indication
    - Vector#5 — Send Ring#2 Completion

In case of Multivector selection, it is not mandatory to enable all 4 RSS queues or all RX queues. Therefore, in case of multivector RSS, vector#1 through vector#4 are only useful in conjunction with the respective number of RSS queues or CPU# enablement. This is to say that, as an example, if only three receive queues are receiving traffic and the Multivector mode is selected by driver, then even though vectors 0, 1, 2 and 3 remain active and vector#4 will never trigger. Hence, in Multivector RSS Mode, the receive queues are hard mapped one to one with MSI-X vectors 1 through 4. Thus it is not permissible to regroup receive queues into vectors arbitrarily.

This same policy applies to Multivector EAV mode as well.

Consequently, Multivector mode must only be selected in conjunction with enablement of either RSS or EAV. The converse is not enforced by the chip. That is, either RSS or EAV Modes may be enabled along with legacy INTx interrupt signaling.

The data structures used in INTx and MSI mode are identical to that of single-vector MSI-X.

In Single Vector Mode all of the receive traffic shall be indicated via Vector#0. In RSS Mode all 4 Receive Queue indications are grouped together in Vector#0. In EAV Mode all 4 RX queues and both TX queues are grouped into Vector#0. Also, this mode must be used when RSS and EAV are both disabled (but MSI-X is enabled), however, this is not seen as a useful case. There are instances when an OS may enable RSS but would allocate only one MSI-X vector to our device - in which case the driver must resort to choosing the Single-vector mode.

Note: The Legacy mode INTx or MSI continues to function without any change.

PCIe Mandated Data Structures

MSI-X Capability Structure

MSI-X requires that its device resident data structures or registers be addressed over BARs - as opposed to MSI which declares its device structure addresses directly in related configuration registers. There are certain restrictions related to these BARs:

- If a BAR is shared with MSI-X for other device purposes, the MSI-X region must be isolated within a 4KB naturally aligned region. Alternatively, the MSI-X structures may be placed in their own captive BARs.
- Broadcom has chosen to place MSI-X in it own set of BARs - BAR2 and BAR3 (64-bit)
- The LAN PCIe Function of the BCM5725/BCM5762/BCM57767 (Function 0) shall advertise availability of their BAR2 and BAR3 as MSI-X BARs — this is done via the MSI-X capability structure.

The MSI-X Capability structure points to two structures that must be implemented inside a device:

- MSI-X Table
- Pending Bit Array (PBA)

There is also a Message Control register.

The capability structure is shown in the table below.

Table 100: MSI-X Capability Structure

31 16	***15 8***	***7 3***	***2 0***
Message Control Reg	Next Pointer	Capability ID	
MSI-X Table Offset			Table BIR
PBA Offset			PBA BIR

The BIR bits shown above point to the BAR registers that a device function uses to base the respective data structures. In the case of BCM5725/BCM5762/BCM57767, as mentioned above, both BIRs shall have a hard wired value of 0x2, which implies BAR2 and BAR3.

Note: The PCI v2.3 spec states that "For all accesses to MSI-X Table and MSI-X PBA fields, software must use aligned full DWORD or aligned full QWORD transactions; otherwise, the result is undefined."

MSI-X Data Structures

The full specification of MSI-X is available in PCI Spec rev 2.3. This document only discusses the salient features that are important to a user of this device.

The MSI-X Table hosted by BCM5725/BCM5762/BCM57767 is described below.

- This structure shall be placed at offset 0x0 pointed to by BAR2 and BAR3. The content of the MSI-X Table structure is shown in the MSI-X Vector Table shown below.
- Depending on whether RSS or EAV Mode is selected or not, the PCIe core shall advertise a 6 entry MSI-X table. Such advertisement choice is made only once following POR and cannot be changed afterwards (see Table Size field description below). Selection of Single-Vector/Multivector mode does not affect Table Size.
- This table is comprised of multiple 4-DWORD-long entries. Each entry corresponds to one MSI-X vector. Therefore in the case of the BCM5725/BCM5762/BCM57767, there is a maximum of 6 such entries.
- Each Entry consists of 4 fields:
  - The Message Address High and Message Address Low fields point to a 64-bit Host Address where the corresponding vector message must be posted.
  - Message Data contains a 32-bit vector data. Every time the BCM5725/BCM5762/BCM57767 wants to send an interrupt message corresponding to this vector, it shall write the value provided by this field into the address pointed to by the Message Address field(s).
  - Host software is allowed to replicate the same physical host address into multiple entries. This will amount to interrupt vector aliasing. The BCM5725/BCM5762/BCM57767 is able to handle such aliasing.
  - Only one bit, bit[0] of the Vector Control field, is implemented by the PCI/PCIe. This is the (per vector) Mask bit. When this bit is 1, the device function must not send a corresponding interrupt vector message to the Host - instead, the function must set the corresponding Pending bit in the PBA. When this bit was 1 and a 0 is written to it, a device must schedule an interrupt vector in case one was already “pending.”
- The PBA Structure in the BCM5725/BCM5762/BCM57767 shall be only 4 DWORDs (128 bits) wide - out of which bits [5:0] are useful, while bits[127:6] are reserved for future use. Each PBA bit index corresponds to the respective MSI-X vector#, that is the reason why only 6 bits are implemented in BCM5725/BCM5762/BCM57767. The PBA shall be placed at the offset 0x120 relative to the addresses pointed to by BAR2 and BAR3. See MSI-X Vector Table shown below.
- Message Control Register - There are only three fields which are important to the respective MAC Core:
  - MSI-X Enable: This is the feature enable/disable bit. The MAC Core must dynamically snoop CFG writes to this bit in order to determine if MSI-X gets enabled or not. When MSI-X is enabled, Line Interrupt Message and MSI Message must be gated off by the MAC Core. (Although the PCIe specification allows host software to enable MSI and MSI-X concurrently, albeit erratically, the BCM5725/BCM5762/BCM57767 will preempt MSI in such a scenario)
  - Function Mask: This bit acts like a device function wide vector mask - when this bit is 1, all vectors in the function, that is, 0 through 16 in BCM5725/BCM5762/BCM57767 must be masked. If any interrupt vector event occurs while Function Mask = 1, the corresponding Pending bit in the PBA must be set. When this bit is 0, the per vector Mask bits found in each table entry determines whether a vector is masked or not.
  - Table Size: This field shall declare a value 6. This is accomplished by device boot code programming the appropriate private register of the PCIe core.

The table below depicts the implemented address regions of BAR2 and BAR3 in BCM5725/BCM5762/ BCM57767. There are only two structures present:

- MSI-X Table
- Pending Bit Array (PBA)

Any host accesses to the non-implemented addresses in these BARs will be gracefully handled by the chip (i.e., reads return 0x0 and writes have no effect).

Table 101: MSI-X Vector Table and PBA Structure

MSI-X Table Entry#	***DW3 Content (32-Bit)***	***DW2 Content (32-Bit)***	***DW1 Content (32-Bit)***	***DW0 Content (32-Bit)***	***BAR3 and BAR2 Offset***
N/A	Reserved Bits [127:6]	(PBA)	Pending Bits[5:0]		0x120
Reserved Entry	–	–	–	–	0x110
Reserved Entries	–	–	–	–	–
5	Vec#5 Control	Msg#5 Data	Msg#5 Addr H	Msg#5 Addr L	0x50
4	Vec#4 Control	Msg#4 Data	Msg#4 Addr H	Msg#4 Addr L	0x40
3	Vec#3 Control	Msg#3 Data	Msg#3 Addr H	Msg#3 Addr L	0x30
2	Vec#2 Control	Msg#2 Data	Msg#2 Addr H	Msg#2 Addr L	0x20
1	Vec#1 Control	Msg#1 Data	Msg#1 Addr H	Msg#1 Addr L	0x10
0	Vec#0 Control	Msg#0 Data	Msg#0 Addr H	Msg#0 Addr L	0x00

MSI-X Host Coalescing

Legacy Host Coalescing (HC) parameter registers do not offer granularity in terms of individual RX return queues or individual TX queues. Instead, all return rings were collectively metered together while the sole send ring was metered separately.

The BCM5725/BCM5762/BCM57767 will offer HC parameters or coalescing control on a per-RX-queue and TX-queue basis — when and only when MSI-X Multivector mode is chosen. To that end, a few more sets of Host Coalescing Parameter registers are added. Each such HC Parameter Set comprises the following registers:

- Receive [n] Coalescing Ticks Register (RCTR)
- Send [m] Coalescing Ticks Register (SCTR)
- Receive [n] Max Coalesced BD Count Register (RMCBCR)
- Send [m] Max Coalesced BD Count Register (SMCBCR)
- Receive [n] Max Coalesced BD Count During Interrupt Register (RMCBCDIR)
- Send [m] Max Coalesced BD Count During Interrupt Register (SMCBCDIR)

Where n ranges from [0 through 4] and m ranges from [0 through 1]. The legacy HC Parameter registers are now referred to as HC Parameter Set [0].

Each of these new HC sets have the exact same behavior or attribute as defined for the legacy HC capabilities except in one aspect: When MSI-X Multivector mode is enabled, each of the HC sets associate with their respective status block. However, when either MSI-X is disabled or MSI-X Single-Vector mode is chosen, none of the additional HC sets would exist. Recall that when MSI-X Single-Vector mode is enabled, even though 5 (or 6) vectors are advertised, only vector #0 is active.

Note: To further clarify, in legacy INTx mode or MSI mode or MSI-X Single-Vector mode, all transmit and receive queues are metered collectively by HC parameter Set 0. Sets 1 through 5 do not exist. Only in multivector MSI-X mode do HC parameter sets 1 through 5 come into existence.

The table below summarizes the existence of HC parameter sets and their association to status blocks.

Table 102: HC Parameter Sets

		EAV Mode		***RSS Mode***	
Valid HC Parameter Register Set	***Invokes Status Block#***	***HC Parameter Registers Address***	***Indication Items***	***HC Parameter Registers Address***	***Indication Items***
RCTR[0]	0	–	Send 1 LinkStat Errors	–	Send Link status change Errors
SCTR[0]		0x3C0C		0x3C0C	
RMCBCR[0]		–		–	
SMCBCR[0]		0x3C14		0x3C14	
RMCBCDIR[0]		–		–	
SMCBCDIR[0]		0x3C24		0x3C24	
RCTR[1]	1	0x3D80	RXQ 0	0x3D80	RSS 0
RMCBCR[1]		0x3D88		0x3D88	
RMCBCDIR[1]		0x3D90		0x3D90	
RCTR[2]	2	0x3D98	RXQ 1	0x3D98	RSS 1
RMCBCR[2]		0x3DA0		0x3DA0	
RMCBCDIR[2]		0x3DA8		0x3DA8	
RCTR[3]	3	0x3DB0	RXQ 2	0x3DB0	RSS 2
RMCBCR[3]		0x3DB8		0x3DB8	
RMCBCDIR[3]		0x3DC0		0x3DC0	
RCTR[4]	4	0X3DC8	RXQ 3	0X3DC8	RSS 3
RMCBCR[4]		0x3DD0		0x3DD0	
RMCBCDIR[4]		0x3DD8		0x3DD8	
SCTR[1]	5	0x3D84	Send 2	N/A	N/A
SMCBCR[1]		0x3D8C			
SMCBCDIR[1]		0x3D94			

End of Receive Stream Interrupt

A new kind of forced interrupt is being introduced in the BCM5725/BCM5762/BCM57767. The End of RX Stream Interrupt attempts to sense the end of a receive burst and if it does it fires an interrupt/MSI-X instantaneously.

After completing the DMA of every return BD to host memory, a hardware finite state machine (FSM) checks if the RX-MBUF is empty (discounting the effects of pre-allocation). If it is then hardware starts counting down a count value. While the count down is in progress, if another RX packet starts to pour into the RX-MBUF, the FSM goes back to idle. However if no other RX packet arrives the counter reaches zero at which point the FSM triggers an interrupt/MSI-X. The counter basically debounces any effects resulting from inter-packet gap or short gaps among packets within a receive burst.

This feature may be enabled or disabled by a register bit. The count-down pre-load value is also programmable. When enabled in conjunction with Multivector MSI-X mode there is a programmable option to either fire vector #0 only or fire all vectors.

Table 103: Host Coalescing Mode Register (Offset 0x3C00)

Name	*Bits*	*Access*	*Default Value*	*Description*
As defined in Legacy	31	–	–	–
End of RX Stream Detector Fires ALL MSI-X Vectors	30	RW	0x0	Write 1 to fire ALL MSI-X vectors when an end-of-RX stream is detected. Write 0 to fire only MSI-X vector#0 when an end-of-RX stream is detected.
Enable End of RX Stream Interrupt	29	RW	0x0	Write 1 to enable the End of RX Stream Interrupt.
Reserved	28:18	RO	0x0	–
Coalesce Now MSI-X Vector# [5–1]	17:13	WC	0x0	Individual Coalesce Now bits associated with MSI-X vector# 5 through 1. These bits are self-clearing.
As defined in Legacy	12:0	–	–	–

Table 104: End Stream Debounce Register (Offset 0x3CD4)

Name	*Bits*	*Access*	*Default Value*	*Description*
As defined in Legacy	31	–	–	–

Table 104: End Stream Debounce Register (Offset 0x3CD4) (Cont.)

Name	Bits	Access	Default Value	Description
End of RX Stream Debounce Count	15:0	RW	0x000F	This field is meaningful only when 0x3C00[29] is 1. After completing the DMA of every return BD to the host memory, a hardware FSM checks if the RX-MBUF is empty (discounting the effects of pre-allocation). If it is, hardware starts counting down a count value programmed by this field. While the count down is in progress, if another RX packet starts to pour into the RX-MBUF, the FSM goes back to idle. However, if no other RX packet arrives, it allows the counter to go down to zero, at which point the FSM triggers an interrupt/MSI-X. The counter basically debounces effects of IPG or short gaps among packets within a burst. The counter counts in core-clocks.

Misc MSI-X Controls

MSI-X One Shot Mode

The BCM5725/BCM5762/BCM57767 introduces a new method of MSI-X acknowledgement known as the One Shot mode. When this mode is set, an ISR would be asked to skip the interrupt acknowledgement step in which it would otherwise write a non-zero value to the respective INT MailBox. This is sensible because MSI-X vector messages are equivalent to edge-triggered non-shared interrupt events, hence there is no need for an ISR to explicitly acknowledge the event.

The One Shot mode shall be enabled by default and could be disabled by writing a 1 to the register bit 0x6000[5]. The One Shot mode setting has no effect on the Line Interrupt or MSI modes.

The chip hardware behavior would be to actually store a non-zero value to an INT Mailbox as soon as a respective MSI-X Message DMA is completed at the EP-RC (PCIe Core) interface.

Coalesce Now or Forced Update

There is a Coalesce Now bit in the legacy Host Coalescing block, 0x3C00[3]. If set, the Host Coalescing block updates the Status Block immediately and sends an interrupt to host. This bit is self-clearing.

In the case of the BCM5725/BCM5762/BCM57767, this bit will retain the same functionality and associate with Status-Block0 and Vector#0 when MSI-X is enabled. Moreover, 5 more Coalesce Now bits shall be added to replicate the same function associated to vector numbers 1 through 5. Below are the definitions of the bits.

- 0x3C00[3]: Coalesce Now (When INTx or MSI enabled)
- 0x3C00[3]: Coalesce vector#0 Now (when MSI-X Enabled)
- 0x3C00[13]: Coalesce vector#1 Now (when MSI-X enabled + Multivector mode enabled)
- 0x3C00[14]: Coalesce vector#2 Now (when MSI-X enabled + Multivector mode enabled)
- 0x3C00[15]: Coalesce vector#3 Now (when MSI-X enabled + Multivector mode enabled)
- 0x3C00[16]: Coalesce vector#4 Now (when MSI-X enabled + Multivector mode enabled)

- 0x3C00[17]: Coalesce vector#5 Now (when MSI-X enabled + Multivector mode enabled)

Misc Coalescing Controls

There are a few host-coalescing controls in legacy NetXtreme design in the Host Coalescing Mode register (0x3C00) and Host Control register (0x68). Some of these controls apply equally to the newly added HC parameters or MSI-X feature in general, while some do not apply equally.

Such controls are listed below for clarity:

- Broadcom Tagged Status mode (0x68[9])

  Enabled by setting the Status Tagged Status mode bit of the Miscellaneous Host Control register. When enabled, a unique eight-bit tag value will be inserted into the Status Block Status Tag at location 7:0. The Status Tag can be returned to the Mailbox 0 register at location 31:24 by the host driver. When the Mailbox 0 register field 23:0 is written with a zero value, the tag field of the Mailbox 0 register is compared with the tag field of the last Status Block to be DMAed to the host. If the tag returned is not equivalent to the tag of the first Status Block DMAed, the interrupt state is entered. This bit applies to all 17 MSI-X vectors.
- Clear Interrupt, Mask Interrupt, Mask mode (0x68[0], 0x68[1], 0x68[8])

  These bits have no effect on MSI-X operations.
- CLear ticks on RX BD Events mode (0x3C00[9])

  Enabled by setting the Clear Ticks mode on RX bit of the Host Coalescing Mode register. When enabled, the counters initialize to the idle state and begin counting only after a receive BD event is detected. This bit also applies to all newly created RCTR Registers
- No interrupt on force update (0x3C00[11])

  Enabled by setting the No Interrupt on Force bit of the Host Coalescing Mode register. After enabling this bit, subsequent writes to the Coalesce Now bit(s) of the Host Coalescing Mode register will cause status block update(s) without the corresponding interrupt event. This bit applies to all MSI-X vectors and respective Status Blocks.
- No interrupt on DMAD force (0x3C00[12])

  Enabled by setting the No Interrupt on DMAD force bit of the Host Coalescing Mode register. When enabled, the BD_FLAG_COAL_NOW bit of the buffer descriptor (BD) may be set to force a status block update without a corresponding interrupt. This feature is associated with Send BDs only - hence shall apply to Vector#0 in MSI-X mode when Multiple Send Queues are not enabled.

Note: The HC RTL honors the Coal_Now Flag coming from both Send or Receive FTQs - the RX FTQ never requests it as the Receive buffer descriptors (RBDs) do not support any such flag.

- Do not interrupt on receives (0x6800[14])

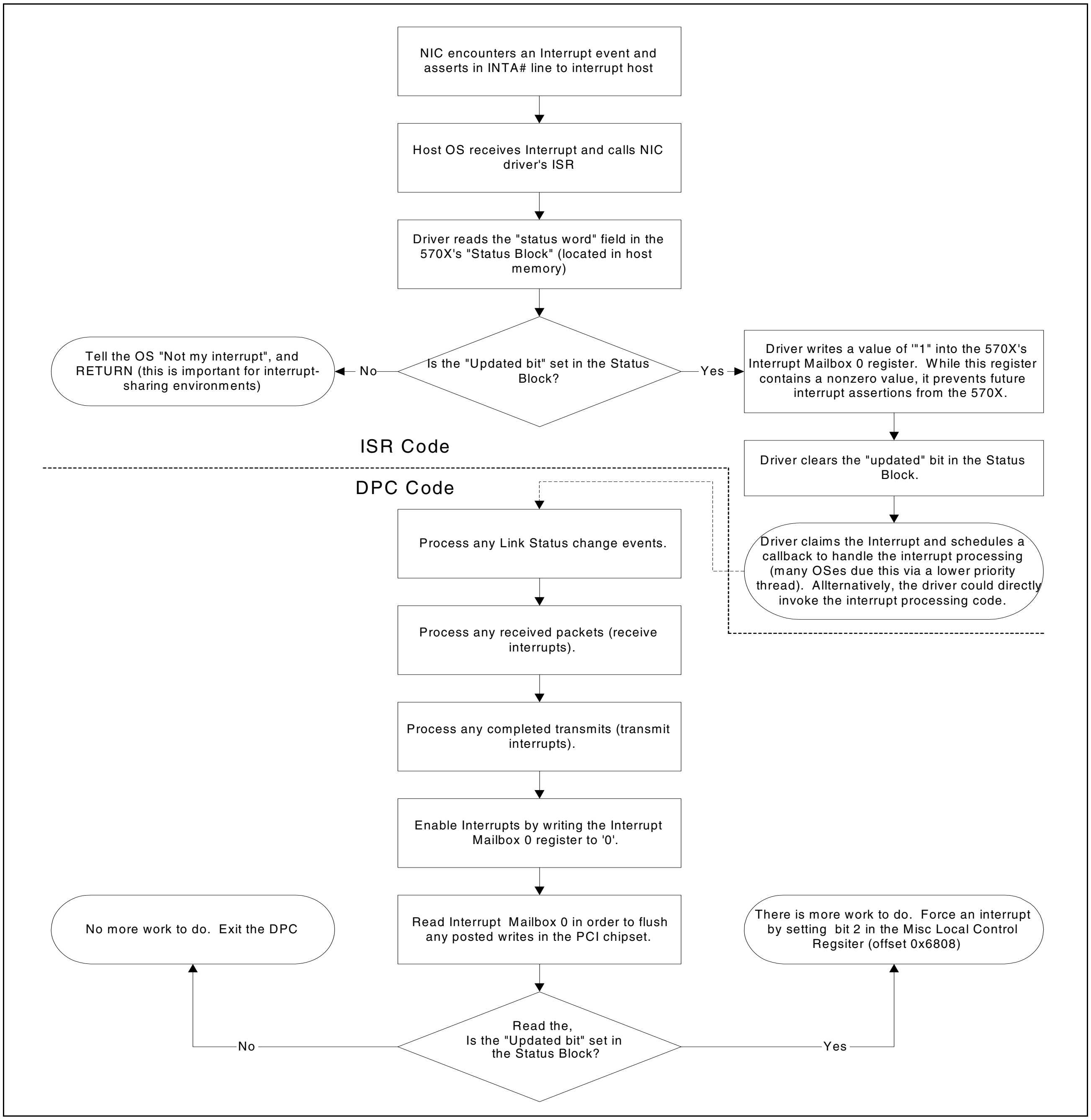
  If set, an interrupt is not generated upon a Receive Return Ring producer update. This bit shall apply equally to vector#0 through vector#16 in Multivector mode.

Basic Driver Interrupt Processing Flow

Flowchart for Servicing an Interrupt

The following figure shows the basic driver interrupt service routine flow.

Figure 59: Basic Driver Interrupt Service Routine Flow


NIC encounters an Interrupt event and asserts in INTA# line to interrupt host
Host OS receives Interrupt and calls NIC driver's ISR
Driver reads the "status word" field in the 570X's "Status Block" (located in host memory)
Is the "Updated bit" set in the Status Block?
No
Tell the OS "Not my interrupt", and RETURN (this is important for interrupt-sharing environments)
Yes
Driver writes a value of '"1" into the 570X's Interrupt Mailbox 0 register. While this register contains a nonzero value, it prevents future interrupt assertions from the 570X.
Driver clears the "updated" bit in the Status Block.
Driver claims the Interrupt and schedules a callback to handle the interrupt processing (many OSes due this via a lower priority thread). Allternatively, the driver could directly invoke the interrupt processing code.
ISR Code
DPC Code
Process any Link Status change events.
Process any received packets (receive interrupts).
Process any completed transmits (transmit interrupts).
Enable Interrupts by writing the Interrupt Mailbox 0 register to '0'.
Read Interrupt Mailbox 0 in order to flush any posted writes in the PCI chipset.
Read the, Is the "Updated bit" set in the Status Block?
No
No more work to do. Exit the DPC
Yes
There is more work to do. Force an interrupt by setting bit 2 in the Misc Local Control Regsiter (offset 0x6808)

Interrupt Procedure

1. Acknowledge interrupt. Write a nonzero value (i.e., value = 1) to the interrupt mailbox 0 (see "Interrupt Mailbox 0 (High Priority Mailbox) Register (offset: 0x200-207)" on page 319 for host standard and "Interrupt Mailbox 0 Register (offset: 0x5800)" on page 513 for indirect mode) to indicate that the driver is currently processing the interrupt. This step disables device interrupts except during interrupt feature.
2. Read and save the value of the Status Tag field of the Status Block (see "Status Block" on page 82).
3. Claim interrupt. Determine if the Ethernet controller action is required. Read the Updated bit of the status word. If the Updated bit is asserted, then the host coalescing engine has updated the status block.
4. Clear the Updated bit of the status word. This indicates that the host driver either has or will touch the status block. If a during interrupt event is driven, the host driver can examine the Updated bit to determine if a fresh status block has been moved to host memory space.
5. Check for RX traffic.
   - Loop through enabled RX Return Rings (0 to 3).
   - Check for difference between RX Return Ring Producer index (Status block) and RX Return Ring Consumer index (value written to mailbox on previous call) are the number of frames to process for RX Return Ring.
   - Process the packet.
   - Update the RX Return Ring consumer pointer in each mailbox for new RX frames.
6. Check for TX completes.
   - Loop through enabled TX Send Rings.
   - Check for difference between previous consumer index (software kept) and current consumer index in the status block. These are the TX BDs which can be made available to next send operation.
   - Update the previous consumer index (i.e., next call) to the value of the status block consumer index.
7. Compare the current value of the Status Tag to the saved value of the Status Tag. Flush status block (i.e., force update of status blocks cached by PCI bridge).
   - Read interrupt mailbox (see "Interrupt Mailbox 0 (High Priority Mailbox) Register (offset: 0x200-207)" on page 319 for host standard and "Interrupt Mailbox 0 Register (offset: 0x5800)" on page 513 for indirect mode).
   - Check the Updated bit in the status word located in the status block. If the Updated bit is asserted, then new data has been DMAed to the host. Repeat steps 5 and 6.
8. Check the Error bit in status word (optional). The driver may check the state machine/FTQ status registers for various attentions.
9. Enable interrupts. When Status Tagged Status mode bit of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256) is set to 1, then write the saved Status Tag to the upper 8 bits of Interrupt Mailbox 0, and 0 to the remaining bits (23 down to 0) to indicate that the ISR is done processing RX/TX. Otherwise, write 0 to Interrupt Mailbox 0 register. This step also clears existing interrupts.

Other Configuration Controls

Broadcom Mask Mode

Enabled by setting the Mask_Interrupt_Mode bit (bit 8) of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). When enabled, setting the mask bit of the Miscellaneous Host Control register will mask (deassert) the INTA signal at the pin, but it will not clear the interrupt state and it will not latch the INTA value. Clearing the mask bit will enable the interrupt state to propagate to the INTA signal. Note that the During Interrupt Coalescence registers are only used when the Mailbox 0 is set.

Broadcom Tagged Status Mode

Enabled by setting the Status Tagged Status mode bit of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). When enabled, a unique eight-bit tag value will be inserted into the Status Block Status Tag at location 7:0. The Status Tag can be returned to the Mailbox 0 register at location 31:24 by the host driver. When the Mailbox 0 register field 23:0 is written with a zero value, the tag field of the Mailbox 0 register is compared with the tag field of the last Status Block to be DMAed to the host. If the tag returned is not equivalent to the tag of the first Status Block DMAed, the interrupt status is entered.

Clear Ticks on BD Events Mode

Enabled by setting the Clear Ticks mode on RX or the Clear Ticks mode on TX bits of the Host Coalescing Mode register (see "Miscellaneous Host Control Register (offset: 0x68) — Function 0" on page 256). When enabled, the counters initialize to the idle state and begin counting only after a receive or transmit BD event is detected.

No Interrupt on Force Update

Enabled by setting the No Interrupt on Force bit of the Host Coalescing Mode register (see "Host Coalescing Status Register (offset: 0x3C04)" on page 444). When enabled, writing the Force update bit of the Host Coalescing Mode register will cause a status block update without a corresponding interrupt event.

No Interrupt on DMAD Force

Enabled by setting the No Interrupt on DMAD force bit of the Host Coalescing Mode register (see "Host Coalescing Status Register (offset: 0x3C04)" on page 444). When enabled, the BD_FLAG_COAL_NOW bit of the buffer descriptor may be set to force a status block update without a corresponding interrupt.

Section 11: Multifunction Interdependencies

Introduction

This section describes the interdependencies that exist between the LAN PCIe function (function 0) of the BCM5725/BCM5762/BCM57767 high-speed controller and the card-reader PCIe function(s).

Abbreviations

BJP Bonjour proxy firmware is one example of power management offload firmware (other "sleep" firmware applications also exist) that is used in system state S3 (device state D3), which gets loaded by the LAN driver in some system designs (and essentially replaces boot code) as part of a system entering a low power sleep state. This firmware, among other things, offloads certain types of network activity to reduce the frequency with which the system must exit S3 and power back up into S0, thereby saving system power.

BC Boot code firmware which loads from an external NVRAM and executes upon either a power-on reset or a warm reset of the BCM5725/BCM5762/BCM57767 controller. Boot code (and proxy firmware) executes on an integrated RISC cpu core inside the BCM5725/BCM5762/BCM57767 controller.

CR Card reader (SD, SDHC, SDXC, MMC, MSPro, xD-Picture)

GRC Reset General register control reset. A warm reset of the LAN function (function 0).

D3 An ACPI device sleep state. D0 = device fully powered.

S3 An ACPI system sleep state. S0 = system fully powered.

Perst_L PCI Express (PCIe) reset signal. Transition of this signal causes boot code to reload and execute.

Vmain Full system power is present when Vmain is active.

Vaux Auxiliary (i.e. standby) power. Minimal logic within the BCM5725/BCM5762/BCM57767 controller is powered when running on Vaux only. The integrated RISC CPU core is powered on Vaux and continues executing code (either BC or BJP).

CR Sig Card-reader signature. Somewhat analogous to the legacy LAN driver signature in shared memory offset 0xB50, the CR sig is used to communicate between BC/BJP firmware and the CR driver.

LAN Sig Legacy LAN signature in shared memory offset 0xB50. BC writes the value of the 1's complement of the ASCII string KevT (i.e. ~KevT). When the LAN driver wishes to perform a warm LAN function reset (a.k.a. GRC reset), it first writes the value KevT to 0xB50. The LAN driver can then poll 0xB50 waiting for ~KevT to know that boot code has fully loaded, initialized the controller, and entered its main service loop and is now safe for the driver to access the LAN function (function 0) of the device.

OOB Out-of-Box. This refers to the power state of the device upon plugging the A/C cord into the wall, which immediately powers Vaux only (not Vmain) to the device. Boot code loads and executes upon entering the OOB system state as the integrated RISC CPU core is powered by Vaux.

Background

The BCM5725/BCM5762/BCM57767 controller contains additional integrated functionality beyond the traditional Ethernet LAN function. Among the additional functionality is support for various CR interfaces.

The CR functionality is implemented in the form of additional separate PCIe functions within the BCM5725/ BCM5762/BCM57767 device, wholly (or mostly) separate from the primary function 0 LAN interface.

While every effort was made in the architecture of the BCM5725/BCM5762/BCM57767 controller to make each PCIe function totally independent from one another, there are still a few inter-function interactions that device driver developers need to be aware of. All functions within the device share the same PCIe bus interface to the host system. For example, if the system wishes to place function 0 (LAN) into a low-power sleep state, the PCIe bus must remain powered if the CR functions are to remain active.

Although SD, SDHC, SDXC, MMC, MSPro, xD-Picture CR types are supported, only SD (on PCIe function 1) is used in the Broadcom "currently shipping product" at the time of initial release of this document.

For the BCM5725/BCM5762/BCM57767 controller, there are two important signatures that are used to coordinate activity between the boot code and the driver(s). In previous LAN-only Broadcom NetXtreme controllers (no CR functionality), there was only a single signature located at controller memory offset 0xB50 (accessible by both boot code and a host LAN driver). 0xB50 would contain essentially garbage upon OOB power-on, and boot code would write a value of 0x49A89AB (a.k.a. ~KevT). The LAN driver should monitor this signature to become KevT following a power-on (cold) reset to know when it is safe to access the device. Whenever the LAN driver wishes to reset the LAN function only (not reset the CR function), it should first write the value 0x4B657654 (a.k.a. ASCII "KevT") to the signature in 0xB50 to signify to the boot code that it is a warm reset, then initiate a GRC reset by setting bit 1 in function 0 register 0x6804 (Miscellaneous Configuration register).

It is important to note that a GRC reset is different from a PCI reset (full device reset, initiated by host) in that the former only resets the LAN function (function 0) and does not disturb the CR function (function 1), whereas a PCI reset (either up cold power-on or upon host action) fully resets the entire device including both the LAN and CR functions. A PCI reset is initiated by the host in one of two ways:

- Host hardware asserts the PCIe bus Perst signal
- Host root complex sends an in-band PCIe reset command

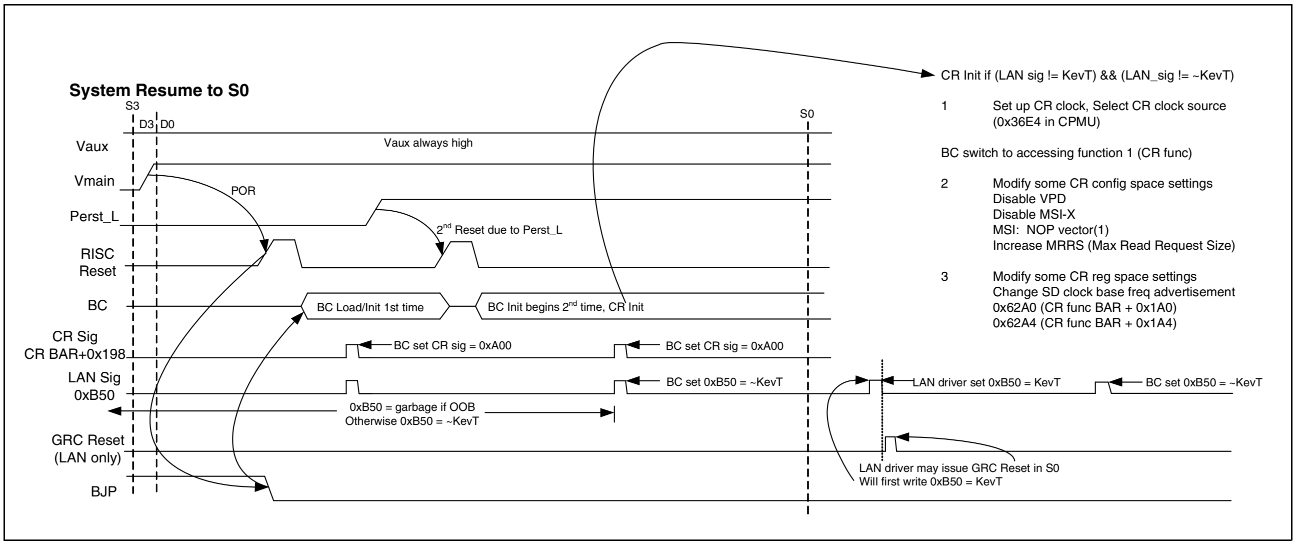
A recently added second signature is for use exclusively by a CR driver to determine the initialization state of the CR function initialization. The new CR signature is located at CR BAR + 0x198 bits 11:8. Bootcode or, alternatively, Bonjour proxy firmware, writes a value of 0xA to this signature location upon completion of CR function initialization following either a power-on reset or a full device (PCIe) reset. The BC/BJP firmware also clears (write 0) this signature upon detecting that Vmain is going down.

The CR signature's functionality is largely analogous to how the 0xB50 signature is used by the LAN driver, but this method allows a CR driver to communicate with the boot code without having to access function 0. This method also eliminates any need to have the two drivers (LAN and CR) handshake with each other, which is undesirable from an overall system design perspective. It is assumed that a system architect would strongly prefer the LAN driver and the CR driver to be able to behave as if the LAN and CR functions exist as two distinctly separate system devices — even though this is not the physical reality.

The original reason for adding the CR signature (in addition to the legacy LAN signature) was because of an issue observed that any time the LAN driver initiated a GRC reset to the LAN function (function 0), the boot code would re-execute and perform a full device initialization, including CR function initialization. This was a problem if there was any ongoing CR activity on any of the other functions. It became necessary to find a way for boot code to know when it is safe to initialize the CR function(s).

The diagrams below show the device/function activities that take place when the system resumes to S0 from a low-power sleep state, and for the reverse use case scenario, entering a sleep state (i.e., transition from S0 to S3).

Figure 60: Device/Function Activities when System Resumes to S0



CR Init if (LAN sig != KevT) && (LAN_sig != ~KevT)

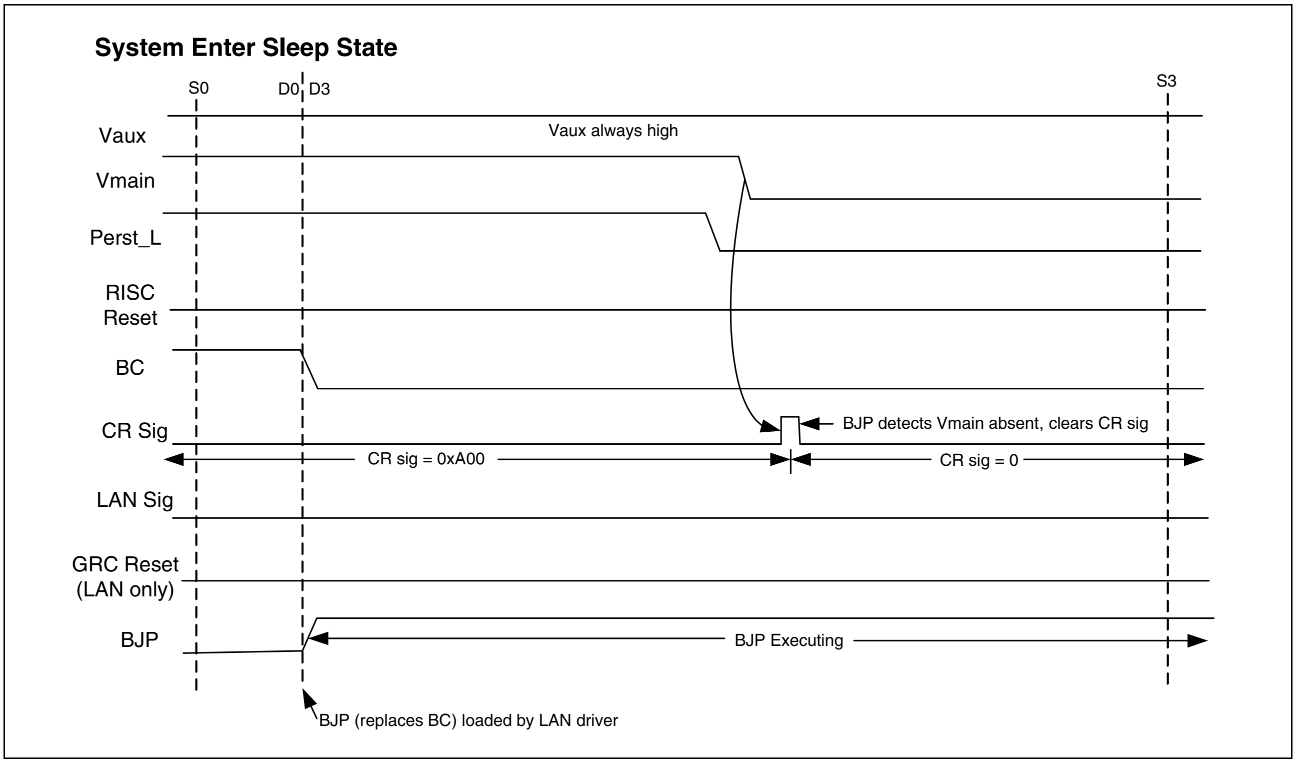
1 Set up CR clock, Select CR clock source
(0x36E4 in CPMU)

BC switch to accessing function 1 (CR func)

2 Modify some CR config space settings
Disable VPD
Disable MSI-X
MSI: NOP vector(1)
Increase MRRS (Max Read Request Size)

3 Modify some CR reg space settings
Change SD clock base freq advertisement
0x62A0 (CR func BAR + 0x1A0)
0x62A4 (CR func BAR + 0x1A4)

Figure 61: Device/Function Activities when System Enters Sleep State



Related Information

The following are related multifunction interdependencies:

- The BCM5725/BCM5762/BCM57767 BC/BJP firmware executes on Vaux only, regardless of Vmain state.
- BC/BJP firmware detects function 0 register 0x6808[19] transitioning from 1 to 0 to know that the system is going down (Vmain died).
- BC/BJP firmware clears the CR sig (CR BAR + 0x198[11:8]=0) whenever the firmware detects that the system is going down (Vmain is going down).
- BC/BJP firmware also clears the CR sig if there is a GRC reset, and Vmain is going down, and the LAN signature is uninitialized (0xB50 = garbage). The use case here would be an "unexpected shutdown."
- BC executes in device state D0. Proxy (i.e. BJP) executes in D3. In some systems, the LAN driver loads the proxy firmware as part of the activity undertaken when entering a system sleep state (i.e. S3). Otherwise, only BC firmware will be running, and it handles all of the same necessary setting/clearing signature duties.
- At the time of this document release, the CR sig activities are not available if using either standard self-boot or OTP-based self-boot. Legacy external NVRAM-based boot code must be used instead. Refer to Broadcom Application Note 5754X_5787X-AN100-R and NetXtreme-AN404-R for additional information regarding self-boot.

Section 12: Ethernet Controller Register Definitions

Purpose

This section details the registers that are implemented in the BCM5725/BCM5762/BCM57767 device. The purpose of this section is to provide accurate register implementation information to engineering teams so that they can develop associated products (such as software suites) and can validate, test, and support all functions in the device effectively.

Scope

The scope of this engineering register specification covers the entire set of MAC registers in the BCM5725/BCM5762/BCM57767 controller. It includes all MAC Core registers and all PCIe registers.

The Gigabit PHY registers are described in Section 13: "Transceiver Registers," on page 574

PCI Configuration Registers

Note: Unless otherwise noted, all register specifications in this document apply to the BCM5778X, BCM5776X, and BCM5779X controllers.

Register Access Legend

- RO = Read-only
- R = Read-only
- FW-RW = Readable and writeable by controller internal CPU only. Not accessible from host CPU.
- Host RW = Readable and writeable by host CPU only.
- Host RO = Read-only by host CPU
- W2C = Write to clear
- RW2C = Read and write to clear
- RW1CS = Read/write 1 to clear and sticky (reset only by POR reset - perst_n can not reset this bit)
- RWS = Read/write sticky
- ROS = Read-only sticky
- W0C = Write 0 to clear
- W1 = Write 1 but read always returns 0
- UUUU = Unknown
- NA = Not applicable

LAN Controller Configuration Register — Function 0

The following describes the registers which are required by the PCI Express specifications for configuration. The primary reset for these registers is PCIe Reset. Registers that are labeled "not used in BCM5725/BCM5762/BCM57767" or "not applicable in BCM5725/BCM5762/BCM57767" will return all 0x0 when read.

Device ID and Vendor ID Register (Offset: 0x00) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Device ID	31:16	FW-RW Host-RO	–	Default for BCM57781 (LAN Function 0): 16B1 Default for BCM5725/BCM5762/BCM57767 (LAN Function 0): 16B5 Default for BCM57761 (LAN Function 0): 16B0 Default for BCM57765 (LAN Function 0): 16B4 Default for BCM57791 (LAN Function 0): 16B2 Default for BCM57795 (LAN Function 0): 16B6 For those members of the controller family that have card reader functionality the card reader device IDs are as follows: • SD/MMC 0x16BC • MS 0x16BE • xD 0x16BF
Vendor ID	15:0	FW-RW Host-RO	0x14E4	

Status and Command Register (Offset: 0x04) — Function 0

This register is reset by PCIe Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Detected Parity Error	31	R/W2C	0x0	When this bit is set, it indicates that the function has received a poisoned TLP
Signaled System Error	30	R/W2C	0x0	This bit is set when a function sends an ERR_FATAL or ERR_NONFATAL message and the SERR enable bit in the command register is set
Received Master Abort	29	R/W2C	0x0	This bit is set when a requester receives a completion with UR completion status
Received Target Abort	28	R/W2C	0x0	This bit is set when a requester receives a completion with completer abort completion status.
Signaled Target Abort	27	R/W2C	0x0	This bit is set when a function acting as a completer terminates a request by issuing Completer abort completion status to the requester
DEVSEL Timing	26:25	RO	0x0	Does not apply to PCIe

Name	***Bits***	***Access***	***Default Value***	***Description***
Master Data Parity Error	24	R/W2C	0x0	The master data parity error bit is set by a requester if the parity error enable bit is set in its command register and either of the following 2 conditions occur. If the requester receives a poisoned completion if the requester poisons a write request If the parity Error enable bit is cleared , the master data parity error status bit is never set
Fast Back-to-back capable	23	RO	0x0	Does not apply to PCIe.
Reserved	22	RO	0x0	These bits are reserved and tied low per the PCI specification.
66MHz Capable	21	RO	0x0	Does not apply to PCIe
Capabilities List	20	RO	0x1	This bit is tied high to indicate that the device supports a capability list. The list starts at address 0x40.
Interrupt Status	19	RO	0x0	Indicates this device generated an interrupt
Reserved	18:16	RO	0x0	These bits are reserved and tied low per the PCIe specification.
Reserved	15:11	RO	0x00	These bits are reserved and tied low per the PCIe specification.
Interrupt Disable	10	RW	0x0	When this bit is set, function is not permitted to generate IntX interrupt messages (deasserted) regardless of any internal chip logic. Setting this bit has no effect on the INT_STATUS bit below. Writing this bit to 0 will un-mask the interrupt and let it run normally.
Fast Back-to-back Enable	9	RO	0x0	Does not apply to PCIe
System Error Enable	8	RW	0x0	When set, this bit enables the non fatal and fatal errors detected by the function to be reported to the Root Complex. The function reports such errors to the Root Complex if it is enabled to do so either through this bit or though PCI express specific bits in DCR
Stepping Control	7	RO	0x0	Does not apply to PCIe
Parity Error Enable	6	RW	0x0	This bit enables the write to the Master data parity error status bit. If this bit is cleared, the master data parity error status bit will never be set.
VGA Palette Snoop	5	RO	0x0	Does not apply to PCIe
Memory Write and Invalidate	4	RO	0x0	Does not apply to PCIe
Special Cycles	3	RO	0x0	Does not apply to PCIe
Bus Master	2	RW	0x0	This bit controls the enabling of the bus master activity by this device. When low, it disables an Endpoint function from issuing memory or IO requests. Also disables the ability to issue MSI messages.

Name	Bits	Access	Default Value	Description
Memory Space	1	RW	0x0	This bit controls the enabling of the memory space. When disabled, memory transactions targeting this device return completion with UR status
I/O Space	0	RO	0x0	This bit indicates that the device does not support I/O space access because it is zero and can not be modified. IO transactions targeting this device return completion with UR status.

PCI Classcode and Revision ID Register (offset: 0x8) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Classcode	31:8	RO	0x020000	Default for (LAN Function 0): 0x020000
Revision ID — All-layer Revision ID	7:4	FW-RW Host-RO	ASIC Rev Input	This field will be updated automatically by hardware based on the External All Layer Revision ID. For example, this field will contain a value of 0x0 after hard reset for BCM5725/BCM5762/BCM57767 A0 silicon. Software shall use this field only to display the Device Silicon Revision ID for application where the user/ customer needs to know the Device Silicon Revision ID. One such application is the B57DIAG Device Banner. Furthermore, Software (Boot Code/Driver/B57DIAG) shall NOT use this field in determining Bug Fixes. It should only use the Internal Revision ID, bits 31:24 and bits 19:16 from Register 68, for that purpose. • 0x0 for A steps • 0x1 for B steps • 0x2 for C steps
Revision ID — Metal Revision ID	3:0	FW-RW Host-RO	ASIC Rev Input	This field will be updated automatically by hardware based on the Metal Revision ID. • 0x0 for metal 0 step • 0x1 for metal 1 step • 0x2 for metal 2 step

BIST, Header Type, Latency Timer, Cache Line Size Register (offset: 0x0C) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
BIST	31:24	RO FW-RW	0x0	The 8-bit BIST register is used to initiate and report the results of any Built-In-Self-Test. This value can be written by firmware through the PCI register space BIST register to modify the read value to the host.
Header Type	23:16	RO	0x80	The 8-bit Header Type register identifies both the layout of bytes 10h through 3Fh of the Configuration space, as well as whether this adapter contains multiple functions. A value of 0x80 indicates a multi function device (Type 0) using the format specified in the PCI specification, while a value of 0x0 indicates a single function Type 0 device.
Latency Timer	15:8	RO	0x0	This register does not apply to PCI express and must be hardwired to zero

Name	*Bits*	*Access*	*Default Value*	*Description*
Cache Line Size	7:0	RO FW-RW	0x0	This field is implemented by PCIe device as a read/ write field for legacy compatibility purposes.

Base Address Register 1 (offset: 0x10) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Base Address	31:xx	RW	0x0	These bits set the address within a 32-bit address space that will be card will respond in. These bits may be combined with the bits in BAR_2 to create a full 64 bit address decode. Only the bits that address blocks bigger than the setting in the BAR1_SIZE value are RW. All lower bits are RO with a value of zero. This value is sticky and only reset by HARD Reset.
Size indication	xx-1:4	RO	0x0	RO bits indicate size of memory space. For BCM5778X, BCM5779X, BCM5776X: 15:4 RO=0 (64KB BAR) if flat_view = 0 24:4 RO=0 (32MB BAR) if flat_view = 1 See Register 0x70 for definition of flat_view.
Prefetchable	3	RO	0x0 Strap input to pcie block	This bit indicates that the area mapped by BAR_1 may be pre-fetched or cached by the system without side effects. Bit can be programmed from shadow register.
Type	2:1	Host RO	0x2 Strap input to pcie block	These bits indicate that BAR_1 may be programmed to map this adapter to anywhere in the 64-bit address space. Encoded with the following values: 00: located anywhere in 32-bit address space 01: reserved 10: located anywhere in 64-bit address space 11: reserved For function 0, this value is 2 (64-bit enabled)
Memory Space Indicator	0	RO	0x0	This bit indicates that BAR_1 maps a memory space and is always read as 0.

Base Address Register 2 (offset: 0x14) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Base Address	31:0	RW RO if bar1_64ena is 0'0	0x0	The 32-bit BAR_2 register programs the upper half of the base address for the memory space mapped by the card onto the PCI bus. These bits set the address upper 32-bit address space. These bits may be combined with the bits in BAR_1 to create a full 64 bit address decode. These bits must be set to zero for the card to respond to single address cycle requests. This value is sticky and only reset by HARD Reset.

Base Address Register 3 (offset: 0x18) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Base Address 2	31:0	RO	0x0	For BCM5725/BCM5762/BCM57767, BAR 3 is disabled.

Base Address Register 4 (offset: 0x1c) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Base Address 2	31:0	RO	0x00	For BCM5725/BCM5762/BCM57767, BAR 4 is disabled.

Cardbus CIS Pointer Register (offset: 0x28) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Cardbus CIS Pointer	31:0	FW-RW Host — RO	0x0	N/A for PCIe Device

Subsystem ID/Vendor ID Register (offset: 0x2C) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Subsystem Device ID	31:16	FW-RW Host — RO	–	Default for BCM57781 (LAN Function 0): 96B1 Default for BCM5725/BCM5762/BCM57767 (LAN Function 0): 96B5 Default for BCM57761 (LAN Function 0): 16B0 Default for BCM57765 (LAN Function 0): 16B4 Default for BCM57791 (LAN Function 0): 96B2 Default for BCM57795 (LAN Function 0): 96B6
Subsystem Vendor ID	15:0	FW-RW Host — RO	0x14E4	Identifies board manufacturer

Expansion ROM Base Address Register (offset: 0x30) — Function 0

This register is reset by PCIe Reset. It becomes aN RW register if bit 5 of PCI State Register is set.

Name	*Bits*	*Access*	*Default Value*	*Description*
ROM Base Address	31:24	RW	0xXXXX	These bits indicate the address of the Expansion ROM area.
ROM Size indication	23:11	RW	0x00	These bits indicate the size of the Expansion ROM area or the address of it. The boundary form RO bits to RW bits is controlled by the EXP_ROM_SIZE bits.
Reserved	10:1	RO	0x000	These bits indicate that the Expansion ROM area is at least 2k bytes. They always read as zero. P
Expansion ROM Enable	0	RW	0x0	This bit indicates that the Expansion ROM BAR is valid when set to one. If it is zero, the expansion BAR should not be programmed or used. This bit will only be RW if it is enabled by the EXP_ROM_ENA bit which defaults to 0.

Capabilities Pointer Register (offset: 0x34) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31:8	RO	0x0	Unused

Name	*Bits*	*Access*	*Default Value*	*Description*
Capabilities pointer	7:0	RO	0x48 (PM Cap)	The 8-bit Capabilities Pointer register specifies an offset in the PCI address space of a linked list of new capabilities. The capabilities are PCI-X, PCI Power Management, Vital Product Data (VPD), and Message Signaled Interrupts (MSI) is supported. The read-only value of this register is controlled by the CAP_ENA register in the PCI register space.

Interrupt Register (offset: 0x3C) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
MAXIMUM_LATENC Y	31:24	RO	0x00	Hardwired to zero
MIN_GRANT	23:16	RO	0x00	Hardwired to zero
Interrupt Pin	15:8	RO	0x01	Indicates which interrupt pin this device uses: 0: no Interrupt 1: Use Interrupt A 2: Use Interrupt B 3: Use Interrupt C 4: Use Interrupt D
Interrupt Line	7:0	RW	0x00	Identifies interrupt routing information

INT Mailbox Register (offset: 0x40-0x44) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Indirect Interrupt mail box	63:0	RW	0	Interrupt Mailbox This register is mapped to the old mailbox register B0 and B4.

Power Management Capability Register (offset: 0x48) — Functions 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PME Support	31:27	RO	0x08 if no aux 0x18 if aux	Indicates the power states in which the device may assert PME. A 0 for any bit indicates that the device is not capable of asserting the PME pin signal while in that power state. Bit 27: PME can be asserted from D0 Bit 28: PME can be asserted from D1 Bit 29: PME can be asserted from D2 Bit 30: PME can be asserted from D3H Bit 31: PME can be asserted from D3C (default depends on the presence of Aux power)
D2 Support	26	RO	0x0	Indicates whether the device supports the D2 PM state. This device does not support D2; hardwired to 0
D1 Support	25	RO FW-RW	0x0	Indicates whether the device supports the D1 PM state. This device does not support D1
Aux Current	24:22	RO FW-RW	0x0	This device supports the data register for reporting Aux Current requirements so this field is N/A.
DSI	21	RO FW-RW	0x0	Indicates that the device requires device specific initialization (beyond PCI configuration header) before the generic class device driver is able to use it. This device hardwires this bit to 0 indicating that DSI is not necessary
Reserved	20	RO	0x0	–
PME Clock	19	RO	0x0	Indicates that the device relies on the presence of the PCI clock for PME operation. This device does not require the PCI clock to generate PME. Therefore, the bit is hardwired to 0
Version	18:16	RO	0x3	A value of 011b indicates that this function complies with revision 1.2 of the PCI PM specification.
PM Next Capabilities	15:8	RO	0x58	Points to the next capabilities block which is Broadcom Vendor Specific Capability Header
PM Capability ID	7:0	RO	0x01	Identifies this item as Power management capabilities

Power Management Control/Status Register (offset: 0x4C) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PM Data	31:24	RO FW-RW	0x00	Contains the power management data indicated by the Data Select field in PMCSR
Reserved	23:16	RO	0x00	–

Name	*Bits*	*Access*	*Default Value*	*Description*
PME Status	15	RW2C	0x0	This bit is set when the device asserts the WAKE signal independent of the PME enable bit. Writing 1 this bit will clear it and cause the deice to stop asserting WAKE
Data Scale	14:13	RO	0x1	Indicates the scaling factor that is used when interpreting the value of the data register (offset 7 in PM capability space). The device hardwires this value to 1 to indicate a scale of 1x
Data Select	12:9	RW	0x0	Indicates which data is to be reported via the Data register (offset 7 in PM capability space)
PME Enable	8	RW	0x1	Enables the device to generate PME when this bit is set to 1. When 0, PME generation is disabled
Reserved	7:4	RO	0x00	–
No Soft Reset	3	RO	0x1	No_Soft_Reset When set (1), this bit indicates that devices transitioning from D3hot to D0 because of PowerState commands do not perform an internal reset. Configuration Context is preserved. Upon transition from the D3hot to the D0 Initialized state, no additional operating system intervention is required to preserve Configuration Context beyond writing the PowerState bits. When clear (0), devices do perform an internal reset upon transitioning from D3hot to D0 via software control of the PowerState bits. Configuration Context is lost when performing the soft reset. Upon transition from the D3hot to the D0 state, full reinitialization sequence is needed to return the device to D0 Initialized. Regardless of this bit, devices that transition from D3hot to D0 by a system or bus segment reset will return to the device state D0 Uninitialized with only PME context preserved if PME is supported and enabled.
Reserved	2	RO	0x0	–
Power State	1:0	RW	0x0	Indicates the current power state of the device when read. When written, it sets the device into the specified power state 00: D0 - Select D0 01: D1 - Select D1 10: D2 - Select D2 11: D3-Hot - Select D3 These bits may be used by the system to set the power state. The register is implemented as two banks of two bits each. Can be written from both configuration space and from the PCI register space as the PM_STATE bits. When written from the PCI bus, only values of 0 and 3 will be accepted. This is the register returned on reads of this register from configuration space. The second bank catches all writes values. The value of the second register is returned when the PM_STATE bits are read from register space. The idea of these registers is to a) Provide compatible operation to 5701 b) Allow implementation of other power states though firmware.

MSI Capability Header (offset: 0x58) — Function 0

The device driver is prohibited from writing to this register.

Name	*Bits*	*Access*	*Default Value*	*Description*
MSI Control	31:25	R/O	0x00	Reserved
MSI_PVMASK_ CAPABLE	24	RO	0	This bit indicates if the function supports per vector masking. This value comes from the MSI_PV_MASK_CAP bit in the register space.
64-bit Address Capable	23	RO	1	Hardwired Advertise 64-bit address capable This bit indicates that the chip is capable of generating 64 bit MSI messages.
Multiple Message Enable	22:20	R/W	0x0	These bits indicate the number of message that the chip is configured (allowed) to generate. Number of allocated message: 0 1 Chip is set to generate 1 message 1 2 Chip is set to generate 2 messages 2 4 Chip is set to generate 4 messages 3 8 Chip is set to generate 8 messages 4 16 Chip is set to generate 16 messages 5 32 Chip is set to generate 32 messages
Multiple Message Capable	19:17	R/O	0x3	These bits indicate the number of messages that the chip is capable of generating. This value comes from the bit in the register space. Number of requested message: 0 1 Chip is set to generate 1 message 1 2 Chip is set to generate 2 messages 2 4 Chip is set to generate 4 messages 3 8 Chip is set to generate 8 messages 4 16 Chip is set to generate 16 messages 5 32 Chip is set to generate 32 messages
MSI Enable	16	R/W	0	When this bit is set, the chip will generate MSI cycles to indicate interrupts instead of asserting the INTA# pin. When this bit is zero, the INTA# pin will be used.
Next Capability Pointer	15:8	RO	A0 (MSIX Cap)	This value continues the PCI capability chain. It's value specified an offset in the PCI address space of the next capability. The read-only value of this register is controlled by the CAP_ENA register in the PCI register space.
MSI capability ID	7:0	RO	0x5	The 8-bit MSI Capability ID is set to 5 to indicate that the next 8 bytes are a Message Signaled Interrupt capability block.

MSI Lower Address Register (offset: 0x5c) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
MSI Lower Address	31:2	R/W	Unknown	MSI Lower Address
Reserved	1:0	RO	0	–

MSI Upper Address Register (offset: 0x60) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
MSI Upper Address	31:0	R/W	Unknown	MSI Upper Address

MSI Data Register (offset: 0x64) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
MSI Data	15:0	R/W	Unknown	MSI Data

Miscellaneous Host Control Register (offset: 0x68) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
ASIC Rev ID	31:28	R	Product ID input	0xF indicates that the new revision ID format is in effect.
	27:24	R	ASIC Rev Input	External All Layer Revision ID. These bits will reflect in offset 8 --- bit mapping description: 0x0: A 0x1: B 0x2: C
	23:16	R	ASIC Rev Input	Metal Rev Number: 0x0: 0 0x1: 1 0x2: 2
Unused	15:10	RW	0	Unused

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Tagged Status Mode	9	RW	0	When set, an unique eight-bit tag value will be inserted into the Status block status tag.
Mask Interrupt Mode	8	RW	0	When set, the interrupt is masked. However, the internal interrupt state (host coalescing event) will not be cleared.
Enable indirect access	7	RW	0	Set this bit to enable indirect addressing mode.
Enable Register Word Swap	6	RW	0	Set this bit to enable word swapping when accessing registers through the PCI target device.
Unused	5	RW	0	Set this bit enable clock control register read/write capability, otherwise, the clock control register is read only.
Enable PCI State register read/write capability	4	RW	0	Set this bit to enable PCI state register read/write capability, otherwise the register is read only.
Enable Endian Word Swap for target access	3	RW	0	Set this bit to enable endian word swapping when accessing through PCIe target interface.
Enable Endian Byte Swap for target access	2	RW	0	Set this bit to enable endian byte swapping when accessing through PCIe target interface
Mask Interrupt	1	RW	0	Setting this bit will mask future interrupt events from being generated. Setting this bit will not clear or deassert the internal interrupt state, nor will it deassert the external interrupt state.
Unused	0	WO	0	Not used in BCM5725/BCM5762/BCM57767. This bit used to be defined as the Clear Interrupt Bit. Setting this bit will clear interrupt as long as the mask interrupt bit is not set. If mask interrupt bit is set, then writing 1 to this bit will not deassert interrupt, however, it will clear the internal unmasked interrupt state, so if the interrupt is later unmasked, the interrupt will deassert.

DMA Read/Write Control Register (offset: 0x6C) — Function 0

Note: The driver does not need to access the DMA read/write control register (0x6C) because the boot code has full control of it.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:25	RW	0x0	–
CR Write Watermark	24:22	RW	0x7	CR DMA Write Watermark: 0 = 32B 1 = 64B 2 = 96B 3 = 128B 4 = 160B 5 = 192B 6 = 224B 7 = 256B
DMA Write Watermark	21:19	RW	0x7	DMA Write Watermark: 0 = 32B 1 = 64B 2 = 96B 3 = 128B 4 = 160B 5 = 192B 6 = 224B 7 = 256B
Reserved	18:10	RW	0	–
Card-Reader DMA Read MRRS	9:7	RW	0	This parameter is compared with the system MRRS. The smaller value of this value and MRRS from pcie_core will be used as for max DMA read length. 0 = 1024B 1 = 128B 2 = 256B 3 = 512B 4 = 512+256B 5 = 1024+512B 6 = 2048B 7 = 4096B

Name	*Bits*	*Access*	*Default Value*	*Description*
DMA read MRRS for slow speed	6:4	RW	0	for 10M/100M Ethernet DMA read, the smaller value of this value and MRRS from pcie_core will be used as for max DMA read length. This configure has no effect for GIGA mode. 0 = 1024B 1 = 128B 2 = 256B 3 = 512B 4 = 512+256B 5 = 1024+512B 6 = 2048B 7 = 4096B
Reserved	3:1	R/W	0	–
disable_cache_ alignment	0	RW	0	Disable cache alignment for DMA write to Host memory

PCI State Register (offset: 0x70) — Function 0

This register is reset by PCIe Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0x0	Reserved
Unused	15	FW-RW	0x1 On Hard reset	Not used in BCM5725/BCM5762/BCM57767. This bit used to be defined as the Config Retry bit which when asserted, forces all config access to be retried After 805 ms timeout from pcie_reset this bit get cleared by hardware or fast_reset (as a strap pin to pcie block) is set or retry_mode(as a strap to pcie block) is cleared this bit get cleared. FW can R/W any time.
Reserved	14:13	RO	0x0	–
PCI Vaux Present	12	RO	0x1	This bit indicates the PCI Vaux Present State
Unused	11:9	R/W	0x0	Not used in BCM5725/BCM5762/BCM57767. Used to Indicate the number of PCI clock cycles before Retry occurs, in multiple of 8. At reset, this field is set to 001. N/A in PCIe.
Unused	8	RW	0x0	Not Used in BCM5725/BCM5762/BCM57767. This bit used to be defined as Flat View
Reserved	7:5	RO	0x0	–
Pci_66_133_mode	3	RO	0x0	N/A to pcie
Pci 32 bit mode	4	RO	0x1	32 bit mode (pcie no 64 bit mode)
Reserved	3:2	RO	0x0	–
Inta_l (status)	1	RO	0x1	Inta_l status
Reserved	0	RO	0x0	–

Reset Counters Initial Values Register (offset: 0x74) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reset Counter 5 Register (PCI CLK Core Syn Reset)	31:28	Host RW	Any	Keep tracks of the number of Core Syn Reset that are synchronized in the PCI Clock Domain
Reset Counter 4 Register (Hot Reset)	27:24	Host RW	Any	Keep tracks of the number of Hot Reset events.
Reset Counter 3 Register (GRC Reset)	23:16	Host RW	Any	Keep tracks of the number of GRC Reset.
Reset Counter 2 Register (Perst Reset)	15:8	Host RW	Any	Keep tracks of the number of Perst events.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reset Counter 1 Register (LinkDown Reset)	7:0	Host RW	Any	Keep tracks of the number of LinkDown Reset events.

Register Base Register (offset: 0x78) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:18	RO	0	–
Register Base Register	17:2	RW	X	Local controller memory address of a register than can be written or read by writing to the register data register
Reserved	1:0	RO	0	–

Memory Base Register (offset: 0x7C) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
Memory Base Register	23:2	RW	X	Local controller memory address of the NIC memory region that can be accessed via Memory Window data register
Reserved	1:0	RO	0	–

Register Data Register (offset: 0x80) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Register Data Register	31:0	RW	X	Register Data at the location pointed by the Register Base Address register

Memory Data Register (offset: 0x84) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Memory Base Register	31:0	RW	X	Memory value at the location pointed by the Memory Window Base Address register

UNDI Receive Return Ring Consumer Index Register (offset: 0x88–0x8C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	63:11	RO	0	Unused
UNDI Receive Return C_Idx	10:0	RW	0	UNDI Receive Return Ring Consumer Index Mailbox. This register is mapped to the old mailbox register at offset 0xA0 and 0xA4 which has an internal Mailbox address of 19'b100000 and 19'b100001.

UNDI Send BD Producer Index Mailbox Register (offset: 0x90–0x94)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	63:9	RO	0	Unused
UNDI Send BD NIC P_Idx	8:0	RW	0	UNDI Send BD NIC Producer Index Mailbox. This register is mapped to the old mailbox register at offset 0xA8 and 0xAC which has an internal Mailbox address of 19'b1100000 and 19'b1100001.

UNDI Receive BD Standard Producer Ring Producer Index Mailbox Register (offset: 0x98–0x9C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	63:9	RO	0	Unused
UNDI Receive BD Standard Ring Producer Index	8:0	RW	0	UNDI Receive BD Std. Ring Producer Index Mailbox. This register has an internal Mailbox address of 19'b11010 and 19'b11011.

MSIX Capability Register (offset: 0xA0) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
MSIX_ENABLE	31	RW	0	If 1, and the MSI enable bit in the MSI message control register is 0, the function is permitted to use MSIX request service and profited from using INTx# messages.

Name	*Bits*	*Access*	*Default Value*	*Description*
FUNC_MASK	30	RW	0	If 1, all of the vectors associated with the function are masked regardless of their per vector Mask bit.
RESERVED	29-27	RO	0	Reserved
TABLE_SIZE	26-16	RO	0	System sw reads this field to determine the MSI-X table size N, which is encoded as N-1
MSIX_NEXT_CAP_ PTR	15-8	RO	0xAC (PCIe Cap)	This value continues the PCI capability chain. It's value specified an offset in the PCI address space of the next capability. The read-only value of this register is controlled by the CAP_ENA register in the PCI register space.
MSIX_CAP_ID	7-0	RO	0x11	Capability ID for MSIX

MSIX_tbl_off_bir Register (offset: 0xA4) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
TABLE_OFFSET	31-3	RO	0	–
TABLE_BIR	2-0	RO	0	Indicates which one of functions BAR is used to map MSI-X table into memory space.

MSIX_PBA_BIR_OFF Register (offset: 0xA8) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
PBA_OFFSET	31-3	RO	0	–
PBA_BIR	2-0	RO	0	Indicates which one of functions BAR is used to map MSI-X PBA into memory space.

PCIe Capabilities Register (offset: 0xAC) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Unused
Message Number	29-25	FW-RW RO	0	Interrupt Message Number:indicate which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this capability structure. For MSI, the value in this register indicates the offset between the base Message Data and the interrupt message that is generated. For MSI-X, the value in this register indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the function implements more than 32 entries.
Slot implemented	24	RO	0	Hardwired to 0 because this is an endpoint device. Slot Implemented. This register is not supported.
Device/Port Type	23-20	RO	0	Hardwired to 0 because this is an endpoint device. Device/Port Type. Device is an End Point.
Capability Version	19-16	RO	2	Capability Version. PCI Express Capability structure version number. These bits are hardwired to 2h.
PCIE_NEXT_CAP_PTR	15-8	RO	0	This registers contains the pointer to the next PCI capability structure.
PCIE_CAP_ID	7-0	RO	0x10	This register contains the PCI Express Capability ID.

Device Capabilities Register (offset: 0xB0) — Function 0

This register defines operational characteristics that are globally applicable to this device. This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
FLR_CAP_ SUPPORTED	28	RO	0	FLR capability is advertised when flr_supported bit in private device_capability register space is set.
Captured Slot Power Limit Scale	27:26	RO	0	This value specifies the scale used for the Power Limit: 00 = 1.0x 01 = 0.1x 10 = 0.01x 11 = 0.001x Specifies the scale used for the Slot Power Limit Value. It is set by the Set_Slot_Power_Limit Message. This field is not set for Root ports.
Captured Slot Power Limit Value	25:18	RO	0	This value specifies the upper limit on the power supplied for this device. Specifies the upper limit on power supplied by slot. It is set by the Set_Slot_Power_Limit Message. This field is not set for Root ports.
Reserved	17:16	RO	0	Unused
Role Based Error Support	15	RO	1	When set to 1, this value indicates that a role based error is supported. Indicate device is conforming to the ECN, PCI Express Base Specification, Revision 1.1., or subsequent PCI Express Base Specification revisions
Reserved	14-12	RO	0	Unused
Endpoint L1 Acceptable Latency	11:9	FW-RW	6h	This value returns the latency that this device can accept when transitioning from the L1 to the L0 state: 000 = less than 1 us 001 = 1 us to less than 2 us 010 = 2 us to less than 4 us 011 = 4 us to less than 8 us 100 = 8 us to less than 16 us 101 = 16 us to less than 32 us 110 = 32 us to 64 us 111 = Greater than 64 us Endpoint L1 Acceptable Latency. These bits are programmable through register space. The bits should be 0 for Root ports.

Name	*Bits*	*Access*	*Default Value*	*Description*
Endpoint L0s Acceptable Latency	8:6	FW-RW	6h	This value returns the total latency that this device can accept when transitioning from the L0s to L0 state: 000 = less than 64 ns 001 = 64 ns to less than 128 ns 010 = 128 ns to less than 256 ns 011 = 256 ns to less than 512 ns 100 = 512 ns to less than 1 us 101 = 1 us to less than 2 us 110 = 2 us to 4 us 111 = Greater than 4 us Endpoint L0s Acceptable Latency. These bits are programmable through register space. The value should be 0 for root ports.
Extended Tag Field Supported	5	RO	0	This value returns the maximum supported tag field size when this function acts as a requester. 0 = 5-bit tag field 1 = 8-bit tag field We do support extended tag. Extended Tag Field Support. This bit is programmable through register space. This capability is not currently supported.
Reserved	4:3	RO	0	Unused
Max Payload Size Supported	2:0	FW-RW	0	This value returns the maximum data payload size (in bytes) that this function supports for TLPs: 0 = 128 1 = 256 2 = 512 3 = 1024 4 = 2048 5 = 4096 6, 7 = Reserved Max Payload Size Supported. These bits are programmable from the register space and default value is based on define in version.v file.

Device Status Control Register (offset: 0xB4) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-22	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Transaction Pending	21	RO	0	When this bit is set to 1, it indicates that this device has issued nonposted request packets which have not been completed. This is bit is read back a 1, whenever a nonposted request initiated by PCIe core is pending to be completed.
Aux Power Detected	20	RO	0	When this bit is set to 1, it indicates that Aux power has been detected. This bit is the current state of the VAUX_PRSNT pin of the device. When it is 1, it is indicating that part needs VAUX and detects the VAUX is present.
Unsupported Request Detected	19	W2C	0	When this bit is set to 1, it indicates that an Unsupported Request has been received. Unsupported Request Detected.
Fatal Error Detected	18	W2C	0	When this bit is set to 1, it indicates that a Fatal Error has been detected. Fatal error detected.
Non-fatal Error Detected	17	W2C	0	When this bit is set to 1, it indicates that a nonfatal error has been detected. Nonfatal error detected.
Correctable Error Detected	16	W2C	0	When this bit is set to 1, it indicates that a correctable error has been detected. Correctable error detected.
FLR_INITIATED	15	RW	0	Initiate Function Level reset. This bit is writeable only if flr_supported bit in private device_capability register is set. A write of 1 to this bit initiates Function Level Reset. The value read by s/w from this bit is always 0.
Max Read Request Size	14:12	RW	2	This value controls the maximum read requests size for this device when acting as the requester: 0 = 128 1 = 256 2= 512 3 = 1024 4 = 2048 5 = 4096 6, 7, = Reserved Maximum Read Request Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions.
Enable No Snoop	11	RW	1	When this bit is set, the memory accessed by this device will not be cached by the processor. Enable No Snoop. When this bit is set to 1, PCIe initiates a read request with the No Snoop bit in the attribute field set for the transactions that request the No Snoop attribute.
Aux Power PM Enable	10	RO	1	When this bit is set, this device is enabled to draw aux power independent of PME power. This bit when set enables device to draw aux power independent of PME AUX power
RESERVED	9	RO	0	Unused

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Tag Field Enable	8	RW	0	When this bit is set, it enables this device to use an 8-bit Tag field as a requester. This capability is not supported.
Max Payload Size	7:5	RW	0	This value sets the maximum TLP data payload size (in bytes) for this device: 0 = 128 1 = 256 2 = 512 3 = 1024 4 = 2048 5 = 4096 6, 7, = Reserved Max Payload Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions.
Enabled Relaxed Ordering	4	RW	1	When this bit is set, this device is permitted to set the Relaxed Ordering bit. Relax Ordering Enable.
Unsupported Request Reporting Enable	3	RW	0	When this bit is set, Unsupported Request reporting is enabled. Unsupported request reporting enable.
Fatal Error Reporting Enabled	2	RW	0	When this bit is set, Fatal Error reporting is enabled. Fatal Error Reporting Enable.
Non-fatal Error Reporting Enabled	1	RW	0	When this bit is set, nonfatal error reporting is enabled. Nonfatal Error reporting enable.
Correctable Error Reporting Enabled	0	RW	0	When this bit is set, Correctable Error reporting is enabled. Correctable Error reporting enable.

Link Capability Register (offset: 0xB8) — Function 0

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Port Number	31:24	RO	0:HWinit	This value indicates the port number associated with this. PCIe Port Number. These bits are programmable through register.
Reserved	23:22	RO	0	–
LINK_BW_NOTIFY	21	RO	0	Link Bandwidth Notification Capability: RC: A value of 1b indicates support for the Link Bandwidth Notification status and interrupt mechanisms. This capability is required for all Root Ports and Switch Downstream Ports supporting Links wider than x1 and/or multiple Link speeds. RC: Field is implemented. EP: Not supported and hardwired to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
DL_ACTIVE_REP	20	RO	0	Data Link Layer Link Active Reporting Capable: RC: this bit must be hardwired to 1b if the component supports the optional capability of reporting the DL_Active state of the Data Link Control and Management State Machine. RC: Implemented (RW) for RC. Default to 0. EP: Not supported and hardwired to 0.
SUR_DWN_ERR_REP	19	RO	0	Surprise Down Error Reporting Capable: RC: this bit must be set if the component supports the optional capability of detecting and reporting a Surprise Down error condition. RC: Not supported and hardwired to 0. EP: Not supported and hardwired to 0.
Clock Power Management	18	Host RO FW R/W	1	1 = clkreq capable 0 = clkreq not capable If it's mobile bonding, the default value will be 1; otherwise, 0.
L1 Exit Latency	17:15	RO FW-RW	2	This value returns the L1 exit latency for this link: 0 = Maximum of 1 us 1 = Maximum of 2 us 2 = Maximum of 4 us 3 = Maximum of 8 us 4 = Maximum of 16 us 5 = Maximum of 32 us 6 = Maximum of 64 us 7 = No limit L1 Exit Latency. These bits are programmable through register space. Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following.
L0s Exit Latency	14:12	RO FW-RW	4	This value returns the L0s exit latency for this link: 0 = less than 64 ns 1 = less than 128 ns 2 = less than 256 ns 3 = less than 512 ns 4 = less than 1 us 5 = less than 2 us 6 = less than 4 us 7 = greater than 4 us L0s Exit Latency. These bits are programmable through register space. Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following.

Name	*Bits*	*Access*	*Default Value*	*Description*
Active State power management support	11:10	RO FW-RW	3	This value returns the supported ASPM states; 0 = reserved 1 = L0s supported 2 = reserved 3 = L0s and L1 supported ASPM Support. These bits are programmable through reg space.
Maximum Link Width	9:4	RO FW-RW	1	This value returns the maximum link width. Allowable values are 1, 2, 4, 8, 12, 16, and 32 only. All other values are reserved. Value Name Description 1 1 One Lane Max 2 2 Two Lanes Max 4 4 Four Lanes Max 8 8 Eight Lanes Max Maximum Link Width. These are programmable through reg space.Bit 9 is always 0 and is not programmable. Default value is based on numLanes field in version.v
Maximum Link Speed	3:0	RO FW-RW	1	This value returns the Maximum Link Speed. 1 = 2.5Gbps. All other values reserved. Value Name Description 1 2_5 2.5 Gbps Max 2 5 5 Gbps Max Value used by internal logic is the smaller of the value programmed for each function.

Link Status_Control Register (offset: 0xBC) — Function 0

This register is reset by Hard Reset or the rising edge of PERST_L.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-30	RO	0	Unused
DL_ACTIVE	29	RO	0	Data Link Layer Link Active: returns a 1b to indicate the DL_Active state, 0b otherwise. Not implemented and hardwire to 0.
Slot Clock Configuration	28	RO	1	This value indicates that this device uses the same physical reference clock that the platform provides on the connector. Slot Clock configuration. This bit is read-only by host, but read/write via backdoor CS bus.
LINK_TRAINING	27	RO	0	EP: This bit is N/A and is hardwired to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	26	RO	0	Unused
Negotiated Link Width	25-20	RO	0	This value returns the negotiated link width. The only valid values are 1, 2, 4, 8, 12, 16, 32
Negotiated Link Speed	19-16	RO	0	This value returns the negotiated link speed. 1 = 2.5 Gbps. Link Speed. These bits indicate the negotiated link speed of the PCI Express link.
Reserved	15:12	RO	0	Unused
LINK_BW_INT_EN	11	RO	0	Link Autonomous Bandwidth Interrupt Enable: When Set, this bit enables the generation of an interrupt to indicate that the Link Autonomous Bandwidth Status bit has been Set. RC: Not implemented and hardwired to 0. EP: N/A and hardwired to 0.
LINK_BW_MGMT_INT_EN	10	RO	0	Link Bandwidth Management Interrupt Enable: when Set, this bit enables the generation of an interrupt to indicate that the Link Bandwidth Management Status bit has been Set. RC: N/A and hardwired to 0. EP: Not implemented and hardwired to 0.
HW_AUTO_WIDTH_DIS	9	RO	0	Hardware Autonomous Width Disable: When Set, this bit disables hardware from changing the Link width for reasons other than attempting to correct unreliable Link operation by reducing Link width. Other functions are reserved. RC: Not applicable and hardwire to 0 EP: If supported, only apply to function0. Not implemented and hardwire to 0.
Clock Request Enable	8	RW	0	1 = clkreq is enabled 0 = clkreq is disabled Enable Clock Power Management: RC: N/A and hardwired to 0. EP: When this bit is set, the device is permitted to use CLKREQ# signal to power management. Feature is enabled through version.v define.
Extended Synch	7	RW	0	When this bit is set, it forces extended sync which gives external devices (such as logic analyzers) additional time to achieve bit and symbol lock. Extended Synch. This bit when set forces the transmission of 4096 FTS ordered sets in the L0s state followed by a single SKP ordered set prior to entering the L0 state, and the transmission of 1024 TS1 ordered sets in the L1 state prior to entering the Recovery state. Value used by logic is resolved to 1 if either function has this bit set.
Common Clock Configuration	6	RW	0	When this bit is set, it indicates that the link partners are using a common reference clock. Common Clock Configuration. Value used by logic is resolved to 1 only if both functions (when enabled) have this bit set.
Reserved (Retrain Link)	5	RO	0	The device does not support this feature. Requesting PHY to retrain the link. This bit is only applicable to RC. So for EP it is read only bit.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved (Link Disable)	4	RO	0	The device does not support this feature. Requesting PHY to disable the link. This bit is only applicable to RC. So for EP it is read only bit.
Read Completion Boundary	3	RW	0	This value indicates the Read Completion Boundary value (in bytes) of the upstream root port: 0 = 64 1 = 128
Reserved	2	RO	0	Unused
Active State Power Management Control	1:0	RW	0	This value control the Active State power management supported on this link: 0 = Disabled 1 = L0s entry enabled 2 = L1 entry enabled 3 = L0s and L1 entry enabled ASPM Control. Value used by logic is dependent on the value of this bit for each enabled function and also on the programmed powerstate of each function.

Slot Capability Register (offset: 0xC0) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
PHYSICAL_SLOT_NUMBER	31-19	RO	0	Not Implemented
RESERVED	18-17	RO	0	Unused
SLOT_POWER_LIMIT_SCALE	16-15	RO	0	Not Implemented
SLOT_POWER_LIMIT_VALUE	14-7	RO	0	Not Implemented
RESERVED	6-0	RO	0	Not Implemented

Slot Control_Status Register (offset: 0xC4) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
SLOT_STATUS	31-23	RO	0	Not Implemented
PRESENCE_DETECT	22	RO	0	Not Implemented
RESERVED	21-16	RO	0	Not Implemented
SLOT_CONTROL	15-0	RO	0	Not Implemented

Root_Capability Control Register (offset: 0xC8) — Function 0

For EP, this register is not applicable and hardwired to 0.

Root_Status Register (offset: 0xCC) — Function 0

For EP this register is not applicable and hardwired to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31-0	RO	0	Not Implemented

Device Capability 2 Register (offset: 0xD0) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31-12	RO	0	Unused
LTR_MECHANISM_ SUPPORTED	11	RO	0	Latency Tolerance Reporting Mechanism Supported, Programmable through register space. This field will read 1, when bit 5 of ext_cap_ena field in private register space is set.
RESERVED	10-5	RO	0	Unused
CMPL_TIMEOUT_DISABL_ SUPPORTED	4	RO	1	Completion Timeout Disable Supported, Programmable through register space
CMPL_TIMEOUT_RANGES _SUPPORTED	3-0	RO	F	Completion Timeout Ranges Supported. Programmable through register space. Value Name Description 15 ABCD Ranges A,B,C, and D

Device Status_Control 2 Register (offset: 0xD4) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
DEVICE_STATUS_2	31-16	RO	0	Placeholder for Gen2
RESERVED	15-11	RO	0	Unused
LTR_MECHANISM _ENABLE	10	RW	0	Latency Tolerance Reporting Mechanism Enable, This field is writeable, when bit 5 of ext_cap_ena field in private register space is set. This bit is RW only in function 0 and is RsvdP for all other functions.

Name	Bits	Access	Default Value	Description
IDO_CPL_ENABLE	9	RW	0	IDO Completion Enable, This field is writeable, when bit ido_supported bit of private device_capability_2 register is set. When this bit is set, function is permitted to set ID based Ordering Attribute of Completions it returns.
IDO_REQ_ENABLE	8	RW	0	IDO Request Enable, This field is writeable, when bit ido_supported bit of private device_capability_2 register is set. When this bit is set, function is permitted to set ID based Ordering Attribute of Requests it initiates.
RESERVED	7-5	RO	0	Unused
CMPL_TIMEOUT_DISABLE	4	RW	0	Completion Timeout Disable
CMPL_TIMEOUT_VALUE	3-0	RW	0	Completion timeout value. The spec specifies a range, the device uses the max value in the range. Value Name Description 0 50MS 50 ms 1 100US 100 us 2 10MS 10 ms 3 55MS 55 ms 4 210MS 210 ms 5 900MS 900 ms 6 3_5S 3.5s 7 13S 13s 8 64S 64s

Link Capability 2 Register (offset: 0xD8) — Function 0

Name	Bits	Access	Default Value	Description
LINK_CAPABILITY_2	31-0	RO	0	Placeholder for Gen2

Link Status_Control 2 Register (offset: 0xDC) — Function 0

This register will be Read only by default, and will read all 0's to allow compliance with PCIe spec 1.1. To enable this register, reset comply_pcie_1_1 bit in the register space to 0.

Name	Bits	Access	Defaul t Value	Description
LINK_STATUS_2	31-17	RO	0	Placeholder for Gen2

Name	*Bits*	*Access*	*Default Value*	*Description*
CURR_DEEMPH_LEVEL	16	RO	0	curr_deemph_level
RESERVED	15-13	RO	0	Unused
CFG_COMPLIANCE_DEEMPH	12	RW	0	Compliance deemphasis.
CFG_COMPLIANCE_SOS	11	RW	0	Compliance SOS.
CFG_ENTER_MOD_COMPLIANCE	10	RW	0	Enter Modified Compliance.
CFG_TX_MARGIN	9-7	RW	0	Controls the value of non deemphasized voltage level at the TX pins. Value used by logic is resolved to the smaller binary value, if two functions have different values. 0 000 800-1200 mV for full swing and 400-600 mV for half swing 1 001 Values are monotonic with non-zero slope 2 010 Values are monotonic with non-zero slope 3 011 200-400 mV for full swing and 100-200 mV for half swing 4 100 Reserved 5 101 Reserved 6 110 Reserved 7 111 Reserved
SEL_DEEMPHASIS	6	RW	0	When link is operating at Gen2 rates, this bit selects the level of deemphasis. Value used by logic is resolved to 1 if either function has this bit set. 0 0 –6 dB 1 1 –3.5 dB
HW_AUTO_SPEED_DISABLE	5	RO	0	Not Supported and hardwired to 0.
ENTER_COMPLIANCE	4	RW	0	S/W instructs link to enter compliance mode. Value used by internal logic is set when either function has this bit enabled.
TARGET_LINK_SPEED	3-0	RW	0	Upper limit of link speed: 0 2_5 2.5 Gbps 1 5_0 5.0 Gbps

Slot Capability 2 Register (offset: 0xE0) — Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
SLOT_CAPABILITY_2	31-0	RO	0	Not Implemented

Slot Status_Control 2 Register (offset: 0xE4) — Function 0

Name	Bits	Access	Default Value	Description
SLOT_STATUS_2	31-16	RO	0	Not Implemented
SLOT_CONTROL_2	15-0	RO	0	Not Implemented

Product ID and ASIC revision (offset: 0xF4) — Function 0

Name	Bits	Access	Default Value	Description
Product ID and ASIC REV	31:0	RO	PRODUCT ID and ASIC REV ports	Product ID and ASIC revision read only register 31:12: Product ID: 57767 11:8: Base Layer Revision Info 7:0: Metal Layer Revision Info

APE Memory Indirect Address Register (offset: 0xF8)

Name	Bits	Access	Default Value	Description
APE Memory Indirect Address	31:0	RW	0	–

APE Memory Indirect Data Register (offset: 0xFC)

Name	Bits	Access	Default Value	Description
APE Memory Indirect Data	31:0	RW	0	–

Advanced Error Reporting Enhanced Capability Header (offset: 0x100) — Functions (3:0)

Name	Bits	Access	Default Value	Description
Next Capability Offset	31:20	RO	0x13Ch	Next Capabilities Pointer is 0x13C which is Power Budget.
PCI Express Extended Capability ID	15:0	RO	0x0001	PCI Express Extended Capability ID. These bits are hardwired to 0001h indicating the presence of PCI Express Advanced Error Capability.

Uncorrectable Error Status Register (offset: 0x104) — Functions (3:0)

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
Unsupported Request Error Status	20	RW1CS	0	This bit is set when an unsupported request error occurs
ECRC Error Status	19	RW1CS	0	This bit is set when an ECRC error occurs
Malformed TLP Status	18	RW1CS	0	This bit is set when a Malformed TLP error occurs
Receiver Overflow Status	17	RW1CS	0	This bit is set when a Receiver Overflow error occurs
Unexpected Completion Status	16	RW1CS	0	This bit is set when an Unexpected Completion error occurs
Completer Abort Status	15	RW1CS	0	This bit is set when a completer Abort error occurs
Completion Timeout Status	14	RW1CS	0	This bit is set when completion timeout error occurs
Flow control Protocol Error Status	13	RW1CS	0	This bit is set when a Flow control protocol error occurs
Poisoned TLP Status	12	RW1CS	0	This bit is set when a Poisoned TLP error occurs
Reserved	11:5	RO	0	–
Data Link Protocol Error Status	4	RW1CS	0	This bit is set when a Data Link Protocol error occurs
Reserved	3:0	RO	0	–
Training Error Status	0	RW1CS	0	This bit is set when a training error occurs

Uncorrectable Error Mask Register (offset: 0x108) — Functions (3:0)

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
Unsupported Request Error Mask	20	RWS	0	Setting this bit will mask Unsupported Request Error
ECRC Error Mask	19	RWS	0	Setting this bit will mask ECRC error
Malformed TLP Mask	18	RWS	0	Setting this bit will mask Malformed TLP error
Receiver Overflow Mask	17	RWS	0	Setting this bit will mask Receiver overflow error

Name	*Bits*	*Access*	*Default Value*	*Description*
Unexpected Completion Mask	16	RWS	0	Setting this bit will mask unexpected completion error
Completer Abort Mask	15	RWS	0	Setting this bit will mask completer abort error
Completion Timeout Mask	14	RWS	0	Setting this bit will mask completion timeout error
Flow Control Protocol Error Mask	13	RWS	0	Setting this bit will mask flow control protocol error
Poisoned TLP Mask	12	RWS	0	Setting this bit will mask poisoned TLP error
Reserved	11:5	RO	0	–
Data Link Protocol Error Mask	4	RWS	0	Setting this bit will mask data link protocol error
Reserved	3:1	RO	0	–
Training Error Mask	0	RWS	0	Setting this bit will mask training error

Uncorrectable Error Severity Register (offset: 0x10C) — Functions (3:0)

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
Unsupported Request Error Severity	20	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal
ECRC Error Severity	19	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal
Malformed TLP Severity	18	RWS	1	This bit controls the severity 0 = nonfatal 1 = fatal
Receiver Overflow Error Severity	17	RWS	1	This bit controls the severity 0 = nonfatal 1 = fatal
Unexpected completion Error Severity	16	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal
Completer Abort Error Severity	15	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal

Name	*Bits*	*Access*	*Default Value*	*Description*
Completion Timeout Error Severity	14	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal
Flow control Protocol Error Severity	13	RWS	1	This bit controls the severity 0 = nonfatal 1 = fatal
Poisoned TLP Severity	12	RWS	0	This bit controls the severity 0 = nonfatal 1 = fatal
Reserved	11:4	RO	0	–
Surprise down error severity	5	RO	1	Pcie 1.1 spec page 409
Data Link Protocol Error Severity	4	RWS	1	This bit controls the severity 0 = nonfatal 1 = fatal
Reserved	3:1	RO	0	–
Training Error Severity	0	RWS	1	This bit controls the severity 0 = nonfatal 1 = fatal

Correctable Error Status Register (offset: 0x110) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RO	0	–
Advisory Non-Fatal Error Status	13	RO	0	This bit is set when an advisory nonfatal error occurs
Replay Timer Timeout Status	12	RW1CS	0	This bit is set when a replay timer timeout error occurs
Reserved	11:9	RO	0	–
REPLAY_NUM Rollover Status	8	RW1CS	0	This bit is set when a REPLAY_NUM Rollover error occurs
Bad DLLP Status	7	RW1CS	0	This bit is set when a Bad DLLP error occurs
Bad TLP Status	6	RW1CS	0	This bit is set when a Bad TLP error occurs
Reserved	5:1	RO	0	–
Receiver Error Status	0	RW1CS	0	This bit is set when a Receiver error occurs

Correctable Error Mask Register (offset: 0x114) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RO	0	–
Advisory Non-Fatal Error Mask	13	RWS	1	Setting this bit will mask advisory nonfatal errors
Replay Timer Timeout Mask	12	RWS	0	Setting this bit will mask replay timer timeout errors
Reserved	11:9	RO	0	–
REPLAY_NUM Rollover Mask	8	RWS	0	Setting this bit will mask REPLAY_NUM rollover errors
Bad DLLP Mask	7	RWS	0	Setting this bit will mask Bad DLLP errors
Bad TLP Mask	6	RWS	0	Setting this bit will mask Bad TLP errors
Reserved	5:1	RO	0	–
Receiver Error Mask	0	RWS	0	Setting this bit will mask receiver errors

Advanced Error Capabilities and Control Register (offset: 0x118) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
ECRC Check Enable	8	RWS	0	Setting this bit will enable ECRC checking
ECRC Check Capable	7	RO	1	When this bit is set, it indicates that this device supports ECRC checking
ECRC Generation Enable	6	RWS	0	Setting this bit will enable ECRC generation
ECRC Generation Capable	5	RO	1	When this bit is set, it indicates that this device supports ECRC generation
First Error Pointer	4:0	ROS	0	This value indicates the bit position within the "Uncorrectable Error Status Register" 0x104

Header Log Register (offset: 0x11C) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Header Byte 0	31:24	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 1	23:16	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 2	15:8	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 3	7:0	ROS	–	The TLP header of the transaction that has incurred a failure

Header Log Register (offset: 0x120) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Header Byte 4	31:24	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 5	23:16	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 6	15:8	ROS	–	The TLP header of the transaction that has incurred a failure

Name	*Bits*	*Access*	*Default Value*	*Description*
Header Byte 7	7:0	ROS	–	The TLP header of the transaction that has incurred a failure

Header Log Register (offset: 0x124) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Header Byte 8	31:24	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 9	23:16	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 10	15:8	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 11	7:0	ROS	–	The TLP header of the transaction that has incurred a failure

Header Log Register (offset: 0x128) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Header Byte 12	31:24	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 13	23:16	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 14	15:8	ROS	–	The TLP header of the transaction that has incurred a failure
Header Byte 15	7:0	ROS	–	The TLP header of the transaction that has incurred a failure

Root_Error_Command Register (Offset 0x12C) – Functions (3:0)

For EP this register is not applicable and hardwired to 0.

Root_Error_Status Register (Offset 0x130) – Functions (3:0)

For EP this register is not applicable and hardwired to 0.

Root_Error_ID Register (Offset 0x134) – Functions (3:0)

For EP this register is not applicable and hardwired to 0.

Device Serial No Enhanced Capability Header Register (offset: 0x13C) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Next Capability Offset	31:20	RO	0x150 (Power Budget)	Next Capabilities Pointer. When bit 6 of register 7c04 is 1 else 0x0. Bit 6 of register 7c04 is 1 by default. (PCIe Pwr Budget Cap Enable)
Revision ID	19:16	RO	0x1	0x1 for PCI Express
PCIE Capability ID	15:0	RO	0x0003	0x3 for PCIe Device Serial Number Capability ID

Device Serial No Lower DW Register (offset: 0x140) — Functions (3:0)

The register is reset by PCIe Reset. This register has the PCIe Device Serial Number bits [31:0]. This register will contain the data written in the Device Serial Number Access Lower Register (Offset 504h).

Name	*Bits*	*Access*	*Default Value*	*Description*
reserved	31:0	Host RO/ FW RW (bit 23 of 7c04 1) else RO	0xFE	0xFE when bit 23 of 7c04 is clear.
Lower MAC Address	23:0	Host RO/ FW RW (bit 23 of 7c04 1) else RO	0xFFFFFF	MAC Address(23:0) when bit 23 of 7c04 is clear.

Device Serial No Upper DW Register (offset: 0x144) — Functions (3:0)

The register is reset by PCIe Reset. This register has the PCIe Device Serial Number bits [63:32]. This register will contain the data written in the Device Serial Number Access Upper Register (Offset 508h).

Name	*Bits*	*Access*	*Default Value*	*Description*
Upper MAC Address	31:8	Host RO/ FW RW (bit 23 of 7c04 1) else RO	0xFFFFFF	MAC Address(47:24) when bit 23 of 7c04 is clear
reserved	7:0	Host RO/ FW RW (bit 23 of 7c04 1) else RO	0xFF	0xFF when bit 23 of 7c04 is clear.

Power Budgeting Enhanced Capability Header Register (offset: 0x150) — Functions (3:0)

The register is hardwired.

Name	*Bits*	*Access*	*Default Value*	*Description*
Next Capability Offset	31:20	RO	0x160 (VC)	This value continues the PCI capability chain. It's value specified an offset in the PCI address space of the next capability. The read-only value of this register is controlled by the EXT_CAP_ENA register in the PCI register space.

Name	*Bits*	*Access*	*Default Value*	*Description*
Revision ID	19:16	RO	0x1	0x1 for PCI Express
PCIe Power Budget Capability ID	15:0	RO	0x0004	0x4 for PCIe Power Budget Capability ID

Power Budgeting Data Select Register (offset: 0x154) — Functions (3:0)

The register is reset by PCIe Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0x000000	–
Data Select	7:0	RW	0x00	Index Power Budgeting Data reported through the Data Register. This value selects the value visible in the pb_dr. 0 0 Selects pb_cap value from 0x510[31:0]. 1 1 Selects pb_cap value from 0x514[31:0]. 2 2 Selects pb_cap value from 0x518[31:0]. 3 3 Selects pb_cap value from 0x51c[31:0]. 4 4 Selects pb_cap value from 0x520[31:0]. 5 5 Selects pb_cap value from 0x524[31:0]. 6 6 Selects pb_cap value from 0x528[31:0]. 7 7 Selects pb_cap value from 0x52c[31:0].

Power Budgeting Data Register (offset: 0x158) — Functions (3:0)

The register is reset by PCIe Reset. This register provides the power budgeting data for the entry number specified by the pwr_bdgt_data_sel register. The data present in this register is selected from one of the POWER BUDGET DATA ACCESS Registers from offset 510h through 52Ch, based on the value written in Power Budget Data Select register. The field definitions for each selected value are the same. P

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0x0	–
Power Rail	20:18	FW-RW Host-RO	0b000	Specifies the power rail of the operating condition 12V (000) 3.3V (001) 1.8V (010) Thermal (111) Bit20 is hardwired to 0 because we don't support Thermal; Bit 19:18 are programmable via Firmware.
Type	17:15	FW-RW Host-RO	0b000	Specifies the type of the operating condition PME Aux (000) Auxiliary (001) Idle (010) Sustained (011) Maximum (111)
PM State	14:13	FW-RW Host-RO	0b00	Specifies the power management state of operating condition: D0, D3
PM Sub State	12:10	RO	0b000	Specifies the sub states of the operating condition
Data Scale	9:8	FW-RW Host-RO	0b00	Specifies the scale to apply to the base power value; 0x1 for 0.1x scale.
Base Power	7:0	FW-RW Host-RO	0x00	Specifies in Watts the base power value in a given operating conditions. 2.2Watts in D0Max 1.1Watts in D3-Hot 0.54Watts in D3-Cold

Power Budgeting Capability Register (offset: 0x15C) — Functions (3:0)

Name	Bits	Access	Default Value	Description
RESERVED	31:1	RO	0x000000	Unused
LOM Configuration	0	FW-RW Host-RO	1	Indicate that the power budget for the device is included within the system power budget. Derived from NVRAM configuration If Configured as LOM, then write 1 to bit 5 of 0x7C04 else write 0. The "System Allocated" bit when set indicates that the power budget for the device is included within the system power budget. Reported Power Budgeting Data for this device should be ignored by software for power budgeting decisions if this bit is set. This register is Read Only. The value can be written indirectly by writing into Power Budget Capability Register (0x550[0])

Virtual Channel Enhanced Capability Header (offset: 0x160 — Functions (3:0)

The read-back value of this register is controlled by the EXT_CAP_ENA register in the PCI register space)

Name	Bits	Access	Default Value	Description
Next Capability Offset	31:20	RO	0x1b0	This value continues the PCI capability chain. It's value specified an offset in the PCI address space of the next capability.
Capability Version	19:16	RO	0x1	Capability ID Version. These bits are hardwired to 1h indicating the version of the capability ID. Hardwire to 1.
PCI Express Extended Capability ID	15:0	RO	0x0002	Extended Capability ID for the Virtual Channel Capability is 0002h

Port VC Capability Register (offset: 0x164) — Functions (3:0)

Name	Bits	Access	Default Value	Description
PORT_VC_CAPABILITY	31:0	RO	0	Not implemented.

Port VC Capability Register 2 (offset: 0x168) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
PORT_VC_CAPABILITY2	31:0	RO	0	Not implemented.

Port VC Status_Control Register (offset: 0x16C) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
PORT_VC_STATUS	31:16	RO	0	Not implemented.
PORT_VC_CONTROL	15:0	RO	0	Not implemented.

Port Arbitration Table Register (offset: 0x170) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Port Arbitration Table	31-0	RO	0	Not Implemented

VC Resource Control Register (offset: 0x174) — Functions (3:0)

The read-back value of this register is controlled by the EXT_CAP_ENA register in the PCI register space.)

Name	*Bits*	*Access*	*Default Value*	*Description*
VC Enable	31	RO	1	Enables virtual channel. This bit is hardwired to 1 for the default VC0 and writing to this field has no effect.
Reserved	30:8	RO	0	–
TC/VC Map	7:1	RW	0x7f	A 1 at bit n indicates that TC n is mapped to VC0 (bit 0 is read only and is hardwired to 1). This field indicates the TCs that are mapped to the VC resource. This field is valid for all devices. Note: Bit 0 of this field is read only. It is set to 1 for the default VC0.
Default_VC0	0	RO	1	–

VC Resource Status Register (offset: 0x178) — Functions (3:0)

Name	*Bits*	*Access*	*Default Value*	*Description*
VC_RSRC_STATUS	31-16	RO	0	Not Implemented
Reserved	15-0	RO	0	Unused

LTR Capability Register (offset: 0x1b0) — Functions (3:0)

The read-only value of this register is controlled by setting bit 5 of the EXT_CAP_ENA for EP, By default, this capability is disabled (i.e. reading this register will return zeroes). The capability can be enabled by default by defining LTR_ENABLED in version.v and setting bit 5 of EXT_CAP_ENA. This capability when present, will only exist in Functions (3:0) of a multifunction device.

Name	*Bits*	*Access*	*Default Value*	*Description*
NEXT	31-20	RO	0	This value continues the PCI capability chain. It's value specified an offset in the PCI address space of the next capability.
CAP_VER	19-16	RO	0x1	LTR Capability version. Hardwired to 0x1.
LTR_EXT_CAP_ID	15-0	RO	0x18	Vendor Specific Extended Capability ID.

Latency Register (offset: 0x1b4) — Functions (3:0)

The RW value of this register is controlled by setting bit 5 of the EXT_CAP_ENA for EP. By default, this capability is disabled (i.e. reading this register will return zeroes).

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31-29	RO	0	Unused
MAX_NO_SNOOP_ LATE_SCALE	28-26	RW	00	Max No Snoop Latency Scale. This register provides a scale for the value contained within the max_no_snoop_late_value field.
MAX_NO_SNOOP_ LATE_VALUE	25-16	RW	0	Max No Snoop Latency Value. Along with Max No snoop latency scale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platforms max supported latency or less.
RESERVED	15-13	RO	0	Unused
MAX_SNOOP_LATE_ SCALE	12-10	RW	0	Max Snoop Latency Scale. This register provides a scale for the value contained within the max_snoop_late_value field.
MAX_SNOOP_LATE_ VALUE	9-0	RW	0	Max Snoop Latency Value. Along with Max snoop latency scale field, this register specifies the maximum no-snoop latency that a device is permitted to request. Software should set this to the platforms max supported latency or less.

SD/MMC Card-Reader Configuration Registers — Function 1

The following describes the Function 1 PCIe Configuration Registers which are required by SD/MMC Card-Reader Host Controller. The primary reset for these registers is PCIe Reset. Registers that are labeled *not used in BCM5725/BCM5762/BCM5776* or *not applicable in BCM5725/BCM5762/BCM57767* will return all 0x0 when read.

Note: If Wake-n-Card (WoC) insertion/removal is desired to wake the system, it is necessary to program the following registers before the system enters a sleep state.

- Write to register Card Reader BAR + 0x28 with a value of 0x06000000 (This is setting bits 1 and 2 in the Wakeup Control register 0x2B in the standard card reader registers, which are not documented in this Programmer's Guide. Refer to public card reader standards documentation.)
- Set bit 8 of Card Reader PCIe configuration space register 0x4c to enable PME event generation from the card-reader function.

Note: The WoC Insertion/Removal Event is not enabled by default, which means that WoC is not available when the system is in the Out-of-Box state (A/C power cord first attached).

Device ID and Vendor ID Register (offset: 0x00) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Device ID	31:16	FW-RW Host-RO	–	Default for BCM5762/BCM57767 (SD/MMC Function 1): 0x16BC
Vendor ID	15:0	FW-RW Host-RO	0x14E4	–

Status and Command Register (offset: 0x04) – Function 1

This register is reset by PCIe Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Detected Parity Error	31	R/W2C	0x0	When this bit is set, it indicates that the function has received a poisoned TLP
Signaled System Error	30	R/W2C	0x0	This bit is set when a function sends an ERR_FATAL or ERR_NONFATAL message and the SERR enable bit in the command register is set

Name	*Bits*	*Access*	*Default Value*	*Description*
Received Master Abort	29	R/W2C	0x0	This bit is set when a requester receives a completion with UR completion status.
Received Target Abort	28	R/W2C	0x0	This bit is set when a requester receives a completion with completer abort completion status.
Signaled Target Abort	27	R/W2C	0x0	This bit is set when a function acting as a completer terminates a request by issuing Completer abort completion status to the requester
DEVSEL Timing	26:25	RO	0x0	Does not apply to PCIe
Master Data Parity Error	24	R/W2C	0x0	The master data parity error bit is set by a requester if the parity error enable bit is set in its command register and either of the following 2 conditions occur. If the requester receives a poisoned completion if the requester poisons a write request If the parity Error enable bit is cleared , the master data parity error status bit is never set.
Fast Back-to-back capable	23	RO	0x0	Does not apply to PCIe
Reserved	22	RO	0x0	These bits are reserved and tied low per the PCI specification.
66MHz Capable	21	RO	0x0	Does not apply to PCIe
Capabilities List	20	RO	0x1	This bit is tied high to indicate that the device supports a capability list. The list starts at address 0x40.
Interrupt Status	19	RO	0x0	Indicates this device generated an interrupt
Reserved	18:16	RO	0x0	These bits are reserved and tied low per the PCIe specification.
Reserved	15:11	RO	0x00	These bits are reserved and tied low per the PCIe specification.
Interrupt Disable	10	RW	0x0	When this bit is set, function is not permitted to generate IntX interrupt messages (deasserted) regardless of any internal chip logic. Setting this bit has no effect on the INT_STATUS bit below. Writing this bit to 0 will un-mask the interrupt and let it run normally.
Fast Back-to-back Enable	9	RO	0x0	Does not apply to PCIe
System Error Enable	8	RW	0x0	When set, this bit enables the non fatal and fatal errors detected by the function to be reported to the Root Complex. The function reports such errors to the Root Complex if it is enabled to do so either through this bit or though PCI express specific bits in DCR.
Stepping Control	7	RO	0x0	Does not apply to PCIe
Parity Error Enable	6	RW	0x0	This bit enables the write to the Master data parity error status bit. If this bit is cleared, the master data parity error status bit will never be set.
VGA Palette Snoop	5	RO	0x0	Does not apply to PCIe

Name	*Bits*	*Access*	*Default Value*	*Description*
Memory Write and Invalidate	4	RO	0x0	Does not apply to PCIe
Special Cycles	3	RO	0x0	Does not apply to PCIe
Bus Master	2	RW	0x0	This bit controls the enabling of the bus master activity by this device. When low, it disables an Endpoint function from issuing memory or IO requests. Also disables the ability to issue MSI messages.
Memory Space	1	RW	0x0	This bit controls the enabling of the memory space. When disabled, memory transactions targeting this device return completion with UR status.
I/O Space	0	RO	0x0	This bit indicates that the device does not support I/O space access because it is zero and can not be modified. IO transactions targeting this device return completion with UR status .

PCI Classcode and Revision ID Register (offset: 0x8) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Classcode	31:8	RO	0x080500	Default for (SD/MMC Function 1): 0x080500
Revision ID – All-layer Revision ID	7:4	FW-RW Host-RO	ASIC Rev Input	This field will be updated automatically by hardware based on the External All Layer Revision ID. For example, this field will contain a value of 0x0 after hard reset for BCM5762/BCM57767 A0 silicon. Software shall use this field only to display the Device Silicon Revision ID for application where the user/customer needs to know the Device Silicon Revision ID. One such application is the B57DIAG Device Banner. Furthermore, Software (Boot Code/Driver/B57DIAG) shall NOT use this field in determining Bug Fixes. It should only use the Internal Revision ID, bits 31:24 and bits 19:16 from Register 68, for that purpose. 0x0 for A Steps 0x1 for B Steps 0x2 for C Steps
Revision ID – Metal Revision ID	3:0	FW-RW Host-RO	ASIC Rev Input	This field will be updated automatically by hardware based on the Metal Revision ID. 0x0 for metal 0 step 0x1 for metal 1 step 0x2 for metal 2 step

BIST, Header Type, Latency Timer, Cache Line Size Register (offset: 0x0C) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
BIST	31:24	RO FW-RW	0x0	The 8-bit BIST register is used to initiate and report the results of any Built-In-Self-Test. This value can be written by firmware through the PCI register space BIST register to modify the read value to the host.
Header Type	23:16	RO	0x80	The 8-bit Header Type register identifies both the layout of bytes 10h through 3Fh of the Configuration space, as well as whether this adapter contains multiple functions. A value of 0x80 indicates a multi function device (Type 0) using the format specified in the PCI specification, while a value of 0x0 indicates a single function Type 0 device.
Latency Timer	15:8	RO	0x0	This register does not apply to PCI express and must be hardwired to zero
Cache Line Size	7:0	RO FW-RW	0x0	This field is implemented by PCIe device as a read/write field for legacy compatibility purposes.

Base Address Register 1 (offset: 0x10) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
Base Address (BAR_1)	31:8	RW	0x0	These bits set the address within a 32-bit address space that will be card will respond in. These bits may be combined with the bits in XBAR_2 to create a full 64 bit address decode. Only the bits that address blocks bigger than the setting in the BAR1_SIZE value are RW. All lower bits are RO with a value of zero. This value is sticky and only reset by HARD Reset.
Size indication	7:4	RO	0x0	RO bits indicate size of memory space. 7:4 RO=0 (256B BAR
Prefetchable	3	RO	0x0 Strap input to pcie block	This bit indicates that the area mapped by BAR_1 may be pre-fetched or cached by the system without side effects. Bit can be programmed from shadow register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Type	2:1	Host RO	0x0 Strap input to pcie block	These bits indicate that BAR_1 may be programmed to map this adapter to anywhere in the 64-bit address space. Encoded with the following values 00: located anywhere in 32-bit address space 01: reserved 10: located anywhere in 64-bit address space 11: reserved For function 1, this value is 0 (32-bit enabled)
Memory Space Indicator	0	RO	0x0	This bit indicates that BAR_1 maps a memory space and is always read as 0.

Base Address Register 2 (offset: 0x14) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Base Address (XBAR_2)	31:0	RW RO if bar1_64ena is 0'0'	0x0	The 32-bit XBAR_2 register programs the upper half of the base address for the memory space mapped by the card onto the PCI bus. These bits set the address upper 32-bit address space. These bits may be combined with the bits in BAR_1 to create a full 64 bit address decode. These bits must be set to zero for the card to respond to single address cycle requests. This value is sticky and only reset by HARD Reset.

Base Address Register 3 (offset: 0x18) – Function 1

The 32-bit BAR_3 register programs the 2nd base address for the memory space mapped by the card onto the PCI bus. This register can be combined with the Extended BAR_4 to make a 64-bit address for supporting Dual Address cycles systems. The register is used by the BCM5725/BCM5762/BCM57767 devices that require a second BAR.

Name	*Bits*	*Access*	*Default Value*	*Description*
Base Address (BAR_3)	31:8	RW	0x0	These bits set the address within a 32-bit address space that will be card will respond in. These bits may be combined with the bits in XBAR_4 to create a full 64 bit address decode. Only the bits that address blocks bigger than the setting in the BAR2_SIZE value are RW. All lower bits are RO with a value of zero. This value is sticky and only reset by HARD Reset.
Size indication	7:4	RO	0x0	RO bits indicate size of memory space. 7:4 RO=0 (256B BAR
Prefetchable	3	RO	0x0 Strap input to pcie block	This bit indicates that the area mapped by BAR_2 may be pre-fetched or cached by the system without side effects.
Type	2:1	Host RO	0x0 Strap input to pcie block	These bits indicate that BAR_2 may be programmed to map this adapter to anywhere in the 64-bit address space. Encoded with the following values **00**: located anywhere in 32-bit address space 01: reserved 10: located anywhere in 64-bit address space 11: reserved For function 1, this value is 0 (32-bit enabled)
Memory Space Indicator	0	RO	0x0	This bit indicates that BAR_2 maps a memory space and is always read as 0.

Base Address Register 4 (offset: 0x1c) – Function 1

The 32-bit Extended BAR_4 register programs the upper half of the 2nd base address for the memory space mapped by the card onto the PCI bus. This register as above is mainly to meet the BCM5725/BCM5762/BCM57767 device's need for a 2nd BAR.

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Base Address 4 (XBAR_4)	31:0	RO	0x00	These bits set the address upper 32-bit address space. These bits may be combined with the bits in BAR_3 to create a full 64 bit address decode. These bits must be set to zero for the card to respond to single address cycle requests. This value is sticky and only reset by HARD Reset

Base Address Register 5 (offset: 0x20) – Function 1

The 32-bit BAR_5 register programs the 2nd base address for the memory space mapped by the card onto the PCI bus. This register can be combined with Extended BAR_6 to make a 64-bit address for supporting Dual Address cycles systems.

Name	***Bits***	***Access***	***Default Value***	***Description***
Base Address (BAR_5)	31:8	RW	0x0	These bits set the address within a 32-bit address space that will be card will respond in. These bits may be combined with the bits in XBAR_6 to create a full 64 bit address decode. Only the bits that address blocks bigger than the setting in the BAR3_SIZE value are RW. All lower bits are RO with a value of zero. This value is sticky and only reset by HARD Reset.
Size indication	7:4	RO	0x0	RO bits indicate size of memory space. 7:4 RO=0 (256B BAR
Prefetchable	3	RO	0x0 Strap input to pcie block	This bit indicates that the area mapped by BAR_3 may be pre-fetched or cached by the system without side effects.
Type	2:1	Host RO	0x0 Strap input to pcie block	These bits indicate that BAR_3 may be programmed to map this adapter to anywhere in the 64-bit address space. Encoded with the following values **00**: located anywhere in 32-bit address space 01: reserved 10: located anywhere in 64-bit address space 11: reserved For function 1, this value is 0 (32-bit enabled)
Memory Space Indicator	0	RO	0x0	This bit indicates that BAR_3 maps a memory space and is always read as 0.

Base Address Register 6 (offset: 0x24) – Function 1

The 32-bit BAR_6 register programs the upper half of the 3rd base address for the memory space mapped by the card onto the PCI bus.

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Base Address 6 (XBAR_6)	31:0	RW	0x00	These bits set the address upper 32-bit address space. These bits may be combined with the bits in BAR_5 to create a full 64 bit address decode. These bits must be set to zero for the card to respond to single address cycle requests. This value is sticky and only reset by HARD Reset.

Cardbus CIS Pointer Register (offset: 0x28) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Cardbus CIS Pointer	31:0	FW-RW Host - RO	0x0	N/A for PCIe Device

Subsystem ID/Vendor ID Register (offset: 0x2C) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Subsystem Device ID	31:16	FW-RW Host - RO	0x96BC	Default for BCM5762/BCM57767 (SD/MMC Function 1): 0x96BC
Subsystem Vendor ID	15:0	FW-RW Host - RO	0x14E4	Identifies board manufacturer

Expansion ROM Base Address Register (offset: 0x30) – Function 1

This register is reset by PCIe Reset. It becomes a RW register if bit 5 of PCI State Register is set.

Name	*Bits*	*Access*	*Default Value*	*Description*
ROM Base Address	31:24	RW/RO	0xXXXX	These bits indicate the address of the Expansion ROM area.

Name	*Bits*	*Access*	*Default Value*	*Description*
ROM Size indication	23:11	RW/RO	0x00	These bits indicate the size of the Expansion ROM area or the address of it. The boundary form RO bits to RW bits is controlled by the EXP_ROM_SIZE bits.
Reserved	10:1	RO	0x000	These bits indicate that the Expansion ROM area is at least 2k bytes. They always read as zero. P
Expansion ROM Enable	0	RW/RO	0x0	This bit indicates that the Expansion ROM BAR is valid when set to one. If it is zero, the expansion BAR should not be programmed or used. This bit will only be RW if it is enabled by the EXP_ROM_ENA bit which defaults to 0.

Capabilities Pointer Register (offset: 0x34) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31:8	RO	0x0	Unused
Capabilities pointer	7:0	RO	0x48 (PM Cap)	The 8-bit Capabilities Pointer register specifies an offset in the PCI address space of a linked list of new capabilities. The capabilities are PCI-X, PCI Power Management, Vital Product Data (VPD), and Message Signaled Interrupts (MSI) is supported. The read-only value of this register is controlled by the CAP_ENA register in the PCI register space. **For function 1, this should point to offset 0x80 which is the PM Cap. Structure.**

Interrupt Register (offset: 0x3C) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
MAXIMUM_LATENCY	31:24	RO	0x00	Hardwired to zero
MIN_GRANT	23:16	RO	0x00	Hardwired to zero
Interrupt Pin	15:8	RO	0x02	Indicates which interrupt pin this device uses 0: no Interrupt 1: Use Interrupt A 2: Use Interrupt B 3: Use Interrupt C 4: Use Interrupt D
Interrupt Line	7:0	RW	0x00	Identifies interrupt routing information

Slot Information Register (offset: 0x40) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31:7	RO	0x00	Unused
Number of Slots	6:4	RO	0x01	These statuses indicate the number of slots the Host Controller supports. In the case of single function, maximum 6 slots can be assigned. 000b: 1 slot 001b: 2 slot 010b: 3 slot 011b: 4 slot 100b: 5 slot 101b: 6 slot Function 1 supports 2 slots: SD and MMC
RESERVED	3	RO	0x00	Unused
First Base Address Register Number	2:0	RO	0x00	Up to 6 Base Address can be specified in single configuration. These bits indicate first Base Address register number assigned for SD Host Controller register set. In the case of single function and multiple register sets, contiguous base addresses are used. **Number Of Slot** specifies number of base address. 000b: Base Address 10h (BAR0) 001b: Base Address 14h (BAR1) 010b: Base Address 18h (BAR2) 011b: Base Address 1Ch (BAR3) 100b: Base Address 20h (BAR4) 10 1b: Base Address 24h (BAR5)

Power Management Capability Register (offset: 0x48) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PME Support	31:27	RO	0x08 if no aux 0x18 if aux	Indicates the power states in which the device may assert PME. A 0 for any bit indicates that the device is not capable of asserting the PME pin signal while in that power state Bit 27: PME can be asserted from D0 Bit 28: PME can be asserted from D1 Bit 29: PME can be asserted from D2 Bit 30: PME can be asserted from D3H Bit 31: PME can be asserted from D3C (default depends on the presence of Aux power)
D2 Support	26	RO	0x0	Indicates whether the device supports the D2 PM state. This device does not support D2; hardwired to 0
D1 Support	25	RO FW-RW	0x0	Indicates whether the device supports the D1 PM state. This device does not support D1
Aux Current	24:22	RO FW-RW	0x0	This device supports the data register for reporting Aux Current requirements so this field is N/A
DSI	21	RO FW-RW	0x0	Indicates that the device requires device specific initialization (beyond PCI configuration header) before the generic class device driver is able to use it. This device hardwires this bit to 0 indicating that DSI is not necessary
Reserved	20	RO	0x0	–
PME Clock	19	RO	0x0	Indicates that the device relies on the presence of the PCI clock for PME operation. This device does not require the PCI clock to generate PME. Therefore, the bit is hardwired to 0
Version	18:16	RO	0x3	A value of 011b indicates that this function complies with revision 1.2 of the PCI PM specification.
PM Next Capabilities	15:8	RO	0xAC	Points to the next capabilities block which is Broadcom Vendor Specific Capability Header
PM Capability ID	7:0	RO	0x01	Identifies this item as Power management capabilities

Power Management Control/Status Register (offset: 0x4C) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
PM Data	31:24	RO FW-RW	0x00	Contains the power management data indicated by the Data Select field in PMCSR
Reserved	23:16	RO	0x00	–

Name	*Bits*	*Access*	*Default Value*	*Description*
PME Status	15	RW2C	0x0	This bit is set when the device asserts the WAKE signal independent of the PME enable bit. Writing 1 this bit will clear it and cause the deice to stop asserting WAKE
Data Scale	14:13	RO	0x1	Indicates the scaling factor that is used when interpreting the value of the data register (offset 7 in PM capability space). The device hardwires this value to 1 to indicate a scale of 1x
Data Select	12:9	RW	0x0	Indicates which data is to be reported via the Data register (offset 7 in PM capability space)
PME Enable	8	RW	0x1	Enables the device to generate PME when this bit is set to 1. When 0, PME generation is disabled
Reserved	7:4	RO	0x00	–
No Soft Reset	3	RO	0x1	No_Soft_Reset – When set (1), this bit indicates that devices transitioning from $D3_{hot}$ to D0 because of PowerState commands do not perform an internal reset. Configuration Context is preserved. Upon transition from the $D3_{hot}$ to the D0 Initialized state, no additional operating system intervention is required to preserve Configuration Context beyond writing the PowerState bits. When clear (0), devices do perform an internal reset upon transitioning from $D3_{hot}$ to D0 via software control of the PowerState bits. Configuration Context is lost when performing the soft reset. Upon transition from the $D3_{hot}$ to the D0 state, full reinitialization sequence is needed to return the device to D0 Initialized. Regardless of this bit, devices that transition from $D3_{hot}$ to D0 by a system or bus segment reset will return to the device state D0 Uninitialized with only PME context preserved if PME is supported and enabled
Reserved	2	RO	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Power State	1:0	RW	0x0	Indicates the current power state of the device when read When written, it sets the device into the specified power state. 00: D0 – Select D0 01: D1 – Select D1 10: D2 – Select D2 11: D3-Hot – Select D3 These bits may be used by the system to set the power state. The register is implemented as two banks of two bits each. Can be written from both configuration space and from the PCI register space as the PM_STATE bits. When written from the PCI bus, only values of 0 and 3 will be accepted. This is the register returned on reads of this register from configuration space. The second bank catches all writes values. The value of the second register is returned when the PM_STATE bits are read from register space. The idea of these registers is to a) Provide compatible operation to 5701 b) Allow implementation of other power states though firmware.

PCIe Capabilities Register (offset: 0xAC) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Unused
Message Number	29-25	FW-RW RO	0	Interrupt Message Number:indicate which MSI/MSI-X vector is used for the interrupt message generated in association with any of the status bits of this capability structure. For MSI, the value in this register indicates the offset between the base Message Data and the interrupt message that is generated. For MSI-X, the value in this register indicates which MSI-X Table entry is used to generate the interrupt message. The entry must be one of the first 32 entries even if the function implements more than 32 entries.
Slot implemented	24	RO	0	Hardwired to 0 because this is an endpoint device Slot Implemented. This register is not supported.
Device/Port Type	23-20	RO	0	Hardwired to 0 because this is an endpoint device Device/Port Type. Device is an End Point.
Capability Version	19-16	RO	2	Capability Version. PCI Express Capability structure version number. These bits are hardwired to 2h.
PCIE_NEXT_CAP_PTR	15-8	RO	0	This registers contains the pointer to the next PCI capability structure.

Name	Bits	Access	Default Value	Description
PCIE_CAP_ID	7-0	RO	0x10	This register contains the PCIExpress Capability ID.

Device Capabilities Register (offset: 0xB0) – Function 1

This register defines operational characteristics that are globally applicable to this device. This register is reset by Hard Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:29	RO	0	
FLR_CAP_SUPPORTED	28	RO	0	FLR capability is advertised when flr_supported bit in private device_capability register space is set.
Captured Slot Power Limit Scale	27:26	RO	0	This value specifies the scale used for the Power Limit 00 = 1.0x 01 = 0.1x 10 = 0.01x 11 = 0.001x Specifies the scale used for the Slot Power Limit Value. It is set by the Set_Slot_Power_Limit Message. This field is not set for Root ports.
Captured Slot Power Limit Value	25:18	RO	0	This value specifies the upper limit on the power supplied for this device Specifies the upper limit on power supplied by slot. It is set by the Set_Slot_Power_Limit Message. This field is not set for Root ports.
Reserved	17:16	RO	0	Unused
Role Based Error Support	15	RO	1	When set to 1, this value indicates that a role based error is supported. Indicate device is conforming to the ECN, PCI Express Base Specification, Revision 1.1., or subsequent PCI Express Base Specification revisions.
Reserved	14-12	RO	0	Unused

Name	***Bits***	***Access***	***Default Value***	***Description***
Endpoint L1 Acceptable Latency	11:9	FW-RW	6h	This value returns the latency that this device can accept when transitioning from the L1 to the L0 state 000 = less than 1us 001 = 1us to less than 2us 010 = 2us to less than 4us 011 = 4us to less than 8us 100 = 8us to less than 16us 101 = 16us to less than 32us 110 = 32us to 64us 111 = Greater than 64us Endpoint L1 Acceptable Latency. These bits are programmable through register space. The bits should be 0 for Root ports.
Endpoint L0s Acceptable Latency	8:6	FW-RW	6h	This value returns the total latency that this device can accept when transitioning from the L0s to L0 state 000 = less than 64ns 001 = 64ns to less than 128ns 010 = 128ns to less than 256ns 011 = 256ns to less than 512ns 100 = 512ns to less than 1us 101 = 1us to less than 2us 110 = 2us to 4us 111 = Greater than 4us Endpoint L0s Acceptable Latency. These bits are programmable through register space. The value should be 0 for root ports.
Extended Tag Field Supported	5	RO	0	This value returns the maximum supported tag field size when this function acts as a requester. 0 = 5bit tag field 1 = 8bit tag field We do support extended tag Extended Tag Field Support. This bit is programmable through register space. This capability is not currently supported.
Reserved	4:3	RO	0	Unused

Name	*Bits*	*Access*	*Default Value*	*Description*
Max Payload Size Supported	2:0	FW-RW	0	This value returns the maximum data payload size (in bytes) that this function supports for TLPs 0 = 128 1 = 256 2 = 512 3 = 1024 4 = 2048 5 = 4096 6, 7 = Reserved Max Payload Size Supported. These bits are programmable from the register space and default value is based on define in version.v file.

Device Status Control Register (offset: 0xB4) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-22	RO	0	–
Transaction Pending	21	RO	0	When this bit is set to 1, it indicates that this device has issued nonposted request packets which have not been completed This is bit is read back a 1, whenever a nonposted request initiated by PCIe core is pending to be completed.
Aux Power Detected	20	RO	0	When this bit is set to 1, it indicates that Aux power has been detected This bit is the current state of the VAUX_PRSNT pin of the device. When it is 1, it is indicating that part needs VAUX and detects the VAUX is present.
Unsupported Request Detected	19	W2C	0	When this bit is set to 1, it indicates that an Unsupported Request has been received Unsupported request detected.
Fatal Error Detected	18	W2C	0	When this bit is set to 1, it indicates that a Fatal Error has been detected. Fatal error detected.
Non-fatal Error Detected	17	W2C	0	When this bit is set to 1, it indicates that a nonfatal error has been detected. Nonfatal error detected.
Correctable Error Detected	16	W2C	0	When this bit is set to 1, it indicates that a correctable error has been detected. Correctable error detected.
FLR_INITIATED	15	RW	0	Initiate Function Level reset. This bit is writeable only if flr_supported bit in private device_capability register is set. A write of 1 to this bit initiates Function Level Reset. The value read by s/w from this bit is always 0.
Max Read Request Size	14:12	RW	2	This value controls the maximum read requests size for this device when acting as the requester 0 = 128 1 = 256 2 = 512 3 = 1024 4 = 2048 5 = 4096 6, 7, = Reserved Maximum Read Request Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions.
Enable No Snoop	11	RW	1	When this bit is set, the memory accessed by this device will not be cached by the processor. Enable No Snoop. When this bit is set to 1, PCIe initiates a read request with the No Snoop bit in the attribute field set for the transactions that request the No Snoop attribute.

Name	*Bits*	*Access*	*Default Value*	*Description*
Aux Power PM Enable	10	RO	1	When this bit is set, this device is enabled to draw aux power independent of PME power. This bit when set enables device to draw aux power independent of PME AUX power
RESERVED	9	RO	0	Unused
Extended Tag Field Enable	8	RW	0	When this bit is set, it enables this device to use an 8-bit Tag field as a requester. This capability is not supported.
Max Payload Size	7:5	RW	0	This value sets the maximum TLP data payload size (in bytes) for this device 0 = 128 1 = 256 2 = 512 3 = 1024 4 = 2048 5 = 4096 6, 7, = Reserved Max Payload Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions.
Enabled Relaxed Ordering	4	RW	1	When this bit is set, this device is permitted to set the Relaxed Ordering bit. Relax Ordering Enable.
Unsupported Request Reporting Enable	3	RW	0	When this bit is set, Unsupported Request reporting is enabled. Unsupported Request Reporting Enable.
Fatal Error Reporting Enabled	2	RW	0	When this bit is set, Fatal Error reporting is enabled. Fatal Error Reporting Enable.
Non-fatal Error Reporting Enabled	1	RW	0	When this bit is set, nonfatal error reporting is enabled. Nonfatal error reporting enable.
Correctable Error Reporting Enabled	0	RW	0	When this bit is set, correctable error reporting is enabled. Correctable error reporting enable.

Link Capability Register (offset: 0xB8) – Function 1

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Port Number	31:24	RO	0:HWinit	This value indicates the port number associated with this. PCIe Port Number. These bits are programmable through register.
Reserved	23:22	RO	0	

Name	*Bits*	*Access*	*Default Value*	*Description*
LINK_BW_NOTIFY	21	RO	0	Link Bandwidth Notification Capability: RC: A value of 1b indicates support for the Link Bandwidth Notification status and interrupt mechanisms. This capability is required for all Root Ports and Switch Downstream Ports supporting Links wider than x1 and/or multiple Link speeds. RC: Field is implemented. EP: Not supported and hardwired to 0.
DL_ACTIVE_REP	20	RO	0	Data Link Layer Link Active Reporting Capable: RC: this bit must be hardwired to 1b if the component supports the optional capability of reporting the DL_Active state of the Data Link Control and Management State Machine. RC: Implemented (RW) for RC. Default to 0. EP: Not supported and hardwired to 0.
SUR_DWN_ERR_REP	19	RO	0	Surprise Down Error Reporting Capable: RC: this bit must be set if the component supports the optional capability of detecting and reporting a Surprise Down error condition. RC: Not supported and hardwired to 0. EP: Not supported and hardwired to 0.
Clock Power Management	18	Host RO FW R/W	1	1: clkreq capable 0: clkreq not capable If it's mobile bonding, the default value will be 1, otherwise 0.
L1 Exit Latency	17:15	RO FW-RW	2	This value returns the L1 exit latency for this link 0 = Maximum of 1 us 1 = Maximum of 2 us 2 = Maximum of 4 us 3 = Maximum of 8 us 4 = Maximum of 16 us 5 = Maximum of 32 us 6 = Maximum of 64 us 7 = No limit L1 Exit Latency. These bits are programmable through register space. Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following.

Name	*Bits*	*Access*	*Default Value*	*Description*
L0s Exit Latency	14:12	RO FW-RW	4	This value returns the L0s exit latency for this link 0 = less than 64ns 1 = less than 128ns 2 = less than 256ns 3 = less than 512ns 4 = less than 1us 5 = less than 2us 6 = less than 4us 7 = greater than 4us L0s Exit Latency. These bits are programmable through register space. Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following.
Active State power management support	11:10	RO FW-RW	3	This value returns the supported ASPM states 0 = reserved 1 = L0s supported 2 = reserved 3 = L0s and L1 supported ASPM Support. These bits are programmable through reg space.
Maximum Link Width	9:4	RO FW-RW	1	This value returns the maximum link width. Allowable values are 1, 2, 4, 8, 12, 16, and 32 only. All other values are reserved. Value Name Description 1 1 One Lane Max 2 2 Two Lanes Max 4 4 Four Lanes Max 8 8 Eight Lanes Max Maximum Link Width. These are programmable through reg space.Bit 9 is always 0 and is not programmable. Default value is based on numLanes field in version.v.
Maximum Link Speed	3:0	RO FW-RW	1	This value returns the Maximum Link Speed. 1 = 2.5Gbps. All other values reserved. Value Name Description 1 2.5 2.5 Gbps Max 2 5 5 Gbps Max Value used by internal logic is the smaller of the value programmed for each function.

Link Status_Control Register (offset: 0xBC) – Function 1

This register is reset by Hard Reset or the rising edge of PERST_L.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-30	RO	0	Unused
DL_ACTIVE	29	RO	0	Data Link Layer Link Active: returns a 1b to indicate the DL_Active state, 0b otherwise. Not implemented and hardwire to 0.
Slot Clock Configuration	28	RO	1	This value indicates that this device uses the same physical reference clock that the platform provides on the connector. Slot Clock configuration. This bit is read-only by host, but read/write via backdoor CS bus.
LINK_TRAINING	27	RO	0	EP: This bit is N/A and is hardwired to 0.
Reserved	26	RO	0	Unused
Negotiated Link Width	25-20	RO	0	This value returns the negotiated link width. The only valid values are 1, 2, 4, 8, 12, 16, 32
Negotiated Link Speed	19-16	RO	0	This value returns the negotiated link speed. 1 = 2.5Gbps Link Speed. These bits indicate the negotiated link speed of the PCI Express link.
Reserved	15:12	RO	0	Unused
LINK_BW_INT_EN	11	RO	0	Link Autonomous Bandwidth Interrupt Enable: When Set, this bit enables the generation of an interrupt to indicate that the Link Autonomous Bandwidth Status bit has been Set. RC: Not implemented and hardwired to 0. EP: N/A and hardwired to 0.
LINK_BW_MGMT_INT_EN	10	RO	0	Link Bandwidth Management Interrupt Enable: when Set, this bit enables the generation of an interrupt to indicate that the Link Bandwidth Management Status bit has been Set. RC: N/A and hardwired to 0. EP: Not implemented and hardwired to 0.
HW_AUTO_WIDTH_DIS	9	RO	0	Hardware Autonomous Width Disable: When Set, this bit disables hardware from changing the Link width for reasons other than attempting to correct unreliable Link operation by reducing Link width. Other functions are reserved. RC: Not applicable and hardwire to 0 EP: If supported, only apply to function0. Not implemented and hardwire to 0.
Clock Request Enable	8	RW	0	1: clkreq is enabled 0: clkreq is disabled Enable Clock Power Management: RC: N/A and hardwired to 0. EP: When this bit is set, the device is permitted to use CLKREQ# signal to power management. Feature is enabled through version.v define

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended Synch	7	RW	0	When this bit is set, it forces extended sync which gives external devices (such as logic analyzers) additional time to achieve bit and symbol lock. Extended Synch. This bit when set forces the transmission of 4096 FTS ordered sets in the L0s state followed by a single SKP ordered set prior to entering the L0 state, and the transmission of 1024 TS1 ordered sets in the L1 state prior to entering the Recovery state. Value used by logic is resolved to 1 if either function has this bit set.
Common Clock Configuration	6	RW	0	When this bit is set, it indicates that the link partners are using a common reference clock. Common Clock Configuration. Value used by logic is resolved to 1 only if both functions (when enabled) have this bit set.
Reserved (Retrain Link)	5	RO	0	The device does not support this feature. Requesting PHY to retrain the link. This bit is only applicable to RC. So for EP it is read only bit.
Reserved (Link Disable)	4	RO	0	The device does not support this feature. Requesting PHY to disable the link. This bit is only applicable to RC. So for EP it is read only bit.
Read Completion Boundary	3	RW	0	This value indicates the Read Completion Boundary value (in bytes) of the upstream root port 0 = 64 1 = 128 Read Completion Boundary.
Reserved	2	RO	0	Unused
Active State Power Management Control	1:0	RW	0	This value control the Active State power management supported on this link 0 = Disabled 1 = L0s entry enabled 2 = L1 entry enabled 3 = L0s and L1 entry enabled ASPM Control. Value used by logic is dependent on the value of this bit for each enabled function and also on the programmed powerstate of each function.

Slot Capability Register (offset: 0xC0) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
PHYSICAL_SLOT_NUMBER	31-19	RO	0	Not Implemented
RESERVED	18-17	RO	0	Unused
SLOT_POWER_LIMIT_SCALE	16-15	RO	0	Not Implemented

Name	*Bits*	*Access*	*Default Value*	*Description*
SLOT_POWER_LIMIT_VALUE	14-7	RO	0	Not Implemented
RESERVED	6-0	RO	0	Not Implemented

Slot Control_Status Register (offset: 0xC4) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
SLOT_STATUS	31-23	RO	0	Not Implemented
PRESENCE_DETECT	22	RO	0	Not Implemented
RESERVED	21-16	RO	0	Not Implemented
SLOT_CONTROL	15-0	RO	0	Not Implemented

Root_Capability Control Register (offset: 0xC8) – Function 1

For EP this register is not applicable and hardwired to 0.

Root_Status Register (offset: 0xCC) – Function 1

For EP this register is not applicable and hardwired to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31-0	RO	0	Not Implemented

Device Capability 2 Register (offset: 0xD0) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
RESERVED	31-12	RO	0	Unused
LTR_MECHANISM_ SUPPORTED	11	RO	0	Latency Tolerance Reporting Mechanism Supported, Programmable through register space. This field will read 1, when bit 5 of ext_cap_ena field in private register space is set.
RESERVED	10-5	RO	0	Unused
CMPL_TIMEOUT_DISABL_ SUPPORTED	4	RO	1	Completion Timeout Disable Supported, Programmable through register space

Name	*Bits*	*Access*	*Default Value*	*Description*
CMPL_TIMEOUT_RANGES_ SUPPORTED	3-0	RO	F	Completion Timeout Ranges Supported. Programmable through register space. Value Name Description 15 ABCD 1 Ranges A,B,C and D

Device Status_Control 2 Register (offset: 0xD4) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
DEVICE_STATUS_2	31-16	RO	0	Placeholder for Gen2
RESERVED	15-11	RO	0	Unused
LTR_MECHANISM_ENABLE	10	RW	0	Latency Tolerance Reporting Mechanism Enable, This field is writeable, when bit 5 of ext_cap_ena field in private register space is set. This bit is RW only in Function 1 and is RsvdP for all other functions.
IDO_CPL_ENABLE	9	RW	0	IDO Completion Enable, This field is writeable, when bit ido_supported bit of private device_capability_2 register is set. When this bit is set, function is permitted to set ID based Ordering Attribute of Completions it returns.
IDO_REQ_ENABLE	8	RW	0	IDO Request Enable, This field is writeable, when bit ido_supported bit of private device_capability_2 register is set. When this bit is set, function is permitted to set ID based Ordering Attribute of Requests it initiates.
RESERVED	7-5	RO	0	Unused
CMPL_TIMEOUT_DISABLE	4	RW	0	Completion Timeout Disable
CMPL_TIMEOUT_VALUE	3-0	RW	0	Completion timeout value. The spec specifies a range, the device uses the max value in the range. Value Name Description 0 50MS 50 ms 1 100US 100 us 2 10MS 10 ms 3 55MS 55 ms 4 210MS 210 ms 5 900MS 900 ms 6 3_5S 3.5s 7 13S 13s 8 64S 64s

Link Capability 2 Register (offset: 0xD8) – Function 1

Name	*Bits*	*Access*	*Default Value*	*Description*
LINK_CAPABILITY_2	31-0	RO	0	Placeholder for Gen2

Link Status_Control 2 Register (offset: 0xDC) – Function 1

This register will be Read only by default, and will read all 0's to allow compliance with PCIe spec 1.1. To enable this register, reset comply_pcie_1_1 bit in the register space to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
LINK_STATUS_2	31-17	RO	0	Placeholder for Gen2
CURR_DEEMPH_LEVEL	16	RO	0	curr_deemph_level
RESERVED	15-13	RO	0	Unused
CFG_COMPLIANCE_DEEMPH	12	RW	0	Compliance deemphasis.
CFG_COMPLIANCE_SOS	11	RW	0	Compliance SOS.
CFG_ENTER_MOD_COMPLIANCE	10	RW	0	Enter Modified Compliance.
CFG_TX_MARGIN	9-7	RW	0	Controls the value of non deemphasized voltage level at the TX pins. Value used by logic is resolved to the smaller binary value, if two functions have different values. 0 000 800–1200 mV for full swing and 400–600 mV for half swing 1 001 Values are monotonic with non-zero slope 2 010 Values are monotonic with non-zero slope 3 011 200 –400 mV for full swing and 100–200 mV for half swing 4 100 Reserved 5 101 Reserved 6 110 Reserved 7 111 Reserved
SEL_DEEMPHASIS	6	RW	0	When link is operating at Gen2 rates, this bit selects the level of deemphasis. Value used by logic is resolved to 1 if either function has this bit set. 0 0 –6 dB 1 1 –3.5 dB

Name	Bits	Access	Default Value	Description
HW_AUTO_SPEED_DISABLE	5	RO	0	Not Supported and hardwired to 0.
ENTER_COMPLIANCE	4	RW	0	S/W instructs link to enter compliance mode. Value used by internal logic is set when either function has this bit enabled.
TARGET_LINK_SPEED	3-0	RW	0	Upper limit of link speed: 0 2_5 2.5 Gbps 1 5_0 5.0 Gbps

Slot Capability 2 Register (offset: 0xE0) – Function 1

Name	Bits	Access	Default Value	Description
SLOT_CAPABILITY_2	31-0	RO	0	Not Implemented

Slot Status_Control 2 Register (offset: 0xE4) – Function 1

Name	Bits	Access	Default Value	Description
SLOT_STATUS_2	31-16	RO	0	Not Implemented
SLOT_CONTROL_2	15-0	RO	0	Not Implemented

Extended PCIe Configuration Space – Function 1

Refer to the Extended PCIe Configuration Space in Section 4.1.55.

SD Host Standard Register

Summary of SD Register Set

See the following figures for an example of how this SD Register Set is mapped from the BAR_1 Register in the Function 1 of the PCIe Configuration Space. The SD Register Set is mapped starting at the PCI Configuration Register BAR_1 Register located at offset 0x10 in the PCIe Configuration Space of Function 1. Figure 62 shows a summary of the SD Register Set. Please refer to Section 2.2 of the document from the Link below for a detailed description of the SD Host Standard Register definition.

Figure 62: Mapping of SD Register Set for PCI Device

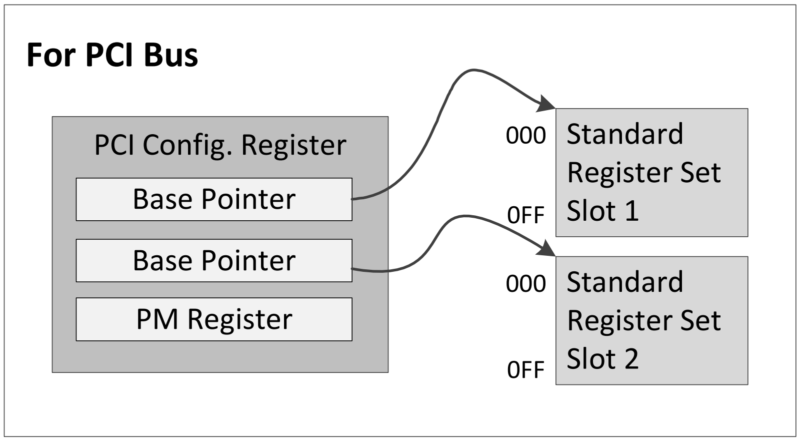
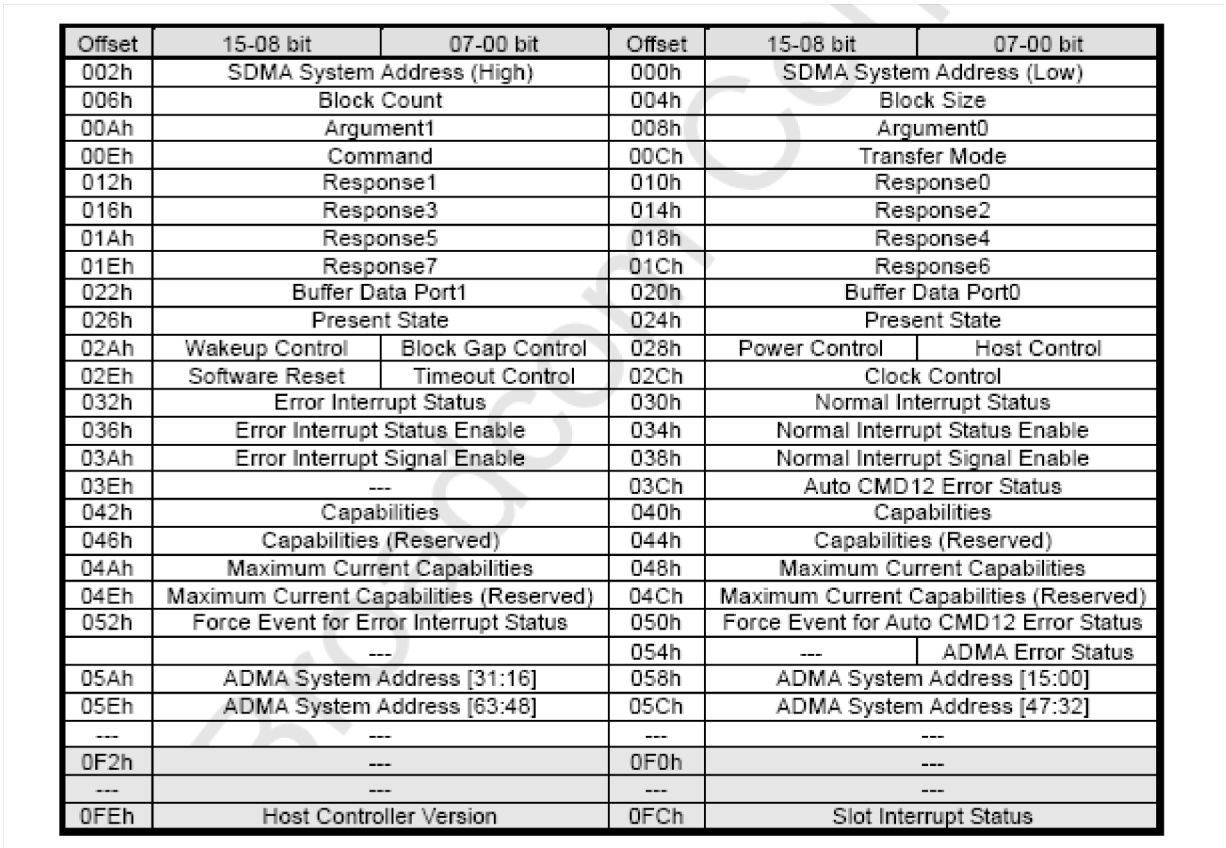


Figure 63: Summary of SD Register Set



Offset	15-08 bit	07-00 bit	Offset	15-08 bit	07-00 bit
002h	SDMA System Address (High)		000h	SDMA System Address (Low)	
006h	Block Count		004h	Block Size	
00Ah	Argument1		008h	Argument0	
00Eh	Command		00Ch	Transfer Mode	
012h	Response1		010h	Response0	
016h	Response3		014h	Response2	
01Ah	Response5		018h	Response4	
01Eh	Response7		01Ch	Response6	
022h	Buffer Data Port1		020h	Buffer Data Port0	
026h	Present State		024h	Present State	
02Ah	Wakeup Control	Block Gap Control	028h	Power Control	Host Control
02Eh	Software Reset	Timeout Control	02Ch	Clock Control	
032h	Error Interrupt Status		030h	Normal Interrupt Status	
036h	Error Interrupt Status Enable		034h	Normal Interrupt Status Enable	
03Ah	Error Interrupt Signal Enable		038h	Normal Interrupt Signal Enable	
03Eh	---		03Ch	Auto CMD12 Error Status	
042h	Capabilities		040h	Capabilities	
046h	Capabilities (Reserved)		044h	Capabilities (Reserved)	
04Ah	Maximum Current Capabilities		048h	Maximum Current Capabilities	
04Eh	Maximum Current Capabilities (Reserved)		04Ch	Maximum Current Capabilities (Reserved)	
052h	Force Event for Error Interrupt Status		050h	Force Event for Auto CMD12 Error Status	
	---		054h	---	ADMA Error Status
05Ah	ADMA System Address [31:16]		058h	ADMA System Address [15:00]	
05Eh	ADMA System Address [63:48]		05Ch	ADMA System Address [47:32]	
---	---		---	---	
0F2h	---		0F0h	---	
---	---		---	---	
0FEh	Host Controller Version		0FCh	Slot Interrupt Status	

High-Priority Mailbox Registers

All registers reset are core reset unless specified.

Interrupt Mailbox 0 (High Priority Mailbox) Register (offset: 0x200-207)

Offset	*Bits/Description*
0x00	Bits 31:24 – Status Tag Bits 7:0 – In_ISR
0x04	Note used

This mailbox register provides two functions:

- Whenever the host writes to this register, the Interrupt State is cleared, regardless of what value is written to this register. This applies to both the internal interrupt state, and the maskable external interrupt state (INTA). For instance, if an interrupt-causing event had previously occurred, but interrupts were masked (i.e., the Mask Interrupt bit in the Miscellaneous Host Control Register was set when the event occurred), an interrupt would be pending internally. However, writing any value to Interrupt Mailbox 0 would clear that internally pending interrupt. Thus, when interrupts are later unmasked, INTA would not be asserted due to that event because the event would have been cleared by the write to this register.
- Whenever In_ISR bits in this register contain a nonzero value (bits 31:0 non-zero when Tagged Status Mode is disabled, 0x68 bit-9 = 0), it indicates to the BCM57XX Ethernet controller that host software is in its interrupt processing routine (ISR). This causes the device to use the "during interrupt" coalescing registers as opposed to the "non-during interrupt" coalescing registers. In addition, since the device thinks the host is running its ISR, the device will not assert an interrupt if a status block is written back while this register contains a nonzero value. This provides host software with the flexibility of another mechanism to reduce interrupts (see Host Coalescing Control Registers).

Receive BD Standard Producer Ring Index (High Priority Mailbox) Register (offset: 0x268-0x26f)

The Receive BD Standard Producer Ring Index Register contains the index of the next buffer descriptor for the standard producer ring that will be produced in the host for the NIC to DMA into NIC memory. Host software writes this register whenever it updates the standard producer ring. This register must be initialized to 0.

Receive BD Return Ring 0 Consumer Index (High Priority Mailbox) Register (offset: 0x280-0x287)

The Receive BD Return Ring 0 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 0 that has been consumed. Host software writes this register whenever it updates the return ring 1. This register must be initialized to 0.

Receive BD Return Ring 1 Consumer Index (High Priority Mailbox) Register (offset: 0x288-0x28F)

The Receive BD Return Ring 1 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 1 that has been consumed. Host software writes this register whenever it updates the return ring 1. This register must be initialized to 0.

Receive BD Return Ring 2 Consumer Index (High Priority Mailbox) Register (offset: 0x290-0x297)

The Receive BD Return Ring 2 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 2 that has been consumed. Host software writes this register whenever it updates the return ring 2. This register must be initialized to 0.

Receive BD Return Ring 3 Consumer Index (High Priority Mailbox) Register (offset: 0x298-0x29F)

The Receive BD Return Ring 3 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 3 that has been consumed. Host software writes this register whenever it updates the return ring 3. This register must be initialized to 0.

Send BD Ring Host Producer Index (High Priority Mailbox) Register (offset: 0x300-0x307)

The Send BD Ring Host Producer Index Register contains the index of the next buffer descriptor for a given send ring that will be produced in the host for the NIC to DMA into NIC memory. Host software writes this register whenever it updates the given send ring. This register must be initialized to 0.

Send BD Ring NIC Producer Index (High Priority Mailbox) Register (offset: 0x380-0x387)

EthernetMAC Registers

All registers reset are core reset unless specified.

EMAC Mode Register (offset: 0x400)

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended magic packet enable	31	RW	0	1: Enable extended magic packet feature. 0: Disable extended magic packet feature.
Magic packet free-running mode enable	30	RW	0	1: Magic packet engine will continue comparing to the pattern of the next packet coming in after a match has been found in the previous packet. 0: Magic packet engine will stop comparing after a match has been found. It will stay in the match state until magic packet enable and extended packet enable bits are clear.
Mac loop back mode control	29	RW	1	1: gate off out-going TX data-path when emac loopback mode is enabled. 0: TX data will show up in normal functional path as well as MAC loopback path.
Enable APE TX path	28	RW	0	This bit must be written a 1 for the EMAC to transmit APE packets.
Enable APE RX path	27	RW	0	This bit must be written a 1 for APE subsystem to receive packets from the EMAC.
Free Running ACPI	26	RW	0	When this bit is set, the ACPI state machine will continue running when a match is found. When this bit is clear, the ACPI state machine will halt when a match is found
Halt Interesting packet PME	25	RW	0	When this bit is set, the WOL signal will not be asserted on an interesting packet match
Keep Frame in WOL	24	RW	0	–
Enable FHDE	23	RW	0	Enable receive Frame Header DMA engine. Must be set for normal operation
Enable RDE	22	RW	0	Enable RDMA engine. Must be set for normal operation
Enable TDE	21	RW	0	Enable Transmit DMA engine
Reserved	20	RO	0	–
ACPI Power-on Enable	19	RW	0	Enable Wake on LAN filters when in powerdown mode
Magic Packet Detection Enable	18	RW	0	Enable Magic Packet detection
Send Config Command	17	RW	0	Send config commands when in TBI mode
Flush TX statistics	16	RW	0	Write transmit statistics to external memory This bit is self-clearing

Name	*Bits*	*Access*	*Default Value*	*Description*
Clear TX statistics	15	RW	0	Clear transmit statistics internal RAM This bit is self-clearing
Enable TX Statistics	14	RW	0	Enable transmit statistics external updates
Flush RX Statistics	13	RW	0	Write receive statistics to external memory This bit is self-clearing
Clear RX Statistics	12	RW	0	Clear receive statistics internal RAM This bit is self-clearing
Enable RX Statistics	11	RW	0	Enable receive statistics external updates
Reserved	10	RO	0	–
Max Defer	9	RW	0	Enable Max Deferral checking statistic
Enable TX Bursting	8	RW	0	Enable transmit bursting in gigabit half-duplex mode
Tagged MAC Control	7	RW	0	Allow the MAC to receive tagged MAC control packets
Reserved	6:5	RO	0	–
Loopback mode	4	RW	0	When set, an internal loopback path is enabled from the transmit MAC to the receive MAC. This bit is provided for diagnostic purposes only
Port Mode	3:2	RW	10	These bits determine what interface the port is running 11: TBI 10: GMII 01: MII 00: None (default)
Half-duplex	1	RW	0	When set, the MII/GMII interface is configured to operate in half-duplex mode and the CSMA/CD state machines in the MAC are set to half-duplex mode
Global Reset	0	RW	0	When this bit is set to 1, the MAC state machine is reset. This is a self-clearing bit

EMAC Status Register (offset: 0x404)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
Interesting packet PME Attention	28	W2C	0	When this bit is set, the WOL signal is asserted when an interesting packet is detected
TX Statistic Overrun	27	W2C	0	Transmit Statistics block has overrun. Generates an attention when enabled
RX Statistic Overrun	26	W2C	0	Receive Statistics block has overrun. Generates an attention when enabled

Name	*Bits*	*Access*	*Default Value*	*Description*
ODI Error	25	RO	0	Output Data Interface block has an overrun or underrun. Will generate attention when enabled. Clear this attention using the Transmit Status register
AP Error	24	RO	0	Auto-polling interface needs service. Generates an attention when enabled. Clear this attention using the Auto-polling Status register
MII Interrupt	23	RO	0	Management interface is signaling an interrupt Generates an attention when enabled
MII Completion	22	W2C	0	Management interface transaction has completed Generates an attention when enabled
Reserved	21:13	RO	0	–
Link State Changed	12	W2C	0	Set when the link state has changed Generates an attention when enabled by bit 12 of the EMAC Event Enable register
Reserved	11:0	RO	0	–

EMAC Event Enable Register (offset: 0x408)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
Interesting packet PME Attention Enable	28	RW	0	When this bit is set, an attention will be asserted on an interesting packet match
TX Statistics Overrun	27	RW	0	Enable attention when transmit statistics block has overrun
RX Statistics Overrun	26	RW	0	Enable attention when receive statistics block has overrun
ODI Error	25	RW	0	Enable attention when an output data interface block has an overrun or underrun
AP Error	24	RW	0	Enable attention when the auto-polling interface has an error
MII Interrupt	23	RW	0	Enable attention when the Management Interface is signaling an interrupt
MII Completion	22	RW	0	Enable attention when the Management Interface transaction has completed
Reserved	21:13	RO	0	–
Link State Changed	12	RW	0	Enable attention when the link has changed state
Reserved	11:0	RO	0	–

LED Control Register (offset: 0x40C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Override Blink Rate	31	RW	1	If set, the blink rate for the Traffic LED is determined by the Blink Period field (bit 30 to bit 9). This bit is rest to 1. If not set, the blink rate assumes a Blink Period of 0x040, corresponding to approximately 15.9Hz
Blink Period	30:19	RW	000001 000000	Specifies the period of each blink cycle (on+off) for Traffic LED in milliseconds. Must be a nonzero value. This 12-bit field is reset to 0x040, giving a default blink period of approximately 15.9Hz
Reserved	18:16	RO	000	–
SPEED10/100 mode	15	RW	0	drives LINKLEDB and SPD100LEDB led pins, instead of SPD1000LEDB, to indicate 1000 link speed
Shared Traffic/Link LED mode	14	RW	1	When this bit is set, the Link LED is solid green when there is a link and blinks when there is traffic. (The LED_MODE field must be set to 00 before enabling this bit)
MAC Mode	13	RW	0	When this bit is set, the traffic LED blinks only when traffic is addressed for the device (The LED_MODE field must be set to 00 before enabling this bit)

LED Mod	12:11	RW	01	00: MAC Mode – LED signal is in active low (on) when link is established and is in tristate (off) when link is not established
				01: PHY Mode 1 – LED signal is in active low (on) when link is established and is in tristate (off) when link is not established
				LINKLEDB = Link 10 (open drain)
				SPD100LEDB = Link 100 (open drain)
				SPD1000LEDB = Link10000 (open drain)
				TRAFFICLEDB = PHY RCVLED or PHY XMTLED
				10: PHY Mode 2 – LED signal is in active low (on) when link is established and is in high (off) when link is not established
				LINKLEDB = Link 10
				SPD100LEDB = Link 100 and valid data or idle
				SPD1000LEDB = Link10000 and valid data or idle
				TRAFFICLEDB = PHY RCVLED or PHY XMTLED
				11: Same as PHY Mode 1

Traffic LED Status	10	RO	n/a	–
10Mbps LED Status	9	RO	n/a	–
100Mbps LED Status	8	RO	n/a	–
1000Mbps LED Status	7	RO	n/a	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Traffic LED	6	RW	0	If set along with the Override Traffic bit, the Traffic LED is turned on. If the Blink Traffic LED bit is also set, the LED will blink with blink rate specified in Override Blink Rate (bit 31) and Blink Period (bits 30-19) fields
Blink Traffic LED	5	RW	0	If set along with the Override Traffic bit and Traffic LED bit, the Traffic LED will blink with the blink rate specified in Override Blink Rate (bit 31) and Blink Period (bits 30-19) fields
Override Traffic LED	4	RW	0	If set, overrides hardware control of the Traffic LED. The Traffic LED will then be controlled via bit 6 and bit 5
10 Mbps LED	3	RW	0	If set along with the LED Override bit, turns on the 10 Mbps LED
100 Mbps LED	2	RW	0	If set along with the LED Override bit, turns on the 100 Mbps LED
1000 Mbps LED	1	RW	0	If set along with the LED Override bit, turns on the 1000 Mbps LED
Override Link LEDs	0	RW	0	If set, overrides hardware control of the three link LEDs. The LEDs will be controlled via bits 3-1

EMAC MAC Addresses 0 High Register (offset: 0x410)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
MAC Address High	15:0	RW	0	Upper 2-bytes of this node's MAC address

EMAC MAC Addresses 0 Low Register (offset: 0x414)

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Address Low	31:0	RW	0	Lower 4-byte of this node's MAC address

EMAC MAC Addresses 1 High Register (offset: 0x418)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
MAC Address High	15:0	RW	0	Upper 2-bytes of this node's MAC address

EMAC MAC Addresses 1 Low Register (offset: 0x41C)

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Address Low	31:0	RW	0	Lower 4-byte of this node's MAC address

EMAC MAC Addresses 2 High Register (offset: 0x420)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
MAC Address High	15:0	RW	0	Upper 2-bytes of this node's MAC address

EMAC MAC Addresses 2 Low Register (offset: 0x424)

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Address Low	31:0	RW	0	Lower 4-byte of this node's MAC address

EMAC MAC Addresses 3 High Register (offset: 0x428)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
MAC Address High	15:0	RW	0	Upper 2-bytes of this node's MAC address

EMAC MAC Addresses 3 Low Register (offset: 0x42C)

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Address Low	31:0	RW	0	Lower 4-byte of this node's MAC address

WOL Pattern Pointer Register (offset: 0x430)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
ACPI Pointer	8:0	RW	0	Specifies the offset into the 6KB BD memory for frame comparison. (Bits 3:0 are ignored to align the memory address to a natural 128-bit boundary)

WOL Pattern Configuration Register (offset: 0x434)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:28	RO	0	–
ACPI Offset	27:16	RW	0	Offset of a frame where the pattern comparison starts
Reserved	15:10	RO	0	–
ACPI Length	9:0	RW	0	Specifies the total number of 64-bit double words inside the MISC_BD memory that are valid for ACPI. For GMII, it should have a value of 2, 4, 6,… For MII, it should have a value of 3, 6, 9, …

Ethernet Transmit Random Backoff Register (offset: 0x438)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
Random Backoff Seed	9:0	RW	0	For half-duplex, initialize with any nonzero seed

Receive MTU Size Register (offset: 0x43C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
MTU	15:0	RW	05F2h	2-byte field which is the largest size frame that will be accepted without being marked as oversize

MII Communication Register (offset: 0x44C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	–

Name	***Bits***	***Access***	***Default Value***	***Description***
Start/Busy	29	RW	0	Set this bit to start a transaction While it is high, it indicates that the current transaction is still ongoing. If enabled, generates an attention via EMAC Status Register MII Completion bit (bit 22)
Read Failed	28	RO	0	When set, the transceiver device did not driver the bus during the attempted read transaction. Valid after the Start/Busy bit is cleared
Command	27:26	RW	0	These bits specify the transaction type: 11: Undefined 10: Read command 01: Write command 00: Undefined
PHY Addr	25:21	RW	0	PHY Address 0x0: PCIe SerDes 0x1: PHY Address
Register Address	20:16	RW	0	Address of the register to be read or written
Transaction Data	15:0	RW	0	When configured for a write command, the data stored at this location is written to the PHY at the specified PHY and register address During a read command, the data returned by the PHY is stored at this location

MII Status Register (offset: 0x450)

Name	***Bits***	***Access***	***Default Value***	***Description***
MII communication register overlap error	31	RO	0	This bit indicates MII management interface has an overlapped transaction (i.e., a new MII register transaction has been issue before current one is finished; 0x44c.29 is set again while 0x44c is already set) This bit is reset by POR only
Reserved	30:2	RO	0	–
Mode 10Mbps	1	RW	0	When read, a value of 1 indicates the transceiver device is operating in 10Mbps mode
Link Status	0	RW	0	The bit will generate an attention if enabled. Indicates status of the link on the transceiver device. When read, a value of 1 indicates the transceiver is linked

MII Mode Register (offset: 0x454)

Name	*Bits*	*Access*	*Default Value*	*Description*
MII communication delay fix disable	31	RW	0	To disable the fix for the extra cycle delay on bit 31:24 when reading MII communication register 0: all 32 bits of MII communication registers are available at the same read 1: bit 31:24 of MII communication registers are available 1 cycle earlier than 23:0 (e.g., when polling for bit 29 (start/busy) 1 extra read is required to obtain the data from 15:0
Reserved	30:21	RO	0	–
MII Clock Count	20:16	RW	0xB	Counter to divide CORE_CLK (62.5 MHz) to generate the MII clock The formula is MII Clock = CORE_CLK/2/(MII Clock Count + 1) The value of this register is not used if bit 15 is enabled
Enable constant MDC clock speed	15	RW	1	Enable constant 500 kHz MDIO interface speed regardless core clock frequency 0: disable – MDIO interface speed scales with core clock frequency. 1: enabled
Reserved	14:10	RO	0	–
PHY Address	9:5	RW	1	This field specifies the PHY Address
Port polling	4	RW	0	Set to enable autopolling of the transceiver link information from the MII management interface If cleared, the device will obtain the link status information from the state of the LINKRDY input signal
Reserved	3	RO	0	–
Auto_control	2	RW	0	–
Use Short Preamble	1	RW	0	Use short preamble while polling, if set
Fast_Clock	0	RW	0	–

Autopolling Status Register (offset: 0x458)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:4	RO	0	–
Error Channel	3:1	W2C	0	–
Auto-polling Error	0	W2C	0	Indicates an autopolling error occurred if set

Transmit MAC Mode Register (offset: 0x45C)

Name	*Bits*	*Access*	*Default Value*	*Description*
RR Weight	31:27	R/W	00000	This field may be programmed to assign a weight to the Weighted Round Robin arbitration mode. This field is applicable only when the appropriate arbitration mode is chosen, for example [19:17] of this register is equal to 001.
Transmit FTQ Arbitration Mode	26:24	R/W	000	This field determines the arbitration mode of the TCE block among LAN traffic and APE traffic as below: 000 – Simple Round Robin 001 – Weighted Round Robin 010 – Shut-off APE transmit stream 011 – Shut-off LAN transmit stream 1xx – Reserved for future use ***Caution:*** This field must remain static following boot.
Reserved	23:21	RO	0	–
TX-MBUF Burst Size	20:17	R/W	0000	This filed determines the size of the MA read performed by TCE. 0000 => burst-size 16 0001 – 01111 => reserved 1000 => burst-size 8 1001 => burst-size 9 ……… 1111 => burst-size 15
Do not insert GCM/ GMAC IV	16	R/W	0	If this bit is 0, an IV is generated by the chip and inserted in an offloaded TX ESP or TX AH packet in GCM or GMAC Cipher mode. If this bit is 1, an IV is not inserted. Instead, the IV is extracted from the TX frame.
Do not drop if packet found malformed	15	R/W	0	These bits are for debug purposes. Normally an offloaded TX which packet does not comply to IPSEC protocol restrictions or does not adhere to Snaggletooth limitations or does not associate with a valid SA, unless the bit corresponding to the error symptom is set to 1 here – In that case the particular error is overlooked and the packet is transmitted in clear text. When such a packet is dropped in the chip, an interrupt is generated
Do not drop if SA found in RX direction	14	R/W	0	–
Do not drop if unsupported IPV6 extension found or Ipv4 option found	13	R/W	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Do not drop if SA invalid	12	R/W	0	–
Do not drop if AH/ESP Header not found	11	R/W	0	–
Enable TX AH Offload	10	R/W	0	A value 1 enables the TX AH offload feature – when 0, offloaded AH packet gets dropped. This value must be static.
Enable TX ESP Offload	9	R/W	0	A value 1 enables the TX ESP offload feature – when 0, offloaded ESP packet gets dropped. This value must be static.
Enable Bad TxMBUF Lockup fix	8	RW	1	When set, enables fix for CQ12429, Bad TxMBUF Lockup fix.
Link Aware Enable	7	RW	0	When set, transmission of packets by the MAC is enabled only when link is up
Enable Long Pause	6	RW	0	When set, the Pause time value set in the transmitted PAUSE frames is 0xFFFF. The default value for PAUSE time is 0x1FFF
Enable Big Backoff	5	RW	0	MAC will use larger than normal back-off algorithm
Enable Flow Control	4	RW	0	MAC will send 802.3x flow control frames
Reserved	3:2	RO	0	–
Enable	1	RW	0	Enable Transmit MAC state machines
Reset	0	RW	0	When this bit is set to 1, the Transmit MAC state machine will be reset. This is a self-clearing bit

Transmit MAC Status Register (offset: 0x460)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
ODI Overrun	5	W2C	0	Output data interface has overrun
ODI Underrun	4	W2C	0	Output data interface has underrun
Link Up	3	RO	0	Link is up, if set
Sent XON	2	W2C	0	An XON flow control frame was sent
Sent XOFF	1	W2C	0	An XOFF flow control frame was sent
RX Currently XOFFed	0	RO	0	Received stopped due to flow control

Transmit MAC Lengths Register (offset: 0x464)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RO	0	–
IPG CRS Length	13:12	RW	0	When multiplied by 2, this field indicates the number of bytes from the end of the interpacket gap (IPG) during which incoming carrier is ignored.
IPG Length	11:8	RW	0	When multiplied by 2, this field indicates the number of bytes in the entire IPG
Slot Time Length	7:0	RW	0	When multiplied by 2, this field indicates the number of bytes in the slot time

Receive MAC Mode Register (offset: 0x468)

Name	*Bits*	*Access*	*Default Value*	*Description*
Disable hardware fix 24175 CQ24175 Fix Disable	31	RW	0	When set, disables hardware fix 24175 This bit when set disables CQ24175 fix where EMAC drops first packet on False Carrier Event. 1: Disable Fix 0: Enable Fix
Disable hardware fix 29914 CQ29914 Fix Disable	30	RW	0	When set, disables hardware fix 29914 This bit when set disables CQ29914 fix where a control frame of odd length may be dropped due to an incorrectly flagged CRC error 1: Disable Fix 0: Enable Fix
Disable 802.3 length check fix for VLAN Tag frames	29	R/W	0	If clear, 802.3 length check takes VLAN length into account properly.
Reserved	28	RO	0	–
Reserved	27	RO	0	–
Status Ready New Disable	26	RO	0	–
IPv4 Fragment Fix	25	RW	1	–
IPV6 Enable	24	RW	0	1: Enable IPv6 RX 0: Disable IPv6 RX which includes IPv6 packet parsing, checksum offload and IPv6 RSS
RSS_enable	23	RW	0	1: Enable RSS function. 0: Disable RSS function. FHDE will ignore the RSS_valid from Frame Cracker and set RSS_valid to be 0 in frame descriptor of each packet.

Name	*Bits*	*Access*	*Default Value*	*Description*
RSS Hash Mask Bits	22:20	RW	0x7	These bits specify the number of hash bits that are used to offset into the indirection table. A value of one specifies that only bit 0 of the hash is used to offset into the indirection table (so only the first two entries of the table are utilized.) A value of seven specifies that bits 6:0 of the hash are used to offset into the indirection table. A value of zero will result in undefined behavior and should not be programmed.
RSS TCP/IPV6 Hash Enable	19	RW	0	When this bit is set, 4-tuple hashes are enabled for TCP over IPV6 packets. This bit should be set to 0 if IPv6 RX is disabled.
RSS IPV6 Hash Enable	18	RW	0	When this bit is set, 2-tuple hashes are enabled for IPV6 packets. This bit should be set to 0 if IPv6 RX is disabled.
RSS TCP/IPV4 Hash Enable	17	RW	0	When this bit is set, 4-tuple hashes are enabled for TCP over IPV4 packets.
RSS IPV4 Hash Enable	16	RW	0	When this bit is set, 2-tuple hashes are enabled for IPV4 packets.
Reserved	15:14	RO	0	–
APE promiscuous mode enable	13	R/W	0	When set, no source address or MC hashing checking will be performed on incoming frames on APE filter path. All frames will be accepted and subject to Management filter actions.
CQ42199 fix disable	12	RW	0	This bit disables a fix for CQ42199. EMAC stage I fsm drops a packet if incoming packet's DA has a partial match in both perfect match address and Pause Multicast address.
Filter Broadcast	11	RW	0	When set, reception of broadcast frames is disabled
Keep VLAN Tag Diag Mode	10	RW	0	If set, forces Receive MAC to keep the VLAN tag in the frame. This is for debugging purpose only and should be reset during normal operation
No CRC Check	9	RW	0	When set, mo CRC check by receive MAC on incoming frames. Also, allows the reception of packets received with RXERR on MII/GMII
Promiscuous Mode	8	RW	0	When set, no source address or MC hashing checking will be performed on incoming frames All frames will be accepted
Length Check	7	RW	0	If set, 802.2 length checking is done on LLC frames
Accept Runts	6	RW	0	If set, MAC accepts packets less than 64 bytes
Keep Oversized	5	RO	0	–
Keep Pause	4	RW	0	If set, MAC forwards pause frame to host buffer
Keep MFC	3	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Flow Control	2	RW	0	Enable automatic processing of 802.3x flow control frames This bit is orthogonal to the Keep Pause bit
Enable	1	RW	0	This bit controls whether the Receive MAC state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read.
Reset	0	RW	0	When this bit is set to 1, the Receive MAC state machine will be reset. This is a self-clearing bit

Receive MAC Status Register (offset: 0x46C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
ACPI packet received	5	W2C	0	ACPI packet was detected by h/w block and caused PME to be asserted; reset by POR only
Magic packet received	4	W2C	0	Magic packet was detected by h/w block and caused PME to be asserted; reset by POR only
RX FIFO Overrun	3	W2C	0	RX FIFO has encountered an overrun condition
XON received	2	W2C	0	MAC control frame with the PAUSE opcode was received with PAUSE TIME field set to zero The bit is sticky and must be written to clear
XOFF received	1	W2C	0	MAC control frame with the PAUSE opcode was received with PAUSE TIME field set to nonzero The bit is sticky and must be written to clear
Remote Transmitter XOFFed	0	RO	0	A previously received XOFF timer has not expired yet

MAC Hash Register 0 (offset: 0x470)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash value	31:0	RW	0	Hash value for multicast destination address matching

MAC Hash Register 1 (offset: 0x474)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash value	31:0	RW	0	Hash value for multicast destination address matching

MAC Hash Register 2 (offset: 0x478)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash value	31:0	RW	0	Hash value for multicast destination address matching

MAC Hash Register 3 (offset: 0x47C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash value	31:0	RW	0	Hash value for multicast destination address matching

Receive Rules Control Registers (offset: 0x480 + 8*N)

The BCM5725/BCM5762/BCM57767 employs eight receive rules (N = 0 to 7).

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable	31	RW	0	Corresponding Rule is enabled when set
And With Next	30	RW	0	This rule and next must both be true to match. The class fields must be the same. A disabled next rule is considered true. Processor activation bits are specified in the first rule in series
Activate Processor 1	29	RW	0	If the rule matches, the processor is activated in the queue descriptor for the Receive Queue Placement state machine
Reserved	28	RO	0	Reserved
Reserved	27	RO	0	Reserved
Mask	26	RW	0	IF set, specifies that the value/mask field is split into a 16-bit mask instead of a 32bit value
Discard	25	RW	0	Discard frame if it matches the rule
Map	24	RW	0	Use the masked value and map it to the class
Reserved for future use	23:18	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Comparison Operator	17:16	RW	0	Specifies how to determine the match: 00: Equal 01: Not Equal 10: Greater Than 11: Less Than
Header Type	15:13	RW	0	Specifies which header the offset is for: 000: Start of Frame (always valid) 001: Start of IP Header (if present) 010: Start of TCP Header (if present) 011: Start of UDP Header (if present) 100: Start of Data (always valid, context sensitive)
Class	12:8	RW	0	The class this frame is place into if the rule matches. 0-16 where 0 means discard. The number of valid classes is the number of active queues divided by the Number of Interrupt Distribution Groups. Ring 1 has the highest priority
Offset	7:0	RW	0	Number of bytes offset specified by the header type

Receive Rules Value/Mask Registers (offset: 0x484 + 8*N)

The BCM5725/BCM5762/BCM57767 employs eight receive rules (N = 0 to 7).

Name	*Bits*	*Access*	*Default Value*	*Description*
Mask/Value	31:16	RW	0	For each bit cleared, the corresponding bit in the value field is ignored during the rule match process If bit 26 of the corresponding rule control register is 0, the field is used as the upper 16-bit value for rule comparison
Value	15:0	RW	0	This field specifies a 16-bit value for rule comparison

Receive Rules Configuration Register (offset: 0x500)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
No Rules Matches Default Class	7:3	RW	0	Specifies the default class of service for the frame if no rules are matched A value of 1 is the highest priority A value of zero will cause the frame to be discarded

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	2:0	RO	0	–

Low Watermark Maximum Receive Frame Register (offset: 0x504)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
TXFIFO Almost Empty Threshold	20:16	RW	0xC	When the remaining entries of TXFIFO are less than this threshold, TXFIFO_almost_empty will be asserted. This value is used in conjunction with Buffer Manager Mode register bit31 to prevent EMAC TXFIFO underrun.
Low Watermark Max Receive Frames	15:0	RW	0	Specifies the number of good frames to receive after RX MBUF Low Watermark has been reached. After the RX MAC receives this number of frames, it will drop subsequent incoming frames until the MBUF High Watermark is reached Default to zero (i.e., drop frames ones RX MBUF Low Watermark is reached)

Regulator Voltage Control Register (offset: 0x590)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0x0	Reserved
Regclt_1_2V_Core	23:20	RW	0x9	1.2V core regulator output voltage trip control. Power-up default is hard coded (0x9) Default can be over-written from OTP using bit 142:140/12:10. Setting is programmable. OTP bit 139/9 can be used to power down this regulator. 0x0000 +18% 0x0001 +16% 0x0010 +14% 0x0011 12% 0x0100 +10% 0x0101 +8% 0x0110 +6% 0x0111 +4% 0x1000 +2% 0x1001 +0% (1.20v) 0x1010 -2% 0x1011 -4% 0x1100 -6% 0x1101 -8% 0x1110 -10% 0x1111 -12%
Reserved	19:16	RO	0x0	–
Spd1000 LED pin output override	15	RW	0x0	Override value of Spd1000 LED pin output
Spd1000 LED pin oe override	14	RW	0x0	Override value of Spd1000 LED pin output enable
Enable Spd1000 LED pin override	13	RW	0x0	Set to 1 to override Spd1000 LED pin
Spd1000 LED pin input	12	RO	0x0	Spd1000 LED pin input
Spd100 LED pin output override	11	RW	0x0	Override value of Spd100 LED pin output
Spd100 LED pin oe override	10	RW	0x0	Override value of Spd100 LED pin output enable
Enable Spd100 LED pin override	9	RW	0x0	Set to 1 to override Spd100 LED pin
Spd100 LED pin input	8	RO	0x0	Spd100 LED pin input
Link LED pin output override	7	RW	0x0	Override value of Link LED pin output
Link LED pin oe override	6	RW	0x0	Override value of Link LED pin output enable

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Link LED pin override	5	RW	0x0	Set to 1 to override Link LED pin
Link LED pin input	4	RO	0x0	Link LED pin input
Traffic LED pin output override	3	RW	0x0	Override value of Traffic LED pin output
Traffic LED pin oe override	2	RW	0x0	Override value of Traffic LED pin output enable
Enable Traffic LED pin override	1	RW	0x0	Set to 1 to override Traffic LED pin
Traffic LED pin input	0	RO	0x0	Traffic LED pin input

EAV: TX Time Stamp LSB Register (offset: 0x5c0)

Name	*Bits*	*Access*	*Default Value*	*Description*
TX Time Stamp [lower half]	31:0	RO	U	LSB of the TX Time Stamp – Reading this LSB freezes the time stamp and is only unfrozen when the corresponding MSB is read

EAV: TX Time Stamp MSB Register (offset: 0x5c4)

Name	*Bits*	*Access*	*Default Value*	*Description*
TX Time Stamp [Upper half]	31:0	RO	U	MSB of the TX Time Stamp – Reading this MSB unfreezes the time stamp which was earlier frozen by the corresponding LSB read.

EAV: AV transmit Tolerance Window Register (offset: 0x5c8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:20	RO	0x0	–
AV Transmit Tolerance	19:15	RW	0x4	This value is multiplied by **32.768µs** to determine the AV Transmit Tolerance Window value.
Reserved	14:1	RO	0x0	–
Disable Real-Time Transmit Algorithm	0	RW	0	When this bit is 1, the TX-EMAC treats the Real-Time FTQ or FTQ2 just as another Best-effort queue. The Real-Time scheduling algorithm is forgone and FTQ1 and FTQ2 are arbitrated on a fair Round-Robin basis.

EAV: REAL-TIME TRANSMIT QUALITY1 REG [Offset 0X05CC]

Name	*Bits*	*Access*	*Default Value*	*Description*
Quality Metric 1	31:0	RO/ ClearsOnRead	0x0	This field counts the number of frames on Real-Time Ring that violated the AV Transmit Tolerance window by no more than 2x of the programmed value in reg 0x5C8. These frames were actually scheduled at a time T, where: • T > Send Launch Time Marker + 1 x AV Transmit Tolerance Window And • T <= Send Launch Time Marker + 2 x AV Transmit Tolerance Window

EAV: REAL-TIME TRANSMIT QUALITY2 REG [Offset 0X05D0]

Name	*Bits*	*Access*	*Default Value*	*Description*
Quality Metric 2	31:0	RO/ ClearsOnRead	0x0	This field counts the number of frames on Real-Time Ring that violated the AV Transmit Tolerance window by no more than 4x of the programmed value in reg 0x5C8. These frames were actually scheduled at a time T, where: • T > Send Launch Time Marker + 2 x AV Transmit Tolerance Window And • T <= Send Launch Time Marker + 4 x AV Transmit Tolerance Window

EAV: REAL-TIME TRANSMIT QUALITY3 REG [Offset 0X05D4]

Name	Bits	Access	Default Value	Description
Quality Metric 3	31:0	RO/ ClearsOnRead	0x0	This field counts the number of frames on Real-Time Ring that violated the AV Transmit Tolerance window by no more than 8x of the programmed value in reg 0x5C8. These frames were actually scheduled at a time T, where: • T > Send Launch Time Marker + 4 x AV Transmit Tolerance Window And • T <= Send Launch Time Marker + 8 x AV Transmit Tolerance Window

EAV: REAL-TIME TRANSMIT QUALITY4 REG [Offset 0X05D8]

Name	Bits	Access	Default Value	Description
Quality Metric 4	31:0	RO/ ClearsOnRead	0x0	This field counts the number of frames on Real-Time Ring that violated the AV Transmit Tolerance window more than 8x of the programmed value in reg 0x5C8. These frames were actually scheduled at a time T, where: T > Send Launch Time Marker + 8 x AV Transmit Tolerance Window

RSS Registers

All registers reset are core reset unless specified.

Indirection Table Register 1 (offset: 0x630)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry0	29:28	RW	0	The RSS_ring value for entry 0.
Reserved	27:26	RO	0	Not Used
table_entry1	25:24	RW	0	The RSS_ring value for entry 1.
Reserved	23:22	RO	0	Not Used
table_entry2	21:20	RW	0	The RSS_ring value for entry 2.
Reserved	19:18	RO	0	Not Used
table_entry3	17:16	RW	0	The RSS_ring value for entry 3.
Reserved	15:14	RO	0	Not Used
table_entry4	13:12	RW	0	The RSS_ring value for entry 4.
Reserved	11:10	RO	0	Not Used
table_entry5	9:8	RW	0	The RSS_ring value for entry 5.
Reserved	7:6	RO	0	Not Used
table_entry6	5:4	RW	0	The RSS_ring value for entry 6.
Reserved	3:2	RO	0	Not Used
table_entry7	1:0	RW	0	The RSS_ring value for entry 7.

Indirection Table Register 2 (offset: 0x634)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry8	29:28	RW	0	The RSS_ring value for entry 8.
Reserved	27:26	RO	0	Not Used
table_entry9	25:24	RW	0	The RSS_ring value for entry 9.
Reserved	23:22	RO	0	Not Used
table_entry10	21:20	RW	0	The RSS_ring value for entry 10.
Reserved	19:18	RO	0	Not Used
table_entry11	17:16	RW	0	The RSS_ring value for entry 11.
Reserved	15:14	RO	0	Not Used

Name	*Bits*	*Access*	*Default Value*	*Description*
table_entry12	13:12	RW	0	The RSS_ring value for entry 12.
Reserved	11:10	RO	0	Not Used
table_entry13	9:8	RW	0	The RSS_ring value for entry 13.
Reserved	7:6	RO	0	Not Used
table_entry14	5:4	RW	0	The RSS_ring value for entry 14.
Reserved	3:2	RO	0	Not Used
table_entry15	1:0	RW	0	The RSS_ring value for entry 15.

Indirection Table Register 3 (offset: 0x638)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry16	29:28	RW	0	The RSS_ring value for entry 16.
Reserved	27:26	RO	0	Not Used
table_entry17	25:24	RW	0	The RSS_ring value for entry 17.
Reserved	23:22	RO	0	Not Used
table_entry18	21:20	RW	0	The RSS_ring value for entry 18.
Reserved	19:18	RO	0	Not Used
table_entry19	17:16	RW	0	The RSS_ring value for entry 19.
Reserved	15:14	RO	0	Not Used
table_entry20	13:12	RW	0	The RSS_ring value for entry 20.
Reserved	11:10	RO	0	Not Used
table_entry21	9:8	RW	0	The RSS_ring value for entry 21.
Reserved	7:6	RO	0	Not Used
table_entry22	5:4	RW	0	The RSS_ring value for entry 22.
Reserved	3:2	RO	0	Not Used
table_entry23	1:0	RW	0	The RSS_ring value for entry 23.

Indirection Table Register 4 (offset: 0x63C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry24	29:28	RW	0	The RSS_ring value for entry 24.
Reserved	27:26	RO	0	Not Used
table_entry25	25:24	RW	0	The RSS_ring value for entry 25.
Reserved	23:22	RO	0	Not Used
table_entry26	21:20	RW	0	The RSS_ring value for entry 26.
Reserved	19:18	RO	0	Not Used
table_entry27	17:16	RW	0	The RSS_ring value for entry 27.
Reserved	15:14	RO	0	Not Used
table_entry28	13:12	RW	0	The RSS_ring value for entry 28.
Reserved	11:10	RO	0	Not Used
table_entry29	9:8	RW	0	The RSS_ring value for entry 29.
Reserved	7:6	RO	0	Not Used
table_entry30	5:4	RW	0	The RSS_ring value for entry 30.
Reserved	3:2	RO	0	Not Used
table_entry31	1:0	RW	0	The RSS_ring value for entry 31.

Indirection Table Register 5 (offset: 0x640)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry32	29:28	RW	0	The RSS_ring value for entry 32.
Reserved	27:26	RO	0	Not Used
table_entry33	25:24	RW	0	The RSS_ring value for entry 33.
Reserved	23:22	RO	0	Not Used
table_entry34	21:20	RW	0	The RSS_ring value for entry 34.
Reserved	19:18	RO	0	Not Used
table_entry35	17:16	RW	0	The RSS_ring value for entry 35.
Reserved	15:14	RO	0	Not Used
table_entry36	13:12	RW	0	The RSS_ring value for entry 36.
Reserved	11:10	RO	0	Not Used
table_entry37	9:8	RW	0	The RSS_ring value for entry 37.
Reserved	7:6	RO	0	Not Used
table_entry38	5:4	RW	0	The RSS_ring value for entry 38.

Name	Bits	Access	Default Value	Description
Reserved	3:2	RO	0	Not Used
table_entry39	1:0	RW	0	The RSS_ring value for entry 39.

Indirection Table Register 6 (offset: 0x644)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	Not Used
table_entry40	29:28	RW	0	The RSS_ring value for entry 40.
Reserved	27:26	RO	0	Not Used
table_entry41	25:24	RW	0	The RSS_ring value for entry 41.
Reserved	23:22	RO	0	Not Used
table_entry42	21:20	RW	0	The RSS_ring value for entry 42.
Reserved	19:18	RO	0	Not Used
table_entry43	17:16	RW	0	The RSS_ring value for entry 43.
Reserved	15:14	RO	0	Not Used
table_entry44	13:12	RW	0	The RSS_ring value for entry 44.
Reserved	11:10	RO	0	Not Used
table_entry45	9:8	RW	0	The RSS_ring value for entry 45.
Reserved	7:6	RO	0	Not Used
table_entry46	5:4	RW	0	The RSS_ring value for entry 46.
Reserved	3:2	RO	0	Not Used
table_entry47	1:0	RW	0	The RSS_ring value for entry 47.

Indirection Table Register 7 (offset: 0x648)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	Not Used
table_entry48	29:28	RW	0	The RSS_ring value for entry 48.
Reserved	27:26	RO	0	Not Used
table_entry49	25:24	RW	0	The RSS_ring value for entry 49.
Reserved	23:22	RO	0	Not Used
table_entry50	21:20	RW	0	The RSS_ring value for entry 50.
Reserved	19:18	RO	0	Not Used
table_entry51	17:16	RW	0	The RSS_ring value for entry 51.
Reserved	15:14	RO	0	Not Used

Name	*Bits*	*Access*	*Default Value*	*Description*
table_entry52	13:12	RW	0	The RSS_ring value for entry 52.
Reserved	11:10	RO	0	Not Used
table_entry53	9:8	RW	0	The RSS_ring value for entry 53.
Reserved	7:6	RO	0	Not Used
table_entry54	5:4	RW	0	The RSS_ring value for entry 54.
Reserved	3:2	RO	0	Not Used
table_entry55	1:0	RW	0	The RSS_ring value for entry 55.

Indirection Table Register 8 (offset: 0x64C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry56	29:28	RW	0	The RSS_ring value for entry 56.
Reserved	27:26	RO	0	Not Used
table_entry57	25:24	RW	0	The RSS_ring value for entry 57.
Reserved	23:22	RO	0	Not Used
table_entry58	21:20	RW	0	The RSS_ring value for entry 58.
Reserved	19:18	RO	0	Not Used
table_entry59	17:16	RW	0	The RSS_ring value for entry 59.
Reserved	15:14	RO	0	Not Used
table_entry60	13:12	RW	0	The RSS_ring value for entry 60.
Reserved	11:10	RO	0	Not Used
table_entry61	9:8	RW	0	The RSS_ring value for entry 61.
Reserved	7:6	RO	0	Not Used
table_entry62	5:4	RW	0	The RSS_ring value for entry 62.
Reserved	3:2	RO	0	Not Used
table_entry63	1:0	RW	0	The RSS_ring value for entry 63.

Indirection Table Register 9 (offset: 0x650)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry64	29:28	RW	0	The RSS_ring value for entry 64.
Reserved	27:26	RO	0	Not Used
table_entry65	25:24	RW	0	The RSS_ring value for entry 65.

Name	Bits	Access	Default Value	Description
Reserved	23:22	RO	0	Not Used
table_entry66	21:20	RW	0	The RSS_ring value for entry 66.
Reserved	19:18	RO	0	Not Used
table_entry67	17:16	RW	0	The RSS_ring value for entry 67.
Reserved	15:14	RO	0	Not Used
table_entry68	13:12	RW	0	The RSS_ring value for entry 68.
Reserved	11:10	RO	0	Not Used
table_entry69	9:8	RW	0	The RSS_ring value for entry 69.
Reserved	7:6	RO	0	Not Used
table_entry70	5:4	RW	0	The RSS_ring value for entry 70.
Reserved	3:2	RO	0	Not Used
table_entry71	1:0	RW	0	The RSS_ring value for entry 71.

Indirection Table Register 10 (offset: 0x654)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	Not Used
table_entry72	29:28	RW	0	The RSS_ring value for entry 72.
Reserved	27:26	RO	0	Not Used
table_entry73	25:24	RW	0	The RSS_ring value for entry 73.
Reserved	23:22	RO	0	Not Used
table_entry74	21:20	RW	0	The RSS_ring value for entry 74.
Reserved	19:18	RO	0	Not Used
table_entry75	17:16	RW	0	The RSS_ring value for entry 75.
Reserved	15:14	RO	0	Not Used
table_entry76	13:12	RW	0	The RSS_ring value for entry 76.
Reserved	11:10	RO	0	Not Used
table_entry77	9:8	RW	0	The RSS_ring value for entry 77.
Reserved	7:6	RO	0	Not Used
table_entry78	5:4	RW	0	The RSS_ring value for entry 78.
Reserved	3:2	RO	0	Not Used
table_entry79	1:0	RW	0	The RSS_ring value for entry 79.

Indirection Table Register 11 (offset: 0x658)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry80	29:28	RW	0	The RSS_ring value for entry 80.
Reserved	27:26	RO	0	Not Used
table_entry81	25:24	RW	0	The RSS_ring value for entry 81.
Reserved	23:22	RO	0	Not Used
table_entry82	21:20	RW	0	The RSS_ring value for entry 82.
Reserved	19:18	RO	0	Not Used
table_entry83	17:16	RW	0	The RSS_ring value for entry 83.
Reserved	15:14	RO	0	Not Used
table_entry84	13:12	RW	0	The RSS_ring value for entry 84.
Reserved	11:10	RO	0	Not Used
table_entry85	9:8	RW	0	The RSS_ring value for entry 85.
Reserved	7:6	RO	0	Not Used
table_entry86	5:4	RW	0	The RSS_ring value for entry 86.
Reserved	3:2	RO	0	Not Used
table_entry87	1:0	RW	0	The RSS_ring value for entry 87.

Indirection Table Register 12 (offset: 0x65C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry88	29:28	RW	0	The RSS_ring value for entry 88.
Reserved	27:26	RO	0	Not Used
table_entry89	25:24	RW	0	The RSS_ring value for entry 89.
Reserved	23:22	RO	0	Not Used
table_entry90	21:20	RW	0	The RSS_ring value for entry 90.
Reserved	19:18	RO	0	Not Used
table_entry91	17:16	RW	0	The RSS_ring value for entry 91.
Reserved	15:14	RO	0	Not Used
table_entry92	13:12	RW	0	The RSS_ring value for entry 92.
Reserved	11:10	RO	0	Not Used
table_entry93	9:8	RW	0	The RSS_ring value for entry 93.
Reserved	7:6	RO	0	Not Used
table_entry94	5:4	RW	0	The RSS_ring value for entry 94.

Name	Bits	Access	Default Value	Description
Reserved	3:2	RO	0	Not Used
table_entry95	1:0	RW	0	The RSS_ring value for entry 95.

Indirection Table Register 13 (offset: 0x660)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	Not Used
table_entry96	29:28	RW	0	The RSS_ring value for entry 96.
Reserved	27:26	RO	0	Not Used
table_entry97	25:24	RW	0	The RSS_ring value for entry 97.
Reserved	23:22	RO	0	Not Used
table_entry98	21:20	RW	0	The RSS_ring value for entry 98.
Reserved	19:18	RO	0	Not Used
table_entry99	17:16	RW	0	The RSS_ring value for entry 99.
Reserved	15:14	RO	0	Not Used
table_entry100	13:12	RW	0	The RSS_ring value for entry 100.
Reserved	11:10	RO	0	Not Used
table_entry101	9:8	RW	0	The RSS_ring value for entry 101.
Reserved	7:6	RO	0	Not Used
table_entry102	5:4	RW	0	The RSS_ring value for entry 102.
Reserved	3:2	RO	0	Not Used
table_entry103	1:0	RW	0	The RSS_ring value for entry 103.

Indirection Table Register 14 (offset: 0x664)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	Not Used
table_entry104	29:28	RW	0	The RSS_ring value for entry 104.
Reserved	27:26	RO	0	Not Used
table_entry105	25:24	RW	0	The RSS_ring value for entry 105.
Reserved	23:22	RO	0	Not Used
table_entry106	21:20	RW	0	The RSS_ring value for entry 106.
Reserved	19:18	RO	0	Not Used
table_entry107	17:16	RW	0	The RSS_ring value for entry 107.
Reserved	15:14	RO	0	Not Used

Name	*Bits*	*Access*	*Default Value*	*Description*
table_entry108	13:12	RW	0	The RSS_ring value for entry 108.
Reserved	11:10	RO	0	Not Used
table_entry109	9:8	RW	0	The RSS_ring value for entry 109.
Reserved	7:6	RO	0	Not Used
table_entry110	5:4	RW	0	The RSS_ring value for entry 110.
Reserved	3:2	RO	0	Not Used
table_entry111	1:0	RW	0	The RSS_ring value for entry 111.

Indirection Table Register 15 (offset: 0x668)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry112	29:28	RW	0	The RSS_ring value for entry 112.
Reserved	27:26	RO	0	Not Used
table_entry113	25:24	RW	0	The RSS_ring value for entry 113.
Reserved	23:22	RO	0	Not Used
table_entry114	21:20	RW	0	The RSS_ring value for entry 114.
Reserved	19:18	RO	0	Not Used
table_entry115	17:16	RW	0	The RSS_ring value for entry 115.
Reserved	15:14	RO	0	Not Used
table_entry116	13:12	RW	0	The RSS_ring value for entry 116.
Reserved	11:10	RO	0	Not Used
table_entry117	9:8	RW	0	The RSS_ring value for entry 117.
Reserved	7:6	RO	0	Not Used
table_entry118	5:4	RW	0	The RSS_ring value for entry 118.
Reserved	3:2	RO	0	Not Used
table_entry119	1:0	RW	0	The RSS_ring value for entry 119.

Indirection Table Register 16 (offset: 0x66C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
table_entry120	29:28	RW	0	The RSS_ring value for entry 120.
Reserved	27:26	RO	0	Not Used
table_entry121	25:24	RW	0	The RSS_ring value for entry 121.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	23:22	RO	0	Not Used
table_entry122	21:20	RW	0	The RSS_ring value for entry 122.
Reserved	19:18	RO	0	Not Used
table_entry123	17:16	RW	0	The RSS_ring value for entry 123.
Reserved	15:14	RO	0	Not Used
table_entry124	13:12	RW	0	The RSS_ring value for entry 124.
Reserved	11:10	RO	0	Not Used
table_entry125	9:8	RW	0	The RSS_ring value for entry 125.
Reserved	7:6	RO	0	Not Used
table_entry126	5:4	RW	0	The RSS_ring value for entry 126.
Reserved	3:2	RO	0	Not Used
table_entry127	1:0	RW	0	The RSS_ring value for entry 127.

Hash Key Register 0 (offset: 0x670)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash_key[7:0]	31:24	RW	0	The first byte of the hash_key. Bit31 is the first bit of the hash_key. It's the big endian format.
Hash_key[15:8]	23:16	RW	0	The 2nd byte of the hash_key. The bits are in the big endian format
Hash_key[23:16]	15:8	RW	0	The 3rd byte of the hash_key. The bits are in the big endian format
Hash_key[31:24]	7:0	RW	0	The 4th byte of the hash_key. The bits are in the big endian format

Hash Key Registers 1-8 (offset: 0x674-0x693)

The rest of the Hash Keys for 5th through 36th bytes. They follow the same format as above.

Hash Key Register 9 (offset: 0x694)

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash_key[295:288]	31:24	RW	0	The 37th byte of the hash_key. The bits are in the big endian format
Hash_key[303:296]	23:16	RW	0	The 38th byte of the hash_key. The bits are in the big endian format

Name	*Bits*	*Access*	*Default Value*	*Description*
Hash_key[311:304]	15:8	RW	0	The 39th byte of the hash_key. The bits are in the big endian format
Hash_key[319:312]	7:0	RW	0	The 40th byte of the hash_key. The bits are in the big endian format

Receive MAC Programmable IPv6 Extension Header Register (offset: 0x6A0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Programmable Extension Header Type #2 Enable	31	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [15:8] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [15:8]. This bit should be set to 0 if IPv6 RX is disabled.
Programmable Extension Header Type #1 Enable	30	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [7:0] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [7:0]. This bit should be set to 0 if IPv6 RX is disabled.
Reserved	29:16	RO	0	Reserved bits
Programmable Extension Header Type #2	15:8	R/W	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	R/W	0	These bits contain the programmable extension header value for programmable header #1.

Extended Magic Pack Registers

All registers reset are core reset unless specified.

Extended Magic Packet Length and Mask Register (offset: 0x6E0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended magic packet pattern length	31:16	RW	16'b0	Extended magic packet pattern length in byte. Only support up-to 8 byte in Logan.

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended magic packet byte mask	15:0	RW	16'b0	Byte-mask bit for extended magic packet pattern. 1: Corresponding pattern of byte will be masked out during packet comparison. 0: Corresponding pattern of byte will be enabled during packet comparison. Only meaningful from bit-7 to bit-0 in Logan.

Extended Magic Packet Pattern Register 0 (offset: 0x6E4)

When comparing with incoming network byte stream, N0N1N2N3N4N5. N0 is compared with B5 and N5 is compared with B0 in little endian format.

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended magic packet pattern byte 3	31:24	RW	8'b0	Extended magic packet pattern Byte 3.
Extended magic packet pattern byte 2	23:16	RW	8'b0	Extended magic packet pattern Byte 2.
Extended magic packet pattern byte 1	15:8	RW	8'b0	Extended magic packet pattern Byte 1.
Extended magic packet pattern byte 0	7:0	RW	8'b0	Extended magic packet pattern Byte 0.

Extended Magic Packet Pattern Register 1 (offset: 0x6E8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Extended magic packet pattern byte 7	31:24	RW	8'b0	Extended magic packet pattern Byte 7.
Extended magic packet pattern byte 6	23:16	RW	8'b0	Extended magic packet pattern Byte 6.
Extended magic packet pattern byte 5	15:8	RW	8'b0	Extended magic packet pattern Byte 5.
Extended magic packet pattern byte 4	7:0	RW	8'b0	Extended magic packet pattern Byte 4.

Statistics Registers

Transmit MAC Static Counters

ifHCOutOctets (offset: 0x800)

The number of octets transmitted out of the interface, including frame characters.

etherStatsCollisions (offset: 0x808)

The number of collisions experienced.

outXonSent (offset: 0x80C)

Sent Xon.

outXoffSent (offset: 0x810)

Sent Xoff.

dot3StatsInternalMacTransmitErrors (offset: 0x818)

A count of frames for which transmission on a particular interface fails due to an internal MAC sublayer transmit error.

dot3StatsSingleCollisionFrames (offset: 0x81C)

A count of successfully transmitted frames on a particular interface for which transmission is inhibited by exactly one collision.

dot3StatsMultipleCollisionFrames (offset: 0x820)

A count of successfully transmitted frames on a particular interface for which transmission is inhibited by more than one collision.

dot3StatsDeferredTransmissions (offset: 0x824)

A count of frames for which the first transmission attempt on a particular interface is delayed because of the medium is busy.

dot3StatsExcessiveTransmissions (offset: 0x82C)

A count of frames for which transmission on a particular interface fails due to excessive collisions.

dot3StatsLateCollisions (offset: 0x830)

The number of times that a collision is detected on a particular interface later than 512 bit-times into the transmission of a packet.

iHCOutUcastPkts (offset: 0x86C)

The number of packets that higher-level protocols requested be transmitted, and that were not addressed to a multicast or broadcast address at this sublayer, including those that were discarded or not sent.

iHCOutMulticastPkts (offset: 0x870)

The number of packets that higher-level protocols requested be transmitted, and that were addressed to a multicast address at this sublayer, including those that were discarded or not sent.

iHCOutBroadcastPkts (offset: 0x870)

The number of packets that higher-level protocols requested be transmitted, and that were addressed to a broadcast address at this sublayer, including those that were discarded or not sent.

Receive MAC Static Counters

ifHCOutOctets (offset: 0x880)

The number of octets received on the interface, including frame characters.

etherStatsFragments (offset: 0x888)

A frame size that is less than 64 bytes with a bad FCS.

ifHCInUcastPkts (offset: 0x88C)

The number of packets delivered by this sublayer to a higher sublayer, which were not addressed to a multicast or broadcast address at this sublayer.

ifHCInMulticastPkts (offset: 0x890)

The number of packets delivered by this sublayer to a higher sublayer, which were addressed to a multicast address at this sublayer.

ifHCInBroadcastPkts (offset: 0x894)

The number of packets delivered by this sublayer to a higher sublayer, which were addressed to a broadcast address at this sublayer.

dot3StatsFCSErrors (offset: 0x898)

A count of frames received on a particular interface that are an integral number of octets in length and do not pass the FCS check.

dot3StatsAlignmentErrors (offset: 0x89C)

A count of frames received on a particular interface that are not an integral number of octets in length and do not pass the FCS check.

xonPauseFrameReceived (offset: 0x8A0)

MAC control frames with pause command and length equal to zero.

xoffPauseFrameReceived (offset: 0x8A4)

MAC control frames with pause command and length greater than zero.

macControlFramesRecevied (offset: 0x8A8)

MAC control frames with no pause command.

xoffStateEntered (offset: 0x8AC)

Transmitting is disabled.

dot3StatsFramesTooLongs (offset: 0x8B0)

A count of frames received on a particular interface that exceeds the maximum permitted frame size.

etherStatsJabbers (offset: 0x8B4)

Frames exceed jabber time.

etherStatsUndersizePkts (offset: 0x8B8)

Frames with a size less than 64 bytes.

Send Data Initiator Registers

All registers reset are core reset unless specified.

Send Data Initiator Mode Register (offset: 0xC00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
Multiple Segment Enable	5	RW	0	Enable RDMA to read multisegment (up to four segments) in one DMA request during TCP segmentation
Pre-DMA Debug	4	RW	0	When this bit is set, Send Data Initiator state machine will be halted if the pre-DMA bit of the Send BD is set
Hardware Pre-DMA Enable	3	RW	0	Enable HW LSO pre-DMA processing
Stats Overflow Attn Enable	2	RW	0	Enable attention for statistics overflow
Enable	1	RW	1	This bit controls whether the Send Data Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read
Reset	0	RW	0	When this bit is set to 1, Send Data Initiator state machine is reset This is a self-clearing bit

Send Data Initiator Status Register (offset: 0xC04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Stats Overflow Attention	2	RO	0	A statistics managed by Send Data Initiator has overflowed
Reserved	1:0	RO	0	–

Send Data Initiator Statistics Control Register (offset: 0xC08)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Zap Statistic	4	RW	0	–
Flush Statistic	3	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Statistics Clear	2	RW	0	If set, resets local statistics counters to zero Clears only masked statistics Self-clearing when don
Faster Update	1	RW	0	–
Statistics Enable	0	RW	0	When set, allows the local statistics counters to increment. When reset, counters hold their values until next update to NIC memory Enables only masked statistics

Send Data Initiator Statistics Mask Register (offset: 0xC0C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	–
Counters Enable Mask	0	RW	0	Controls whether Class of Service 0 statistics can be updated, cleared, or flushed

Send Data Initiator Statistics Increment Mask Register (offset: 0xC10)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
Counters Increment Mask	23:19	WO	0	Writing 1 to the bit position forces the corresponding statistics counters to increment by 1. Not affected by Statistics Enable Mask Bits 16:23 correspond to Set Send Producer Index, Status Updated, Interrupts, Avoided Interrupts, Send Threshold Hit respectively
Reserved	18:16	RO	0	–
Counters Increment Mask	15:0	WO	0	Writing 1 to the bit position forces the corresponding statistics counters to increment by 1. Not affected by Statistics Enable Mask Bits 15:0 correspond to statistics for Class of Service 16:1

Local Statistics Register (offset: 0xC80 – 0xCDF)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–

Name	Bits	Access	Default Value	Description
Counter Value	9:0	RO	0	The current counter value for statistics kept by the Send Data Initiator

TCP Segmentation Control Registers

All registers reset are core reset unless specified.

Lower Host Address Register for TCP Segmentation (offset: 0xCE0)

Name	Bits	Access	Default Value	Description
Lower Host Address	31:0	RW	0	Specifies the lower 32bits of the starting address in host memory where the transmit data buffer resides

Upper Host Address Register for TCP Segmentation (offset: 0xCE4)

Name	Bits	Access	Default Value	Description
Upper Host Address	31:0	RW	0	Specifies the upper 32bits of the starting address in host memory where the transmit data buffer resides

Length/Offset Register for TCP Segmentation (offset: 0xCE8)

Name	Bits	Access	Default Value	Description
Reserved	31:23	RO	0	–
MBUF Offset	22:16	RW	0	MBUF offset It specifies the offset of the first TXMBUF at where DMA starts putting data. The valid value is between 48 and 128
Length	15:0	RW	0	Specifies the length of data to be transmitted. Although FW can specify up to 64KB, it should not attempt to program more than 8KB because it would exceed the size of TXMBUF

DMA Flag Register for TCP Segmentation (offset: 0xCEC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:20	RO	–	–
MBUF offset valid	19	RW	–	MBUF offset valid When this bit is set, the RDMA engine will DMA the data into the TXMBUF starting at an offset specified in the Length/Offset register
Last Fragment	18	RW	–	Last fragment. This bit is passed transparently to the SDC. When this bit is set, the SDC will inform the HC to increment the Send Ring Consumer Index. The bit is always set by HW if no FW assisted TCP segmentation occurs Otherwise, FW sets it at the end of fragmentation
No Word Swap	17	RW	–	No Word Swap Set to disable endian word swap on data from PCIe bus
Status_dma	16	RW	–	–
MAC source address Select	15:14	RW	–	This 2-bit field determines which of the four MAC addresses should be inserted into the frame
MAC source address insertion	13	RW	–	Indicates that the predetermined source address is inserted into the Ethernet header of the frame
TCP/UDP checksum enable	12	RW	–	TCP/UDP Checksum enable
IP Checksum enable	11	RW	–	IP checksum enable
Force RAW checksum enable	10	RW	–	Force RAW checksum enable
Data_only	9	RO	–	–
Header	8	RW	–	–
VLAN Tag Present	7	RW	–	VLAN Tag present Indicates that the VLAN tag should be copied to the Frame Header by the DMA engine
Force Interrupt	6	RW	–	Following the completion of this DMA, a host interrupt is generated
Last BD in Frame	5	RW	–	Last BD in frame
Coalesce Now	4	RW	–	Pass through Send Buffer Descriptor flag
mbuf	3	RW	–	–
Invoke Processor	2	RW	–	Clears the Pass bit of the entry queued to the SDCQ, so that SDC will invoke CPU: • If the packet is created by HW, this bit will be the same as bit 9 of the flag field in Send BD. • If the packet is created by FW, it will be up to CPU whether it needs to post-process the data.

Name	***Bits***	***Access***	***Default Value***	***Description***
Don't Generate CRC	1	RW	–	Do not generate CRC Pass through Send Buffer Descriptor flag
No Byte Swap	0	RW	–	Set to disable endian byte swap on data from PCIe bus

VLAN Tag Register for TCP Segmentation (offset: 0xCF0)

Name	***Bits***	***Access***	***Default Value***	***Description***
Reserved	31:16	RO	0	–
VLAN Tag	15:0	RW	0	VLAN Tag to be inserted into the Frame Header if bit 7 of DMA Flags register is set

Pre-DMA Command Exchange Register for TCP Segmentation (offset: 0xCF4)

Name	***Bits***	***Access***	***Default Value***	***Description***
READY	31	RW	0	The CPU sets this bit to tell SDI that DMA address, length, flags, and VLAN tag are valid and request is read to go. The CPU polls this bit to be clear for the completion of request
PASS Status	30	RW	1	If this bit is set to 0, the CPU will be responsible for processing the buffer descriptor
SKIP Status	29	RW	0	The CPU sets this bit to 1 to inform the SDI that the TCP Segmentation is completed, and the BD_Index can be incremented
Unsupported_Mss Status	28	RO	0	–
Reserved	27:7	RO	0	–
BD Index	6:0	RO	0	The internal current buffer descriptor pointer that the HW/FW is servicing

EAV: Real-TimeSend Data Initiator Registers

All registers reset are core reset unless specified.

EAV: Real-TimeSend Data Initiator Mode Register (offset: 0xD00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
Multiple Segment Enable	5	RW	0	Enable RDMA to read multi-segment (up to four segments) in one DMA request during TCP segmentation
Pre-DMA Debug	4	RW	0	When this bit is set, Send Data Initiator state machine will be halted if the pre-DMA bit of the Send BD is set
Hardware Pre-DMA Enable	3	RW	0	Enable HW LSO pre-DMA processing
Stats Overflow Attn Enable	2	RW	0	Enable attention for statistics overflow
Enable	1	RW	1	This bit controls whether the Send Data Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read
Reset	0	RW	0	When this bit is set to 1, Send Data Initiator state machine is reset This is a self-clearing bit

EAV: Real-TimeSend Data Initiator Status Register (offset: 0xD04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Stats Overflow Attention	2	RO	0	A statistics managed by Send Data Initiator has overflowed
Reserved	1:0	RO	0	–

EAV: Real-TimeSend Data Initiator Statistics Control Register (offset: 0xD08)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Zap Statistic	4	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Flush Statistic	3	RW	0	–
Statistics Clear	2	RW	0	If set, resets local statistics counters to zero Clears only masked statistics Self-clearing when don
Faster Update	1	RW	0	–
Statistics Enable	0	RW	0	When set, allows the local statistics counters to increment. When reset, counters hold their values until next update to NIC memory Enables only masked statistics

EAV: Real-TimeSend Data Initiator Statistics Mask Register (offset: 0xD0C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	–
Counters Enable Mask	0	RW	0	Controls whether Class of Service 0 statistics can be updated, cleared, or flushed

EAV: Real-TimeSend Data Initiator Statistics Increment Mask Register (offset: 0xD10)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
Counters Increment Mask	23:19	WO	0	Writing 1 to the bit position forces the corresponding statistics counters to increment by 1. Not affected by Statistics Enable Mask Bits 16:23 correspond to Set Send Producer Index, Status Updated, Interrupts, Avoided Interrupts, Send Threshold Hit respectively
Reserved	18:16	RO	0	–
Counters Increment Mask	15:0	WO	0	Writing 1 to the bit position forces the corresponding statistics counters to increment by 1. Not affected by Statistics Enable Mask Bits 15:0 correspond to statistics for Class of Service 16:1

EAV: AV FETCH DELAY (offset: 0xD20)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0x0	–
AVFetch Delay	20:15	RW	0x8	This value is multiplied by 32.768 µs to determine the AVFetch Delay value.
Reserved	14:1	RO	0x0	–
Disable SBD Fetch Delay	0	RW	0x0	When this bit is written 1, Real-TimeSend Data will likely be immediately fetched from host memory after being posted. This is a debug mode. Best Effort Ring performance might get a significant hit when this bit is set.

EAV: AV FETCH CX COMP Register (offset:0xD24)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0x0	–
AV Fetch CX Comp	20:15	RW	0x0	This value is multiplied by **32.768µs** to determine the Real-TimeFetch CX Compensator value. Write a 0x0 to ignore any effect of CPU CX state exit latency
Reserved	14:0	RO	0x0	–

EAV: AV FETCH L1 COMP Register (offset: 0xD28)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0x0	–
AV Fetch L1 Comp	20:15	RW	0x0	This value is multiplied by 32.768 µs to determine the Real-TimeFetch L1 Compensator value.
Reserved	14:2	RO	0x0	–
Override automatic L1 exit delay value	1	RW	0	Hardware applies appropriate L1 exist latency value unless a 1 is written to this bit. If this bit is set to one, the [20:15] field of this register must be programmed to an appropriate L1 exist latency value.
Enable L1 Comp	0	RW	0	Write a 1 to enable compensation for PCIe L1 exit latency in the AV Fetch Window

Send Data Completion Control Registers

All registers reset are core reset unless specified.

Send Data Completion Mode Register (offset: 0x1000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:2	RO	0	–
Enable	1	RW	0	This bit controls whether Send Data Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read
Reset	0	RW	0	When this is set to 1, the Send Data Completion state machine is reset This is a self-clearing bit

Pre-DMA Command Exchange Register for TCP Segmentation (offset: 0x1008)

Name	*Bits*	*Access*	*Default Value*	*Description*
PASS	31	RW	1	If this bit is set to 0, the CPU will be invoked to process TXMBUF data. It is same as SDCQ bit 143.
SKIP	30	RW	0	CPU Sets this bit to 1 to inform the SDC that the post-processing is completed and hardware can resume operation
End of Fragmentation	29	RW	1	If this bit is set to 1, SDC will request the HC to increment Send Ring Consumer Index when CPU sets the SKIP bit. It is same as SDCQ bit 12
Reserved	28:12	RO	0	–
Head TXMBUF pointer	11:6	RW	0	Head TXMBUF Pointer They are same as SDCQ bits 11:6
Tail TXMBUF pointer	5:0	RW	0	Tail TXMBUF Pointer They are same as SDCQ bits 5:0

EAV: Real-TimeSend Data Completion Mode Register (offset: 0x1080)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:2	RO	0	–
Enable	1	RW	0	This bit controls whether Send Data Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read
Reset	0	RW	0	When this is set to 1, the Send Data Completion state machine is reset This is a self-clearing bit

EAV: Real-TimePre-DMA Command Exchange Register for TCP Segmentation (offset: 0x1088)

Name	*Bits*	*Access*	*Default Value*	*Description*
PASS	31	RO	1	If this bit is set to 0, the CPU will be invoked to process TXMBUF data. It is same as SDCQ bit 143.
SKIP	30	RW	0	CPU Sets this bit to 1 to inform the SDC that the post-processing is completed and hardware can resume operation
End of Fragmentation	29	RO	1	If this bit is set to 1, SDC will request the HC to increment Send Ring Consumer Index when CPU sets the SKIP bit. It is same as SDCQ bit 12
Reserved	28:12	RO	0	–
Head TXMBUF pointer	11:6	RO	0	Head TXMBUF Pointer They are same as SDCQ bits 11:6
Tail TXMBUF pointer	5:0	RO	0	Tail TXMBUF Pointer They are same as SDCQ bits 5:0

Send BD Selector Control Registers

All registers reset are core reset unless specified.

Send BD Ring Selector Mode Register (offset: 0x1400)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs
Enable	1	RW	0	This bit controls whether Send BD Ring Selector state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read
Reset	0	RW	0	When this is set to 1, the Send BD Ring Selector State machine is reset This is a self clearing bit

Send BD Ring Selector Status Register (offset: 0x1404)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	0	Send BD Ring Selector error status
Reserved	1:0	RO	0	–

Send BD Ring Selector Hardware Diagnostics Register (offset: 0x1408)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

Send BD Ring Selector Local NIC Send BD Consumer Index Register (offset: 0x1440 – 0x147C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Index	8:0	RO		These bits contain the current NIC send BD index

EAV: Real-TimeSend BD Ring Selector Mode Register (offset: 0x1480)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	0	This bit controls whether Send BD Ring Selector state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read.
Reset	0	RW	0	When this is set to 1, the Send BD Ring Selector State machine is reset. This is a self clearing bit.

EAV: Real-TimeSend BD Ring Selector Status Register (offset: 0x1484)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	0	Send BD Ring Selector error status
Reserved	1:0	RO	0	–

EAV: Real-TimeSend BD Ring Selector Hardware Diagnostics Register (offset: 0x1488)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

EAV: Real-TimeSend BD Ring Selector Local NIC Send BD Consumer Index Register (offset: 0x14C0 – 0x14FC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
Index	8:0	RO		These bits contain the current NIC send BD index

Send BD Initiator Control Registers

All registers reset are core reset unless specified.

Send BD Initiator Mode Register (offset: 0x1800)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Pass_bit status	4	RO	1	Unused
Sbdi_rupd_enable	3	RW	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	0	This bit controls whether the Send BD Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read.
Reset	0	RW	0	When this is set to 1, the Send BD Initiator State machine is reset. This is a self clearing bit.

Send BD Initiator Status Register (offset: 0x1804)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	0	Send BD Initiator Error
Reserved	1:0	RO	0	–

Send BD Diagnostic Initiator Local NIC BD N Producer Index Registers (offset: 0x1808 – 0x1844)

This set of registers is used to keep track of the current DMAs queued to move send BDs from the host to the NIC.

Send BD Diagnostic Initiator Local NIC BD N Producer Index Registers (offset:

SEND_BD_FETCH_THRESHOLD_REG (Offset: 0x1850)

This threshold is introduced to limit the b/w in fetching SBD on Ring#1. This does not influence Ring#2.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0x00	Unused
Send Ring#1 BD Fetch Threshold	5:0	RW	0x1F	The value programmed in this field sets a cap to the number of SBDs that would be fetched by a single DMA transaction (although there are other factors which might further limit the DMA size). This parameter applies only to Send Ring#1

EAV: Real-TimeSend BD Initiator Mode Register (offset: 0x1880)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Pass_bit status	4	RO	1	Unused
Sbdi_rupd_enable	3	RW	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs
Enable	1	RW	0	This bit controls whether the Send BD Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read
Reset	0	RW	0	When this is set to 1, the Send BD Initiator State machine is reset This is a self clearing bit

EAV: Real-TimeSend BD Initiator Status Register (offset: 0x1884)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	0	Send BD Initiator Error
Reserved	1:0	RO	0	–

EAV: Real-TimeSend BD Diagnostic Initiator Local NIC BD N Producer Index Registers (offset: 0x1888 – 0x18C4)

This set of registers is used to keep track of the current DMAs queued to move send BDs from the host to the NIC.

Send BD Completion Control Registers

All registers reset are core reset unless specified.

Send BD Completion Mode Register (offset: 0x1C00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs
Enable	1	RW	0	This bit controls whether the Send BD Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read
Reset	0	RW	0	When this is set to 1, the Send BD Completion State machine is reset This is a self clearing bit

Send BD Completion Debug Register (offset: 0x1C04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Rstate	2:0	RO	0	SBDC State (0 is idle)

EAV: Real-TimeSend BD Completion Mode Register (offset: 0x1C80)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs
Enable	1	RW	0	This bit controls whether the Send BD Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read
Reset	0	RW	0	When this is set to 1, the Send BD Completion State machine is reset This is a self clearing bit

EAV: Real-TimeSend BD Completion Debug Register (offset: 0x1C84)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Rstate	2:0	RO	0	SBDC State (0 is idle)

Receive List Placement Registers

All registers reset are core reset unless specified.

Receive List Placement Mode Register (offset: 0x2000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Stats Overflow Attention Enable	4	RW	–	Enable attention for statistics overflow
Mapping out of Range Attention Enable	3	RW	–	Enable attention for mapping out of range error
Class Zero Attention Enable	2	RW	–	Enable attention for zero class field
Enable	1	RW	1	This bit controls whether the Receive List Placement state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read
Reset	0	RW	0	When this bit is set to 1, the Receive List Placement state machine is reset. This is a self clearing bit

Receive List Placement Status Register (offset: 0x2004)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Stats Overflow Attention	4	RO	–	A statistics managed by Receive List Placement has overflowed
Mapping out of Range Attention	3	RO	–	Class of service mapping is out of the range of the active queue number
Class Zero Attention	2	RO	–	Class field extracted from frame descriptor is zero
Reserved	1:0	RO	0	–

Receive Selector Non-Empty Bits Register (offset: 0x200C)

This 32-bit register is used by the RISCs to quickly determine the status of the receive selector. Bit 0 refers to receive selector list 1. Bit 15 refers to receive selector list 16. If this register is nonzero the receive selector nonempty bit is set in the RXCPU event register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
List Non-Empty Bits	15:0	RO	–	If set, the bit indicates that the associated list is not empty (that is the counter is non-zero).

Receive List Placement Configuration Register (offset: 0x2010)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:15	RO	0	–
Default Interrupt Distribution Queue	14:13	RW	0	Default interrupt distribution queue. Number within a class of service group when the frame has errors, is truncated, or is a non-IP frame.
Bad Frames Class	12:8	RO	1	Default class for error or truncated frames. These frames are placed in this class of service group when the Allow Bad Frame bit (bit 11) is set in the Mode Control Register.
Number of Active Lists	7:3	RW	0	The total number of active receive lists. The value must be between 1 and 16. This value must be an integer multiple of the Number of Lists per Distribution Group value.
Number of Lists per Distribution Group	2:0	RW	0	Specifies the number of lists per interrupt distribution group. This register must always be a power of 2. For example, if the system wants four classes of service and four interrupt distribution lists per class of service, this value is set to four and the Number of Active Lists value is set to 16.

Receive List Placement Statistics Control Register (offset: 0x2014)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Statistics Clear	2	RW	0	When set, resets local statistics counters to zero. Clears only masked statistics Self-clearing when done
Reserved	1	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Statistics Enable	0	RW	0	When set, allow the local statistics counters to increment. When reset, counters hold their values until the next update to the NIC memory. Enables only masked statistics

Receive List Placement Statistics Enable Mask Register (offset: 0x2018)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	–
RSS_Priority	25	RW	0x0	This bit enables the receive packet to choose receive return ring in terms of RSS hash value instead of RC class when both RSS and RC rules are matched. Default is to give priority to RC.
RC Return Ring Enable	24	RW	0x0	1: Enable receive packet to use RC rule class as return ring number if RC rule is matched. This bit will be used in conjunction with bit25 to derive the final receive return ring. 0: Disable receive packet to use RC rule class as return ring number. Receive packet only uses RSS hash to select the receive return ring. If no any RSS hash types are applied, the default ring 0 will be used.
CPU MACTQ Priority Disable	23	RW	0x0	1: Disable CPU priority over SDC when arbitrating the MACTQ write requests 0: Enable CPU priority over SDC when arbitrating the MACTQ write requests
Reserved	22	RW	0x1	–
Enable inErrors stats	21	RW	0x1	1: Enabled 0: Disabled
Enable inDiscards stats	20	RW	0x1	1: Enabled 0: Disabled
Enable no_more_RBD stats	19	RW	0x1	1: Enabled 0: Disabled
Reserved	18:3	RW	0XFFFF	(Formerly ASIC ID – See Register 0x3658)
PERST_L	2	RO	0x0	Current state of PERST_L pin input 1: PERST_L is deasserted 0: PERST_L is asserted

Name	Bits	Access	Default Value	Description
A1 Silent indication	1	RO	0x0	1: Not silent 0: Silent Chip is A1 if ASIC rev id is A1 and this bit is 1 Chip is A1_silent if ASIC rev id is A1 and this bit is 0 Chip is A2 if ASIC rev id is A2 and this bit does not care
Enable COS stats	0	RW	0x1	1: Enabled 0: Disabled

Receive List Placement Statistics Increment Mask Register (offset: 0x201C)

Name	Bits	Access	Default Value	Description
Reserved	31:22	RO	0	–
Counters Increment Mask	21:16	WO	0	Writing a 1 to a Counters Increment Mask bit forces the corresponding statistics counter to increment by 1. Not affected by Statistics Enable Mask. Bits 16-21 correspond to statistics for Drop due to filter, DMA Write Queue Full, DMA High Priority Write Queue Full, No More Receive BD, ifInDiscards, and ifInErrors.
Reserved	15:1	RO	0	–
Counters Increment Mask	0	WO	0	Writing a 1 to a Counters Increment Mask bit forces the corresponding statistics counter to increment by 1. Not affected by Statistics Enable Mask. Bit 0 corresponds to statistics Class of Service 1.

Receive Selector List Head and Tail Pointers (offset: 0x2100)

The 16 receive selector lists head and tail pointers are MBUF cluster pointers. The selector list head pointer is the MBUF cluster pointer of the first frame queued in the associated selector list. Similarly, the selector list tail pointer is the MBUF cluster pointer of the last frame queued in that selector list.

Receive Selector List Count Registers (Offset of List N: 0X2108 + 16*[N-1])

These registers indicate how many frames are currently queued to the associated selector list.

Local Statistics Counter Registers (offset: 0x2200-0x2258)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
Counters Value	9:0	RO	–	The current counter value for statistics kept by the Receive List Placement

Receive Data and Receive BD Initiator Control Registers

All registers reset are core reset unless specified.

Receive Data and Receive BD Initiator Mode Register (offset: 0x2400)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Illegal Return Ring Size	4	RW	–	Enables illegal return ring size attention
Frame Size is too large to fit into one Receive BD	3	RW	–	Enables frame size is too large to fit into one Receive BD attention
Reserved	2	RO	0	–
Enable	1	RW	1	This bit controls whether the Receive Data and Receive BD Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Receive Data and Receive BD Initiator state machine is reset. This is a self-clearing bit.

Receive Data and Receive BD Initiator Status Register (offset: 0x2404)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
Illegal Return Ring Size	4	RO	–	One of the return rings contains illegal ring size (e.g., only contains 1024 entries)
Frame size is too large to fit into one Receive BD	3	RO	–	The received frame size is too big for the selected Receive BD
Reserved	2:0	RO	0	–

Jumbo Receive BD Ring RCB Registers (offset: 0x2440)

These registers mirror the functionality of those at 0x2450 for the standard producer ring.

Standard Receive BD Ring RCB Registers

Receive Producer Ring Host Address High Register (offset: 0x2450)

Name	*Bits*	*Access*	*Default Value*	*Description*
Host Address High	31:0	RW	0	The host ring address is the host address of the first ring element. The host ring address is in host address format

Receive Producer Ring Host Address Low Register (offset: 0x2454)

Name	*Bits*	*Access*	*Default Value*	*Description*
Host Address Low	31:0	RW	0	The host ring address is the host address of the first ring element. The host ring address is in host address format

Receive Producer Length/Flags Register (offset: 0x2458)

Name	*Bits*	*Access*	*Default Value*	*Description*
Ring Size	31:16	RW	0	Control number of elements in the Receive Producer Ring. Valid values are: 512 256 128 64 32
Maximum Ethernet Frame Length	15:2	RW	0	Unused for jumbo rings. Otherwise, specifies the maximum size of an Ethernet frame plus VLAN tag.
Disable Ring	1	RW	0	Set to disable the use of the ring
Reserved	0	RO	0	Reserved

Receive Producer Ring NIC Address Register (offset: 0x245C)

Name	*Bits*	*Access*	*Default Value*	*Description*
NIC Address	31:0	RW	0	The NIC ring address is the NIC address of the first ring element. Unlike most NetXtreme® controllers that use the hardware default value in this field, for the BCM5725/BCM5762/BCM57767, the driver must program 0x6000 here (and 0x7000 for the equivalent jumbo ring setting in register 0x244C).

Receive Diagnostic Data and Receive BD Ring Initiator Local NIC Standard Receive BD Consumer Index (offset: 0x2474)

This set of registers keeps track of the current DMAs queued to move receive data from the NIC to the host. The Receive Data and Receive BD Initiator maintains the state of the indices by keeping two local copies, a copy of the NIC's return ring producer index, and a copy of the NIC's receive BD consumer index. The local return ring producer index is set to the value placed in the DMA descriptor. The local NIC receive return consumer index is also set to the value placed in the DMA descriptor.

Receive Diagnostic Data and Receive BD Ring Initiator Local NIC Mini Receive BD Consumer Index (offset: 0x2478)

These registers are not applicable.

Receive Data and Receive Diagnostic BD Initiator Local Receive Return Producer Index Register (offset: 0x2480 – 0x24BC)

The registers from 0x2484 to 0x24BC are not applicable.

Receive Data and Receive BD Initiator Hardware Diagnostic Register (offset: 0x24C0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Diagnostics	31:0	RO	0x08000001	Hardware Diagnostics

Receive Data Completion Control Registers

All registers reset are core reset unless specified.

Receive Data Completion Mode Register (offset: 0x2800)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	1	This bit controls whether the Receive Data Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Receive Data Completion state machine is reset. This is a self-clearing bit.

Receive BD Initiator Control Registers

All registers reset are core reset unless specified.

Receive BD Initiator Mode Register (offset: 0x2C00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Receive BDs available on a disabled Receive BD Ring Enable	2	RW	0	Attention enable for Receive BDs available on a disabled Receive BD ring.
Enable	1	RW	1	This bit controls whether the Receive BD Initiator state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Receive BD Initiator state machine is reset. This is a self-clearing bit.

Receive BD Initiator Status Register (offset: 0x2C04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Receive BDs available on a disabled Receive BD Ring status	2	RO	0	Host requests to DMA Receive BDs to a disabled ring.
Reserved	1:0	RO	0	–

Receive BD Initiator Local NIC Receive BD Producer Index Register (offset: 0x2C08 – 0x2C13)

This set of registers is used to keep track of the current DMAs queued to move receive BDs from the host to the NIC.

Standard Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C18)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
BD Number	9:0	RW	0	Number of buffer descriptors indicated by the receive producer index for the DMA engine to initiate a transfer of buffer descriptors for replenishing the ring.

Jumbo Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C1C)

Standard Ring Replenish Watermark Register (offset: 0x2d00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
BD Number	9:0	RW	0	Number of buffer descriptors indicated by the receive producer index for the DMA engine to initiate a transfer of buffer descriptors for replenishing the ring.

Jumbo Ring Replenish Watermark Register (offset: 0x2d04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
BD Number	9:0	RW	0	Number of buffer descriptors indicated by the receive producer index for the DMA engine to initiate a transfer of buffer descriptors for replenishing the ring.

Receive BD Completion Control Registers

All registers reset are core reset unless specified.

Receive BD Completion Mode Register (offset: 0x3000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	1	This bit controls whether the Receive BD Completion state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Receive BD Completion state machine is reset. This is a self-clearing bit.

Receive BD Completion Status Register (offset: 0x3004)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	0	Receive BD Completion Error Status
Reserved	1:0	RO	0	–

NIC Jumbo Receive BD Producer Index Register (offset: 0x3008)

This register is not applicable.

NIC Standard Receive BD Producer Index Register (offset: 0x300C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
NIC Standard Receive BD Producer Index	8:0	RW	–	–

NIC Mini Receive BD Producer Index Register (offset: 0x3010)

This register is not applicable.

CPMU Registers

The following describes the registers which are required by BCM5725/BCM5762/BCM57767 CPMU specifications for configuration.

CPMU Control Register (offset: 0x3600)

This register is reset by POR Reset except for bit 2, 6, 7, 11, 12, 13.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31	R/W	0x0	Mission serdes test mode pipe_mdio_reset_n (inverted, set 1 to assert reset)
Reserved	30	R/W	0x0	Mission serdes test mode pipe_reset_n (inverted, set 1 to assert reset)
Reserved	29	R/W	0x0	Reserved
Always force GPHY DLL on	28	R/W	0x0	Always force the GPHY DLL on whenever this bit is set and the chip is not in low power mode 1: Enable 0: Disable
Enable GPHY powerdown in D0u	27	R/W	0x0	Enable CPMU to powerdown GPHY when the device enters D0u 1: Enable 0: Disable
Reserved	26:19	R/W	0x0	Reserved
Reserved	25	RW	0x0	Reserved
Reserved	24	R/W	0x0	Reserved
Reserved	23	R/W	0x0	Reserved
SW Controlled APE Reset	22	R/W SC	0x0	SW controlled APE reset 1: In reset 0: Out of reset
SW Controlled Core Reset	21	R/W SC	0x0	SW controlled core reset 1: In reset 0: Out of reset
Media Sense Power Mode Enable	20	R/W	0x0	Media Sense Power Mode Enable 1: Enable 0: Disable
Reserved	19	R/W	0x0	–
Legacy Timer Enable	18	R/W	0x0	This bit controls cpmu_legacy_timer_enable output.

Name	*Bits*	*Access*	*Default Value*	*Description*
Frequency Multiplier Enable	17	R/W	0x1	Frequency Multiplier Enable 1: Enable 0: Disable
GPHY 10MB Receive Only Mode Enable	16	R/W	0x1	Enables GPHY 10MB Receive Only Mode when this bit is set to 1
Play Dead Mode Enable	15	R/W	0x0	Play Dead Power Mode Enable 1: Enable 0: Disable
Link Speed Power Mode Enable	14	R/W	0x0	Enable clock adjustment based on the link speed in mission mode
Hide PCIe Function	13:11	R/W	0x0	SW controlled bits to hide PCIe functions. These bits are cleared by a rising edge of PERST_L 000: PCIe Functions 3,2,1 are on 001: Hide PCIe Function 1 010: Hide PCIe Function 2 100: Hide PCIe Function 3 111: Hide PCIe functions 1,2,3 110: Hide PCIe Functions 3 and 2 011: Hide PCIe Functions 1 and 2
Link Aware Power Mode Enable	10	R/W	0x0	Link Aware Power Mode Enable 1 : Enable 0 : Disable
Link Idle Power Mode Enable	9	R/W	0x0	Link Idle Power Mode Enable 1 : Enable 0 : Disable
Card Reader Idle Enable	8	R/W	0x0	Card Read Idle Power Mode Enable 1: Enable 0: Disable
Card Read IDDQ	7	R/W	0x0	SW controlled card reader IDDQ. This bit is cleared by a rising edge of PERST_L.
LAN IDDQ	6	R/W	0x0	SW controlled LAN IDDQ. This bit is cleared by a rising edge of PERST_L.
APE Deep Sleep Mode Enable	5	R/W	0x0	Enable APE deep sleep power management mode

Name	*Bits*	*Access*	*Default Value*	*Description*
APE Sleep Mode Enable	4	R/W	0x0	Enable APE sleep power management mode
Reserved	3	RO	0x0	–
Power-down	2	R/W	0x0	Legacy Address: 0x6804:[20] Force CPMU into Low Power State, LAN function will be powered down This bit is cleared by a rising edge of PERST_L
CPMU Register Software Reset	1	R/W SC	0x0	Software reset for resetting all the registers to default
CPMU Software Reset	0	R/W SC	0x0	Software reset for all the CPMU logic expect for registers

Link Speed 10MB/No Link Power Mode Clock Policy Register (offset: 0x3604)

This register is reset by POR Reset or CPMU Register Software Reset. Please note that clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x17	Software Controlled MAC Core Clock Speed Select: 00000: Core = 62.5 MHz (GPHY DLL/2) 10001: Core = 12.5 MHz (CK25/2) 10011: Core = 6.25 MHz (CK25/4) 10101: Core = 3.125 MHz (CK25/8) 10111: Core = 1.563 MHz (CK25/16) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
Reserved	15:0	R/O	0x0	–

Link Speed 100MB Power Mode Clock Policy Register (offset: 0x3608)

This register is reset by POR Reset or CPMU Register Software Reset. Please note that clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x11	Software Controlled MAC Core Clock Speed Select: 00000: Core = 62.5 MHz (GPHY DLL/2) 10001: Core = 12.5 MHz (CK25/2) 10011: Core = 6.25 MHz (CK25/4) 10101: Core = 3.125 MHz (CK25/8) 10111: Core = 1.563 MHz (CK25/16) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
Reserved	15:0	R/O	0x0	–

Link Speed 1000Mb Power Mode Clock Policy Register (offset: 0x360C)

This register is reset by POR Reset or CPMU Register Software Reset. Please note that clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x0	Software Controlled MAC Core Clock Speed Select: 00000: Core = 62.5 MHz (GPHY DLL/2) 10001: Core = 12.5 MHz (CK25/2) 10011: Core = 6.25 MHz (CK25/4) 10101: Core = 3.125 MHz (CK25/8) 10111: Core = 1.563 MHz (CK25/16) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
Reserved	15:0	R/O	0x0	–

Link Aware Power Mode Clock Policy Register (offset: 0x3610)

This register is reset by POR Reset or CPMU Register Software Reset. Please note that clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x13	Software Controlled MAC Core Clock Speed Select: 00000: Core = 66.67 MHz (PLL/2) 10001: Core = 12.5 MHz (CK25/2) 10011: Core = 6.25 MHz (CK25/4) 10101: Core = 3.125 MHz (CK25/8) 10111: Core = 1.563 MHz (CK25/16) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
Reserved	15:13	R/O	0x0	–
APE Clock Switch	12:8	R/W	0x11	Software Controlled APE Clock Speed Select 00000: 133MHz (PLL) 10001: 25.0MHz (CK25) 10011: 12.5MHz (CK25/2) 10101: 6.25MHz (CK25/4) 10111: 3.125MHz (CK25/8) 11001: 1.563MHz (CK25/16) 11110: PLL 133MHz/2
Reserved	7:0	R/O	0x0	–

D0u Clock Policy Register (offset: 0x3614)

This register is reset by POR Reset or CPMU Register Software Reset. Please note that clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x13	Software Controlled MAC Core Clock Speed Select: 00001: Core = 60.0 MHz (Alt Source/2) 00011: Core = 30.0 MHz (Alt Source/4) 00101: Core = 15.0 MHz (Alt Source/8) 00111: Core = 7.5 MHz (Alt Source/16) 01001: Core = 3.75 MHz (Alt Source/32) 10001: Core = 12.5 MHz (CK25/2) 10011: Core = 6.25 MHz (CK25/4) 10101: Core = 3.125 MHz (CK25/8) 10111: Core = 1.563 MHz (CK25/16) 11001: Core = 781 kHz (CK25/32) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2)
Reserved	15:0	R/O	0x0	–

Link Idle Power Mode Clock Policy Register (offset: 0x3618)

This register is reset by POR Reset or CPMU Register Software Reset.

Note: Clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x13	Software Controlled MAC Core Clock Speed Select: 00000: Core = 66.67MHz (PLL/2) 10001: Core = 12.5MHz (CK25/2) 10011: Core = 6.25MHz (CK25/4) 10101: Core = 3.125MHz (CK25/8) 10111: Core = 1.563MHz (CK25/16) 11111: Core = 12.5MHz/1.25MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
Reserved	15:0	R/O	0x0	–

APE Sleep State Clock Policy Register (offset: 0x3620)

This register is reset by POR Reset or CPMU Register Software Reset.

Note: Clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
APE Sleep HCLK Disable	31	R/W	0x1	Software Controlled APE HCLK shutoff in sleep and deep sleep state
Reserved	30:21	DC	0x000	–

Name	*Bits*	*Access*	*Default Value*	*Description*
APE Deep Sleep FCLK Switch	20:16	R/W	0x11	Software Controlled APE Clock Speed Select 00000: 133MHz (PLL) 10001: 25.0MHz (CK25) 10011: 12.5MHz (CK25/2) 10101: 6.25MHz (CK25/4) 10111: 3.125MHz (CK25/8) 11001: 1.563MHz (CK25/16) 11110: PLL 133MHz/2
Reserved	15:5	DC	0x000	–
APE Sleep FCLK Switch	4:0	R/W	0x11	Software Controlled APE Clock Speed Select 00000: 133MHz (PLL) 10001: 25.0MHz (CK25) 10011: 12.5MHz (CK25/2) 10101: 6.25MHz (CK25/4) 10111: 3.125MHz (CK25/8) 11001: 1.563MHz (CK25/16) 11110: PLL 133MHz/2

Clock Speed Override Policy Register (offset: 0x3624)

This register is reset by POR Reset or CPMU Register Software Reset.

Note: Clocks generated by digital frequency multiplier could be up to 3% slower than intended clock speed.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	R/O	0x0	–
MAC Clock Switch	20:16	R/W	0x0	Software Controlled MAC Core Clock Speed Select: 00000: Core = 66.67MHz (PLL/2) 10001: Core = 12.5MHz (CK25/2) 10011: Core = 6.25MHz (CK25/4) 10101: Core = 3.125MHz (CK25/8) 10111: Core = 1.563MHz (CK25/16) 11111: Core = 12.5MHz/1.25MHz (MII_CLK/2) ** Use of MII_CLK as source is discouraged. With GPHY auto-power-down feature enabled, MII_CLK will be stopped.
APE Clock Switch	12:8	R/W	0x0	Software Controlled APE Clock Speed Select 00000: 133MHz (PLL) 10001: 25.0MHz (CK25) 10011: 12.5MHz (CK25/2) 10101: 6.25MHz (CK25/4) 10111: 3.125MHz (CK25/8) 11001: 1.563MHz (CK25/16) 11110: PLL 133MHz/2
Reserved	7:0	R/O	0x0	–

Clock Override Enable Register (offset: 0x3628)

This register is reset by POR Reset or CPMU Register Software Reset. *Please note that Force Disable bit has higher priority than Override Enable.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	R/O	0x0	–
MAC Clock Speed Override Enable	13	R/W	0x0	Enable MAC clock speed override* 1: Enable 0: Disable
Reserved	12:0	R/W	0x0	–

Status Register (offset: 0x362C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:23	R/O	0x0	–
WOL ACPI Detection Enable Status	22	R/O	–	1: ACPI detection enabled 0: ACPI detection disabled
WOL Magic Packet Detection Enable Status	21	R/O	–	1: Magic Packet Detection enabled 0: Magic Packet Detection disabled
Ethernet link status	20:19	R/O	–	Ethernet Link Status 11: no link 10: 10 MB 01: 100 MB 00: 1000 MB
Link idle status	18	R/O	–	Link Idle status 1: Idle 0: Active
Reserved	17:16	R/O	0x0	–
Reserved	15:14	R/O	0x0	–
VMAIN power status	13	R/O	–	VMAIN Power Status 1: On 0: Off
IDDQ Status	12:10	R/O	0x0	IDDQ Status 000: Tactical Mode 001: Playdead Mode 010: SW IDDQ Mode 011: HW IDDQ Mode 100 : Media Sense Mode
Power State	9:8	R/O	–	Device Power State Status 11: D3 00: D0
Energy Detect Status	7	R/O	–	Energy Status 1: On 0: Off
CPMU Power State	6:4	R/O	–	Indicates the current power state of the CPMU 0x0: MISSION 0x1: LOWPWR 0x2: LINK_AWARE 0x3: LINK IDLE 0x4: PLAY DEAD

Name	Bits	Access	Default Value	Description
Power Management State Machine State	3:0	R/O	–	Indicates the current state of hardware power management state machine 0x0: MISSION_AC 0x1: LOWPWR_MODE 0x2 : LOWPWR_MODE_AC 0x3 : LINK_AWARE_MODE 0x4 : LINK_AWARE_MODE_AC 0x5 : LINK_IDLE_MODE 0x6 : LINK_IDLE_MODE_AC 0x7 : PM_MISSION 0x8 : PLAYDEAD MODE 0x9 : PLAYDEAD MODE AC

Clock Status Register (offset: 0x3630)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:30	DC	0x0	–
Flash CLOCK Disable Status	29	R/O	–	Flash clock disable status
Reserved	28:26	DC	–	–
APE HCLK Disable Status	25	R/O	–	APE FCLK clock disable status
APE FCLK Disable Status	24	R/O	–	APE HCLK clock disable status
Reserved	23:21	DC	0x0	–
MAC Clock Switch Status	20:16	R/O	–	MAC Core Clock Speed Select Status
Reserved	15:13	DC	0x0	–
APE Clock Switch Status	12:8	R/O	–	APE Clock Speed Select Status
Reserved	7	DC	0x0	–
Flash Clock Switch Status	6:4	R/O	–	Flash Clock Speed Select Status
Reserved	3:0	DC	0x0	–

PCIe Status Register (offset: 0x3634)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
pcie_dl_active	31	R/O	–	–
PCIE debug vector sel 2	30:27	R/W	–	–
PCIE debug vector 2	26:16	R/O	–	–
pcie_phylinkup	15	R/O	–	–
PCIE debug vector sel 1	14:11	R/W	–	–
PCIE debug vector 1	10:0	R/O	–	–

GPHY Control/Status Register (offset: 0x3638)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Logan SKU Setting	31:29	RO		//Logan SKU mapping based on BOND_ID[2:0] //"011" = BCM57101 CR SKU //"001" = BCM57787 Consumer SKU //"010" = BCM57764 TB Dongle SKU //"000" = BCM5762 Enterprise Client SKU //"100" = BCM5725 Server SKU //"101"~"111": Reserved These bits are derived from the device din_bond_id{2:0} New BCM57767 SKU is based on OTP user config bit [127] when setting to "1". BOND_ID needs to be set to "000" in this case.
Reserved	28:14	RO	0x0	–
GPHY 10MB Receive Only Mode TX Idle Debounce Timer	13:12	RO	0x1	10MB Receive Only Mode TX Idle Debounce Timer. 0x0 = 0 us 0x1 = 6 us 0x2 = 12 us 0x3 = 18 us
Reserved	11	RO	0x0	–
GPHY DLL IDDQ state	10	R/O	0x0	Gphy_iddq_dll_act state
GPHY pwrdn	9	R/O		CPMU controlled output to GPHY
GPHY set_bias_iddq	8	R/O		CPMU controlled output to GPHY

Name	*Bits*	*Access*	*Default Value*	*Description*
GPHY force_dll_on	7	R/O	0x1	CPMU controlled output to GPHY
GPHY dll_pwrdn_ok	6	R/O	–	CPMU controlled output to GPHY
SW controlled POR to GPHY	5	R/W	0x0	This bit is allows the boot code to reset the GPHY during an unexpected shutdown, so that it enters 100BT instead of remaining in the Gig mode to avoid unnecessary power consumption.
GPHY Reset Control	4	RW	0x0	Controls GPHY reset 0: use r0b11_ovrd which only resets the DSP registers 1: use gphy_reset which resets both MII and DSP registers
Reserved	3	RO	0x0	–
GPHY DLL Handshaking Disable	2	R/W	0x0	Legacy Address: 0x6804:[25] When this bit is set, disable GPHY DLL handshaking
BIAS IDDQ	1	R/W	0x0	Legacy Address: 0x6804:[22] When this bit is set, BIAS will be powered down
GPHY IDDQ	0	R/W	0x0	Legacy Address: 0x6804:[21] When this bit is set, GPHY will be powered down

RAM Control Register (offset: 0x363C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Core RAM Power down	31	R/W	0x0	Legacy Address: 0x6804:[24] Core RAM power-down Power down all rams except misc_bd and Txmbuf
BD RAM Power down	30	R/W	0x0	Legacy Address: 0x6804:[24] Misc_bd/Txmbuf RAM power-down
Reserved	29:22	R/O	0x0	–
Disable Secure Firmware Loading Status	21	R/O	0	This bit is generated from the OTP bit 251 which is defined as Secure Firmware Loading Enabled/Disable Bit. RXCPU firmware reads this register bit 31 to determine its code path of whether or not to enable or disable secure firmware loading function 1: Disable Secure Firmware 0: Enable Secure Firmware (Default)
Remove LAN Function	20	R/O	0	This bit is generated from the OTP bit 250. This OTP Bit 250 when program to 1 will Remove the LAN function/PCIE Function 0 and shift all the upper PCIE Functions 1-7 down to PCIE Functions 0-6. 1: Remove LAN PCIE Function 0 and shift upper PCIE Functions 1-7 down to PCIE Function 0-6. 0: Lan Function maps to PCIE Function 0

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	19:18	R/O	0x0	Reserve
Hide Function 7	17	R/O	0	This bit is derived from OTP Bit 247 1 – Hide Function 7 0 – Do Not Hide Function 7
Hide Function 6	16	R/O	0x0	This bit is derived from OTP Bit 246 1 – Hide Function 6 0 – Do Not Hide Function 6
Hide Function 5	15	R/O	0x0	This bit is derived from OTP Bit 245 1 – Hide Function 5 0 – Do Not Hide Function 5
Hide Function 4	14	R/O	0x0	This bit is derived from OTP Bit 244 1 – Hide Function 4 0 – Do Not Hide Function 4
Hide Function 3	13	R/O	0	This bit is derived from OTP Bit 243 1 – Hide Function 3 0 – Do Not Hide Function 3
Hide Function 2	12	R/O	0x0	This bit is derived from OTP Bit 242 1 – Hide Function 2 0 – Do Not Hide Function 2
Hide Function 1	11	R/O	0x0	This bit is derived from OTP Bit 241 1 – Hide Function 1 0 – Do Not Hide Function 1
Hide Function 0	10	R/O	0x0	This bit is derived from OTP Bit 240 1 – Hide Function 0 0 – Do Not Hide Function 0
Reserved	9:8	R/O	0x0	Reserve
Ram control bank 7 dis	7	R/W	0x0	Ram Control Bank 7 Disable 1: Disable 0: Enable
Ram control bank 6 dis	6	R/W	0x0	Ram Control Bank 6 Disable 1: Disable 0: Enable
Ram control bank 5 dis	5	R/W	0x0	Ram Control Bank 5 Disable 1: Disable 0: Enable
Ram control bank 4 dis	4	R/W	0x0	Ram Control Bank 4 Disable 1: Disable 0: Enable

Name	*Bits*	*Access*	*Default Value*	*Description*
Ram control bank3 dis	3	R/W	0x0	Ram Control Bank 3 Disable 1: Disable 0: Enable
Ram control bank 2 dis	2	R/W	0x0	Ram Control Bank 2 Disable 1: Disable 0: Enable
Ram control bank1 dis	1	R/W	0x0	Ram Control Bank 1 Disable 1: Disable 0: Enable
Ram control bank 0 dis	0	R/W	0x0	Ram Control Bank 0 Disable 1: Disable 0: Enable

Card Reader Idle Detection Debounce Control Register (offset: 0x3640)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	R/O	0x0	–
Card Reader Idle Detection Debounce Timer	15:0	R/W	0xA	Debounce timer setting for card reader from active to nonactive. Unit is in number of us. Default Value: 10 us.

EEE Debug Register (offset: 0x3644)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
EEE Debug Information	31:0	R/O	–	–

Core Idle Detection Debounce Control Register (offset: 0x3648)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	R/O	0x0	–
Link Idle Detection Debounce Timer	7:0	R/W	0x20	Debounce timer setting for core link from active to nonactive. Unit is in number of CPMU clock cycles. Range: up to 2^{32} CPMU clock cycles. Default Value (32 CPMU CLK) multiplied by given Core CLK scale parameter below: x1 00000: Core = 62.5 MHz (GPHY DLL/2) x8 10001: Core = 12.5 MHz (CK25/2) x16 10011: Core = 6.25 MHz (CK25/4) x32 10101: Core = 3.125 MHz (CK25/8) x64 10111: Core = 1.563 MHz (CK25/16) x64 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2)

PCIe Idle Detection Debounce Control Register (offset: 0x364C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	R/O	0x0	–
PCIE Idle Detection Debounce Timer	7:0	R/W	0x19	Debounce timer setting for PCIe link from active to nonactive. Unit is in number of CPMU clock cycles. Range: up to 2^8 CPMU clock cycles. Default: 25 CPMU clocks.

Energy Detection Debounce Timer (offset: 0x3650)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	R/O	0x0	–
Energy Detect Select	9	R/W	0x0	Legacy Address: 0x6890:[25] 1: use energy_det_apd from GPHY core 0: use output from the energy debounce logic

Name	*Bits*	*Access*	*Default Value*	*Description*
Select HW_Energy_Det	8	R/W	0x1	This bit selects the source of the System Energy_Det output. This bit is allows the boot code to reset the GPHY during an unexpected shutdown, so that it enters 100BT instead of remaining in the Gig mode to avoid unnecessary power consumption. 1: Select output of debounce logic 0: Select combination of software Force_Energy_Det and output of debounce logic to generate system Energy_Det
Select SW_HW_Oring_Energy_Det	7	R/W	0x0	This bit is allows the boot code to reset the GPHY during an unexpected shutdown, so that it enters 100 BT instead of remaining in the Gig mode to avoid unnecessary power consumption. 1: Generate system Energy_Det by Oring the SW_Force_Energy_Det with the output of the debounce logic 0: Generate system Energy_Det based on the SW_Force_Energy_Det Output
SW_Force_Energy_Det_Value	6	R/W	0x1	This bit allows the software to control the state of the System Energy_Det signal if the Select_HW_Energy_Det control bit (b12) is 0. 1: Drive System Energy_Det high 0: Drive System Energy_Det low
Disable Energy_Det Debounce Low	5	R/W	0x1	This bit is used to disable the Energy_Det_Debounce_Low logic from debouncing the GPHY Energy_Det_APD signal going low. When disabled, the Energy_Det signal will go low if the GPHY Energy_Det input signal is low for at least 640 ns. 0: Enable debounce low 1: Disable debounce low
Disable Energy_Det Debounce High	4	R/W	0x1	This bit is used to disable the Energy_Det_Debounce_High Logic from debouncing the GPHY Energy_Det_APD signal going high. When disabled, the Energy_Det signal will go high if the GPHY Energy_Det input signal is high for at least 640 ns. 0: Enable debounce high 1: Disable debounce high
Energy_Det Debounce High Limit	3:2	R/W	0x0	This parameter is used to control the debounce limit of the GPHY Energy_Det_APD signal going high. 00: 128 million CPMU clocks (5 seconds if CPMU clock frequency is 25 Mhz) 01: 256 million CPMU clocks (10 seconds if CPMU clock frequency is 25 Mhz) 10: 512 million CPMU clocks (20 seconds if CPMU clock frequency is 25 Mhz) 11: 1024 million CPMU clocks (40 seconds if CPMU clock frequency is 25 Mhz)

Name	*Bits*	*Access*	*Default Value*	*Description*
Energy_Det Debounce Low Limit	1:0	R/W	0x1	This parameter is used to control the debounce limit of the GPHY Energy_Det_APD signal going low. 00: 128 million CPMU clocks (5 seconds if CPMU clock frequency is 25 Mhz) 01: 256 million CPMU clocks (10 seconds if CPMU clock frequency is 25 Mhz) 10: 512 million CPMU clocks (20 seconds if CPMU clock frequency is 25 Mhz) 11: 1024 million CPMU clocks (40 seconds if CPMU clock frequency is 25 Mhz)

DLL Lock Timer Register (offset: 0x3654)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:12	R/O	0x0	–
GPHY DLL Lock Timer Enable	11	R/W	0x1	Use GPHY DLL lock timer instead of GPHY dll_lock signal: 1: Enable (use GPHY DLL lock timer) 0: Disable (use GPHY dll_lock signal from GPHY)
GPHY DLL Lock Timer	10:0	RO	0x3FF	GPHY DLL Lock timer value. Unit is in number of CPMU clock cycles. Range: up to 81920 CPMU clock cycles. Default: 1024 CPMU clocks (40.9 micro-seconds if CPMU clock frequency is 25 Mhz)

CHIP ID Register (offset: 0x3658)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Chip ID	31:28	R/O	0x5	–
Chip ID	27:12	R/O	0x7780	–
Base Layer Revision Information	11:8	R/O	0x0	Base layer revision history: 0000: A0 0001: B0 0010: C0
Metal Layer Revision Information	7:0	R/O	0x0	Metal layer revision history

Mutex Request Register (offset: 0x365C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:13	R/O	0x0	–
Set Request/Request Pending 12	12	R/W1S	0x0	Writing a 1 to any of these bits (2,4,8,12) pends a Mutex lock request on behalf of a s/w agent. The bit is subsequently latched by h/w and shall read 1 as long as the request is pending. Writing a 0 to a bit shall have no effect. Reading this field may return zero or more bits with value 1 – each bit with value 1 indicates a pending request.
Reserved	11:9	R/O	0x0	–
Set Request/Request Pending 8	8	R/W1S	0	–
Reserved	7:5	R/O	0x0	–
Set Request/Request Pending 4	4	R/W1S	0	–
Reserved	3	R/O	0x0	–
Set Request/Request Pending 2	2	R/W1S	0	–
Reserved	1:0	R/O	0x0	–

Mutex Grant Register (offset: 0x3660)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:13	RO	0x0	–
Request Grant 12	12	R/W1S	0x0	Reading bits 2, 4, 8, 12 shall return a maximum of one set bit at any time. The set bit shall point to the lock owner. If the Mutex is not locked, then a read shall return a value 0x0000. Writing a 1 to the already set bit shall relinquish the lock and the set bit shall be cleared. Writing a 1 to an unset bit shall cancel the corresponding pending request if there was any – and the pending bit in the Mutex_Request_Reg shall be cleared. Writing a 0 to any bits has no effect.
Reserved	11:9	RO	0x0	–
Request Grant 8	8	R/W1S	0	–
Reserved	7:5	RO	0x0	–
Request Grant 4	4	R/W1S	0	–
Reserved	3	RO	0x0	–
Request Grant 2	2	R/W1S	0	–
Reserved	1:0	RO	0x0	–

Padring Control Register – Debug Controls (offset: 0x3668)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Cpmu_power_sm_or_state	31:28	R/W	0x0	This 4-bits parameter is used to force the CPMU State Machine to the specified state when the cpmu_power_sm_override bit is set PM_MISSION_AC: 4'h0 PM_LOWPWR_MODE: 4'h1 PM_LOWPWR_MODE_AC: 4'h2 PM_LOWPWR_ED_MODE: 4'h3 PM_LOWPWR_ED_MODE_AC: 4'h4 PM_LINK_AWARE_MODE: 4'h5 PM_LINK_AWARE_MODE_AC: 4'h6 PM_AIRPLANE_MODE:4'h7 PM_AIRPLANE_MODE_AC:4'h8 PM_LINK_IDLE_MODE: 4'h9 PM_LINK_IDLE_MODE_AC: 4'hA PM_MISSION : 4hB
Cpmu_power_sm_override	27	R/W	0x0	This bit when set forces the CPMU State Machine to the state specified by the cpmu_power_sm_or_state parameter.
Reserved	26:16	R/W	0x0	–
Card reader IO hys_en	26	R/W	0	Disable card reader IO hys.
Enable card activity led	25	R/W	0	Enable card activity LED. 0: Disable. 1: Enable.
Switching regulator power off option	24	R/W	0	0: Use internal signal dout_cr_bus_pow to turn off switching regulator. 1:Use OTP bit [130] to control.
Disable CR_BUS_POW	23	R/W	0	0: Enable CR_BUS_POW feature. 1: Disable CR_BUS_POW feature.
PCIE Serdes PLL Tuning Bypass	19	R/W	0x0	Fast Pll tuning sequence, Keep prev pll_range value and skip this in next pll_tuning seq 0 = disable 1 = enable
PCIE Serdes lfck_rx select cnt0	18	R/W	0x0	Select div2(high) or div4(low) div4 of refclk for lfck_rx to fast rx_seq, Fast L1,L0s exit time. 0 = lfck_rx is using refclk/div4 1 = lfck_rx is using refclk/div2
PCIE Serdes lfck_rx select refclk	17	R/W	0x0	Select refclk lfck_rx to fast rx_seq, Fast L1,L0s exit time 0 = lfck_rx is based on bit 18 1 = lfck_rx is using refclk

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	16	R/W	0x0	–
Disable Clkreq_l in low power mode improvement	15	R/W	0x0	Clear this bit to force Clkreq_l to be deasserted in low power mode or during powering up Set this bit to disable this improvement
CQ31984 Option 2 Fix Disable	14	R/W	0x0	Disable fix for CQ31984 option 2 1 = Disable 0 = Enable
Serdes Stanalone mode	13	R/W	0x0	SerDes Standalone mode 1 = Put device into SerDes into Standalone mode 0 = Normal operation
pipe_StandAloneMode Control	12	R/W	0x0	pipe_StandAloneMode Control 1 = Assert pipe_StandAloneMode 0 = Normal operation (deassert pipe_StandAloneMode)
CQ31984 Option 4 Fix Enable	11	R/W	0x0	Enable fix for CQ31984 option 4 1 = Enable 0 = Disable
CQ31177 Fix Disable	10	R/W	0x0	Disable fix for incorrect checksum on LSO packets 1 = Disable 0 = Enable
CQ30674 Fix Enable	9	R/W	0x0	Enable fix for GRC_clkreq_enable (0x6890 bit 27) 1 = Enable 0 = Disable
Capability version for completion timeout ECN for PCIE 1.1	8	R/W	0x0	Chicken bit for CQ31116 1 = Version 1 ; Fix Disable 0 = Version 2 ; Fix Enable
CQ31984 Option 3 Fix Disable	7	R/W	0x1	Disable fix for CQ31984 option 3 1 = Disable 0 = Enable
Disable default gigabit advertisement	6	R/W	0x0	Set this bit to change GPHY gigabit advertisement register default value to disable state. This bit is used in the 10/100 SKU.
Enable gphy reset on PERST_L deassertion	5	R/W	0x1	Enable gphy reset on perst_l deassertion 1 = Enable 0 = Disable
CQ39842 fix disable	4	R/W	0x0	Disables the fix where our device does not sent out adequate number of TS2 order set during recovery 0: Enable the fix 1: Disable the fix

Name	*Bits*	*Access*	*Default Value*	*Description*
CQ39544 fix disable	3	R/W	0x0	Disables the fix whereby BIOS does not need to put the device into D3 Hot when deposit the DEADDEAD signature in Shared Memory or setting software low power bit (0x3600[2] to IDDQ the device 1: Disable the fix 0: Enable the fix
Reserved	2	R/W	0x1	–
Eclk switch using link status Disable	1	R/W	0x1	Switch emac clocks to core clock based on lnk state (CQ32677) 0: Enable 1: Disable
PERST_L pad hysteresis Enable	0	R/W	0x1	PERST_L Pad Hysteresis Enable 1: Enable 0: Disable

Link Idle Control Register (offset: 0x3670)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	R/W	0	Readable and writeable reserved bits
ISO SBD Producer Ring Empty	28	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
ISO TX MBUF Empty	27	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
SMBus FIFO empty	26	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
SMBus SM Idle	25	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode

Name	*Bits*	*Access*	*Default Value*	*Description*
PCIE Idle	24	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
ISO SBDI Idle	23	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
ISO RDMA Idle	22	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
DBU Idle	21	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
NVM Idle	20	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
SBDI Idle	19	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
RBDI Idle	18	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
MB Idle	17	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
ISO FTQ Empty	16	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode

Name	*Bits*	*Access*	*Default Value*	*Description*
WDMA Idle	15	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
RDMA Idle	14	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
MSI Idle	13	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
TXMAC FIFO empty	12	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
RXMAC FIFO empty	11	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
COL = 0	10	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
CRS = 0	9	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
TXAMAC Idle	8	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
RXER = 0	7	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode

Name	*Bits*	*Access*	*Default Value*	*Description*
RXDV = 0	6	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
MDIO Idle	5	R/W	1	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
FTQ empty	4	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
GRC Idle	3	R/W	1	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
MBUF empty	2	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
MA Idle	1	R/W	1	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode
No core reset	0	R/W	0	Link idle/Host Access condition control 1: disable this idle condition when entering link idle mode and host access mode 0: enable this idle condition when entering link idle mode and host access mode

Link Idle Status Register (offset: 0x3674)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Defaul t Value*	*Description*
Reserved	31:29	RO	0	Readable and writeable reserved bits

Name	***Bits***	***Access***	***Defaul t Value***	***Description***
ISO SBD Producer Ring Empty	28	RO		Idle Status 1: Idle 0: Busy
ISO TX MBUF Empty	27	RO		Idle Status 1: Idle 0: Busy
SMBus FIFO empty	26	RO	0	Idle Status 1: Idle 0: Busy
SMBus SM Idle	25	RO	0	Idle Status 1: Idle 0: Busy
PCIE Idle	24	RO		Idle Status 1: Idle 0: Busy
ISO SBDI Idle	23	RO		Idle Status 1: Idle 0: Busy
ISO RDMA Idle	22	RO		Idle Status 1: Idle 0: Busy
DBU Idle	21	RO		Idle Status 1: Idle 0: Busy
NVM Idle	20	RO		Idle Status 1: Idle 0: Busy
SBDI Idle	19	RO		Idle Status 1: Idle 0: Busy
RBDI Idle	18	RO		Idle Status 1: Idle 0: Busy
MB Idle	17	RO		Idle Status 1: Idle 0: Busy
ISO FTQ Empty	16	RO		Idle Status 1: Idle 0: Busy

Name	*Bits*	*Access*	*Defaul t Value*	*Description*
WDMA Idle	15	RO		Idle Status 1: Idle 0: Busy
RDMA Idle	14	RO		Idle Status 1: Idle 0: Busy
MSI Idle	13	RO		Idle Status 1: Idle 0: Busy
TXMAC FIFO empty	12	RO		Idle Status 1: Idle 0: Busy
RXMAC FIFO empty	11	RO		Idle Status 1: Idle 0: Busy
COL = 0	10	RO		Idle Status 1: Idle 0: Busy
CRS = 0	9	RO		Idle Status 1: Idle 0: Busy
TXAMAC Idle	8	RO		Idle Status 1: Idle 0: Busy
RXER = 0	7	RO		Idle Status 1: Idle 0: Busy
RXDV = 0	6	RO		Idle Status 1: Idle 0: Busy
MDIO Idle	5	RO		Idle Status 1: Idle 0: Busy
FTQ empty	4	RO		Idle Status 1: Idle 0: Busy
GRC Idle	3	RO		Idle Status 1: Idle 0: Busy

Name	Bits	Access	Defaul t Value	Description
MBUF empty	2	RO		Idle Status 1: Idle 0: Busy
MA Idle	1	RO		Idle Status 1: Idle 0: Busy
No core reset	0	RO		Idle Status 1: Idle 0: Busy

CPMU Play Dead Mode IDDQ Debounce Control Register (offset: 0x3678)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:19	RO	0x0	–
IDDQ Detection Debounce Timer for Play Dead Mode	18:0	R/W	0xA	IDDQ Detection Debounce Timer for Play Dead Mode Unit is in us. Default is 10 us.

CPMU TOP MISC. Control 1 Register (offset: 0x367C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0x0	–
PHY Reset Mask During PERST De-assertion	7	R/W	0	ECO fix added to address Cotopaxi CQ52255, link flap issue. When this bit is set, the PERST_L deassertion from 0 to 1 does not cause the GPHY's reset_n pin to toggle. When this bit is clear, the PERST_L deassertion from 0 to 1 will cause the GPHY's reset_n pin to toggle.
Reserved	5:0	RO	0x0	–

CPMU Debug Register (offset: 0x3680)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Debug bus	31:0	R/O	0x0	–

CPMU Debug Select Register (offset: 0x3684)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

CPMU LTR Control Register (offset: 0x3690)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	–
SW_User_Send_LTR2	29	RW	0	This bit is used by the firmware to control the user_send_ltr2 message. When the HW_LTR2_Mode is 0, the firmware polls the Link Status along with the Card-Reader Insertion Status to drive this signal. When there is no Ethernet Link and No Card is inserted, the firmware may drive bit 29 to 1. When a Link or Card Insertion is detected, the firmware drives this control bit to 0. The polling of the Link Status and Card Insertion Status must be in the Main Service Loop. When the HW_LTR2_Mode is 1, the hardware takes full control and the firmware does not need to poll for the status of the Link nor Card Insertion in the Main Loop.
HW_LTR2_MODE	28	RW	0	This bit is used to enable or disable Hardware LTR2 Mode. 1: Enable HW LTR Mode. 0: Disable HW LTR Mode.
User_Send_LTR2	27	RO	0	Actual user_send_ltr2 signal sending to the PCIe IP address.
Mux_user_send_ltr2	26	RO	0	Internal Muxed User_send_LTR2 signal.
Reserved	25:0	RO	0	Reserved.

SWRegulator Control 1 Register (offset: 0x36A0)

Name	*Bits*	*Access*	*Default Value*	*Description*
byptimer	0	RW	0x0	Bypass the timers in test mode
seqON_b	1	RW	0x0	Use sequential turn on for power PMOS switch
swaux_enb	2	RW	0x0	Use whole power switch (0) or a quarter of it (1)
vclampenb	3	RW	0x0	Enable clamp of vctl
gsen10	4	RW	0x0	Rsum value option

Name	*Bits*	*Access*	*Default Value*	*Description*
vref_adj	9:5	RW	0x0	11000: 0.933 11001: 1.967 11010: 1.000 11011: 1.033 11100: 1.067 11101: 1.100 11110: 1.133 11111: 1.167 00000: 1.200 BCM5725/BCM5762/BCM57767 00001: 1.233 00010: 1.267 00011: 1.300 00100: 1.333 00101: 1.367 00110: 1.400 00111: 1.433 01000: 1.467 01001: 1.500 01010: 1.533 01011: 1.567 01100: 1.600 01101: 1.633 BCM5725/BCM5762/BCM57767 01110: 1.667
cp<1:0>	11:10	RW	0x0	PMOS predriver strength
gmictl<1:0>	13:12	RW	0x0	gmC integrator current adjust
cn<1:0>	15:14	RW	0x0	NMOS Predriver Strength
burst_hys<1:0>	17:16	RW	0x0	Burst comparator hysteresis adjust
isenbias<1:0>	19:18	RW	0x0	Burst comparator hysteresis adjust
Tmode	20	RW	0x0	Set to test mode
dis_vok	21	RW	0x0	Disable vok signals
noltdly<3:0>	25:22	RW	0x0	nol gen timing adjust
ovri_adj<2:0>	28:26	RW	0x0	Overcurrent limit adjust
ovri_tadj<1:0>	30:29	RW	0x0	Overcurrent timer for adjust for fault detection
ovrien_b	31	RW	0x0	Enable overcurrent protection

SWRegulator Control 2 Register (offset: 0x36A4)

Name	*Bits*	*Access*	*Default Value*	*Description*
vcm<2:0>	2:0	RW	0x0	Isense dc offset adjust
ratio<2:0>	5:3	RW	0x0	Compensation ramp adjust
rzsel<1:0>	7:6	RW	0x0	Rzero in Loopfilter adjust
blkdly<1:0>	9:8	RW	0x0	Blanking delay adjust for Isense
soft_st<1:0>	11:10	RW	0x0	Soft start options
gmrctl	12	RW	0x0	gmC integrator resistor adjust
vclamp<1:0>	14:13	RW	0x0	vctl clamping voltage adjust
pd_snub	15	RW	0x0	Power-down snubber
enfaultb	16	RW	0x0	Enable fault detection
sel_c1	17	RW	0x0	Period adjust in burst with capacitor
tmux<4:0>	22:18	RW	0x0	Mux selection in test mode
sel_i1<1:0>	24:23	RW	0x0	Period adjust in burst with current mirrors
ptat_adj<2:0>	27:25	RW	0x0	Band-gap ptat adjust
ctat_adj<2:0>	30:28	RW	0x0	Band-gap ctat adjust
Sel_c2	31	RW	0x0	On-time adjust in burst with capacitor

SWRegulator Control 3 Register (offset: 0x36A8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Sel_i2<1:0>	1:0	RW	0x0	On-time adjust in burst with current mirrors
Sel_r1<1:0>	3:2	RW	0x0	On-time adjust in burst with resistor
Sel_r2<1:0>	5:4	RW	0x0	On-time adjust in burst with resistor
Brst_Count<1:0>	7:6	RW	0x0	Burst pulse counter for ovri
Indsel<1:0>	9:8	RW	0x0	Inductor selection (4.7uF default)
Ref_lpf_enb	10	RW	0x0	Enable RC filter for reference
DisablePOK	11	RW	0x0	Disable buck_pok signal
Fc<2:0>	14:12	RW	0x0	Internal oscillator frequency coarse adjust
Fi<3:0>	18:15	RW	0x0	Internal oscillator frequency fine adjust
sel_ext_clk (Clock 25/9 enable)	19	RW	0x0	Select external clock This bit enables ck25div9.
clkext_inv	20	RW	0x0	Invert external clock before use
rco_off	21	–	–	Power-down rco
Reserved	29:25	RW	0x0	Reserved
SR_LX override	24	RW	0x0	Force SR_LX node to be always low

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	23:22	RW	0x0	Reserved
SWR buck down control	30	RW	0x0	Controls the cpmu_swr_buck_down output
SWR burst mode control	31	RW	0x0	This bit controls the cpmu_swr_burst_mode output when 0x36AC[2] is set.

Miscellaneous Control Register (offset: 0x36AC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:27	RW	0x0	Reserved
disable_ols_cr_bus_pow	26	RW	0x0	0: default. Enable OLS card read bus power output indication on GPIO0 pin for consumer SKU. 1: disable OLS card read bus power output indication on GPIO0 pin for consumer SKU. ***Note:*** Enable_card_activity_led needs to be set to "0" and sel_gpio0_n_cr_bus_pow needs to be set to "1" in GRC to enable the right path to GPIO0 similar to legacy design. This means that in this configuration, the cr_bus_pow from the OLS (2nd Card Reader IP) is mapped to the GPIO0.
Speculative early L1 exit mode	25:24	RW	0x0	Mode control for speculative early L1 exit 2'b00: Enable speculative early L1 exit when Ethernet is linked to 1000 mb/s only 2'b01: Enable speculative early L1 exit regardless link speed 2'b10: Disabled speculative early L1 exit 2'b11: Reserved

Name	*Bits*	*Access*	*Default Value*	*Description*
Speculative early L1 exit delay	23:16	RW	0x0	Delay control for speculative early L1 exit detection. The detection circuit uses the earliest receive packet indicator after mac address filtering circuit. Each increment increases the delay by 10.24 0uS. 0x0: no delay 0x1: 10.24 us delay 0x2: 20.48 us delay .. .. 0xFF: 2621.44 us delay The optimal value of this delay is a function of link speed, packet size, L1 exit latency, L1 entrance inactive timer value and the following simulation data. 1000 mb/s 1518byte packet—advances l1aspm exiting by 13.880 us with no delay 1000 mb/s 64byte packet—advances l1aspm exiting by 2.488 us with no delay 100mb/s 1518byte packet—advances l1aspm exiting by 129.377 us with no delay 100 mb/s 64byte packet—advances l1aspm exiting by 13.217 us with no delay 10 mb/s 1518byte packet—advances l1aspm exiting by 1274.94 us with no delay 10 mb/s 64byte packet—advances l1aspm exiting by 113.98 us with no delay
Reserved	15:12	RW	0x0	Reserved
Disable Legacy Power Lost Detection	11	RW	0x0	This bit, when set will disable the old legacy way of generating reset for unexpected shutdown based on Vmain Present Going Away while device is in D3
Disable New Power Loss Detection	10	RW	0x0	This bit, when set will disable the new legacy way of generating reset for unexpected shutdown based on PCIe Link not in L23 State while Perst_L is asserting
Clock ck25/9 shut down delay timer value select	9:8	RW	0x0	This bits indicate the amount of time to delay the ck25/9 clock shutdown after 0x36AC[1] is set. 2'b00: 5us 2'b01: 10us 2'b10: 15us 2'b11: 20us

Name	*Bits*	*Access*	*Default Value*	*Description*
Burst Mode IDDQ Delay timer value select	7:6	RW	0x0	These bits indicate the amount of time to delay the assertion of low power IDDQ to generate cpmu_swr_burst_mode when 0x36AC[2] is 1'b0. 2'b00: 4us 2'b01: 8us 2'b10: 12us 2'b11: 16us
IDDQ Delay Output Select	5	RW	0x0	This bit selects the cpmu_swr_iddq_dly output. 1'b0: Original lp_iddq signal ORed with delayed version base on the timer value in 0x36AC[4:3]. 1'b1: Original lp_iddq signal
IDDQ delay timer value select	4:3	RW	0x0	These bits indicate the amount of time to delay the deassertion of low power IDDQ output to the SWRegulator by. 2'b00: 400us 2'b01: 800us 2'b10: 1200us 2'b11: 1600us
Disable New burst mode logic	2	RW	0x0	This bit disables the burst mode logic that's tied to the lp_iddq.
Reserved	1	RW	0x0	Reserved
SWR control enable	0	RW	0x0	FW sets this bit after it changes the SWR control registers or set it at the beginning of boot code execution.

EEE Mode Register (offset: 0x36B0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:22	R/O	0x0	Reserved
Bias 10bte Enable Control	21	RW	0x0	Enable Control bit for Bias 10bte
EEE Enable Control	20	RW	OTP Bit 137	Enable Control bit for EEE function
Drive Allow LPI Enable	19	RW	0x0	Enable Control bit allocated for driver to allow CPMU to go into EEE mode.
EEE Block Time	18:11	R/W	0x0	Block out time from deassertion of EEE MAC TX request to when the EEE module considers new request to enter LPI again.
Auto Wake Enable	10	R/W	0x0	Enable automatic removal of Link from LPI state when MAC wants to transmit.
RX LPI Enable	9	R/W	0x0	Enable LPI indication in RX direction.
TX LPI Enable	8	R/W	0x0	Enable LPI indication in TX direction.
User LPI Enable	7	R/W	0x0	Main LPI enable bit set by user.
Send Index Detection Enable	6	RW	0x0	This bit allows CPMU to use the send consumer and producer equal term to determine EEE mode.
RX CPU Allow LPI Enable	5	RW	0x0	Enable Control bit allocated for RX CPU to allow CPMU to go into EEE mode.
PCIE L1 Exit Detection Enable	4	RW	0x0	This bit allows CPMU to use the PCIe early L1 exit detection term to determine EEE mode.
EEE Link Idle Detection Enable	3	RW	0x0	This bit allows CPMU to use the EEE link idle detection term to determine EEE mode.
APE TX Detection Enable	2	RW	0x0	This bit allows CPMU to use the APE TX detection term to determine EEE mode.
Drive Allow LPI	1	RW	0x0	Control bit allocated for driver to allow CPMU to go into EEE mode.
RX CPU Allow LPI	0	RW	0x0	Control bit allocated for RX CPU to allow CPMU to go into EEE mode.

EEE Debounce Timer 1 Control Register (offset: 0x36B4)

Name	*Bits*	*Access*	*Default Value*	*Description*
PCIE Early L1 Exit Debounce Timer	31:16	RW	0x3F	PCIe early L1 exit debounce timer in us. Default is 63 us.
EEE Link Idle Debounce Timer	15:0	RW	0x3F	EEE link idle debounce timer in us. Default is 63 us.

EEE Debounce Timer 2 Control Register (offset: 0x36B8)

Name	*Bits*	*Access*	*Default Value*	*Description*
APE Transmit Debounce Timer	31:16	RW	0x3F	APE transmit debounce timer in us. Default is 63 us.
Send Index Equal Debounce Timer	15:0	RW	0x3F	Send producer and consumer index equal debounce timer in us. Default is 63 us.

EEE Link Idle Control Register (offset: 0x36BC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:7	RO	0x0	Reserved
ISO Mbuf Queue, TX Queue and HC Queue empty	6	RO	0x0	ISO Mbuf queue, TX queue and HC queue are empty. 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic
ISO TX Producer Ring Empty	5	RO	0x0	ISO TX producer ring empty. 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic
ISO Ring TXMBuf Empty	4	RO	0x0	ISO Ring TX Mbuf empty. 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic
MDIO Idle	3	RW	0x0	MDIO status control 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic UART is idle. No on-going MDIO access.
Debug UART Idle	2	RW	0x0	Debug UART status control 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic UART is idle.

Name	*Bits*	*Access*	*Default Value*	*Description*
APE TX Packet Buffer Empty	1	RW	0x0	APE subsystem internal packet buffer status control 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic
LAN TX Packet Buffer Empty	0	RW	0x0	LAN core internal packet buffer status control 1: disable this idle condition from the EEE idle detection logic 0: enable this idle condition in the EEE idle detection logic

EEE Link Idle Status Register (offset: 0x36C0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:7	RO	0x0	Reserved
ISO Mbuf Queue, TX Queue and HC Queue empty	6	RO	0x0	ISO Mbuf queue, TX queue and HC queue are empty
ISO TX Producer Ring Empty	5	RO	0x0	ISO TX producer ring empty
ISO Ring TXMBuf Empty	4	RO	0x0	ISO Ring TX Mbuf empty
MDIO Idle	3	RO	0x0	No on-going MDIO access.
Debug UART Idle	2	RO	0x0	Debug UART is idle.
APE TX Packet Buffer Empty	1	RO	0x0	Internal packet buffers in APE subsystem for TX are empty.
LAN TX Packet Buffer Empty	0	RO	0x0	Internal packet buffers in LAN core for TX are empty.

EEE Statistic Counter 1 Register (offset: 0x36C4)

Name	*Bits*	*Access*	*Default Value*	*Description*
EEE Mode Entering Counter	31:0	R/W	0x0	This counter counts the number of timers CPMU goes into EEE mode. The entire 32 bit register can be cleared by writing 0xFFFFFFFF.

EEE Statistic Counter 2 Register (offset: 0x36C8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Receive LPI Entering Counter	31:0	R/W	0x0	This counter counts the number of timers the receive side goes into LPI (same as EEE mode?). The entire 32 bit register can be cleared by writing 0xFFFFFFFF.

EEE Statistics Counter 3 Register (offset: 0x36CC)

Name	*Bits*	*Access*	*Default Value*	*Description*
EEE Link Idle Entering Counter	31:0	R/W	0x0	This counter counts the number of timers the debounced version of EEE link idle is asserted. The entire 32 bit register can be cleared by writing 0xFFFFFFFF.

EEE Control Register (offset: 0x36D0)

Name	*Bits*	*Access*	*Default Value*	*Description*
EEE Minimum Assert	31:16	R/W	0x0	The minimum time from start of LPI message transmission to end of LPI message transmission.
EEE Exit Time	15:0	R/W	0x190	Time from the end of LPI message transmission to when normal transmission is allowed again.

Current Measurement Control Register (offset: 0x36D4)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	R/W	0x0	Reserved.
Freeze 2s Average	4	R/W	0x0	Stop the update of the 2 second average register. The accumulation and holding registers will still update.
Freeze Count	3	R/W	0x0	Stop the accumulation process of the PDRIVE samples. This effectively disables the cur_meas block, and freezes all values of registers. This bit can be set to stop the register updates for register reads, and then restarted.
Recalibrate	2	R/W	0x0	Restart the calibration process and capture a single 500us accumulation period.

Name	*Bits*	*Access*	*Default Value*	*Description*
Current Measurement Read Select	1:0	R/W	0x0	0x0: Current Measurement Upper 32-bit Read Register = 0x0 Current Measurement Lower 32-bit Read Register = 500 us Calibration Count 0x1: Current Measurement Upper 32-bit Read Register = 1s Hold Count[63:32] Current Measurement Lower 32-bit Read Register = 1s Hold Count[31:0] 0x2: Current Measurement Upper 32-bit Read Register = 2s Hold Count[63:32] Current Measurement Lower 32-bit Read Register = 2s Hold Count[31:0] 0x3: : Current Measurement Upper 32-bit Read Register = 2s Average Count[63:32] Current Measurement Lower 32-bit Read Register = 2s Average Count [31:0

Current Measurement Upper 32-bit Read Register (offset: 0x36D8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Upper 32-bit of Current Measurement Count	31:0	R/O	0x0	Please refer to 0x36D4[1:0]

Current Measurement Lower 32-bit Read Register (offset: 0x36DC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lower 32-bit of Current Measurement Count	31:0	R/O	0x0	Please refer to 0x36D4[1:0]

Card Reader Idle Control Register (offset: 0x36E0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Secondary card reader AHB resetb	31	R/W	0x1	Active low AHB reset for secondary card reader. This bit is used by software to induce a reset to the Secondary Card Reader AHB Reset. This event is mapped to the top-level signal cclk_sec_rstahb_n in the logan.v.
Reserved	30:28	R/O	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
PCIE Reset Media Sense Hold Timer	27:24	R/W	0x0	Timer for controlling the number of seconds to hold media sense high after PCIe reset deassertion.
Card Reader Idle Detection Mask	23:20	R/W	0x0	Card Reader Idle Detection Mask 0x1: Mask Active 0x0: Unmasked [23]: Card Reader Not busy and PCIe Idle [22]: PCIe in reset [21]: Vmain not present [20]: Card Reader not present
Card Reader clk_xin select	19	R/W	0x0	Select between pll200 or refclk for clk_xin 0x0: pll200 0x1:refclk
Card Reader clk_xin 25 mhz divider 1 divisor	18:14	R/W	0x0	Clk_xin frequency if bit 12 is 1 0b00000 through 0b01111: 25 MHz (xtal) 0b10000: 25 MHz/2 0b10001: 25 MHz/4 0b10010: 25 MHz/8 .. 0b11111: 25 MHz/65536
Card Reader clk_xin force switch	13	SC	0x0	Clk_xin clock switcher force switch

Name	*Bits*	*Access*	*Default Value*	*Description*
Card Reader clk_xin frequency	12:7	R/W	0x16	clk_xin frequency if bit 19 is 0 0b000000 through 0b001111: 200 MHz (pll200) 0b010000: 200 MHz/2 0b010001: 200 MHz/4 0b010010: 200 MHz/8 .. 0b010110: 200 MHz/128 .. 0b011111: 200 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14 clk_xin frequency if bit 19 is 1 0b000000 through 0b001111: 100 MHz (refclk) 0b010000: 100 MHz/2 0b010001: 100 MHz/4 0b010010: 100 MHz/8 .. 0b011111: 100 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14
Card Reader clk_sleep force switch	6	SC	0x0	Clk_sleep clock switcher force switch
Card Reader clk_sleep frequency	5:0	R/W	0x19	clk_sleep frequency 0b000000 through 0b001111: 25 MHz (xtal) 0b010000: 25 MHz/2 0b010001: 25 MHz/4 0b010010: 25 MHz/8 .. 0b011001: 25 MHz/1024 .. 0b011111: 25 MHz/65536

Card Reader Clock Policy Register (offset: 0x36E4)

Name	*Bits*	*Access*	*Default Value*	*Description*
Stop Card Reader secondary clk_xin	31	R/W	0x1	Card Reader secondary clk_xin Stop 0x0: secondary clk_xin running 0x1: stop secondary clk_xin
Stop Card Reader secondary clk_ahb	30	R/W	0x0	Card Reader secondary clk_ahb Stop 0x0: secondary clk_ahb running 0x1: stop secondary clk_ahb
Card Reader Base Clk Frequency Select	29:28	R/W	0x0	cpmu base clock controlbits[1:0] 00: SD1.0? 50MHz 01: SD2.0? 50MHz 10: SD3.0? 200MHz 11: Reserved
EAV Core Clock Control	27:26	R/W	0x0	2'b00: Stop EAV core clock when EAV is disabled, otherwise keep the clock running 2'b01: Stop EAV core clock when EAV is disabled, otherwise keep the clock running 2'b10: override enabled, force EAV core clock to be on 2'b11: override enabled, force EAV core clock to be off
EAV RX Clock Control	25:24	R/W	0x0	2'b00: Stop EAV RX clock when EAV is disabled, otherwise keep the clock running 2'b01: Stop EAV RX clock when EAV is disabled, otherwise keep the clock running 2'b10: override enabled, force EAV RX clock to be on 2'b11: override enabled, force EAV RX clock to be off
Primary card reader AHB resetb	23	R/W	0x1	Active low AHB reset for primary card reader
Stop Card Reader primary clk_ahb	22	R/W	0x0	Card Reader primary clk_ahb Stop 0x0: primary clk_ahb running 0x1: stop primary clk_ahb
Stop Card Reader clk_xin	21	R/W	0x0	Card Reader clk_xin Stop 0x0: clk_xin running 0x1: stop clk_xin
Stop Card Reader clk_sleep	20	R/W	0x0	Card Reader clk_sleep Stop 0x0: clk_sleep running 0x1: stop clk_sleep
Card Reader clk_xin select	19	R/W	0x0	Select between pll200 or refclk for clk_xin 0x0: pll200 0x1:refclk

Name	Bits	Access	Default Value	Description
Card Reader clk_xin 25mhz divider 1 divisor	18:14	R/W	0x0	Clk_xin frequency if bit 12 is 1 0b00000 through 0b01111: 25 MHz (xtal) 0b10000: 25 MHz/2 0b10001: 25 MHz/4 0b10010: 25 MHz/8 .. 0b11111: 25 MHz/65536
Card Reader clk_xin force switch	13	SC	0x0	Clk_xin clock switcher force switch
Card Reader clk_xin frequency	12:7	R/W	0x11	clk_xin frequency if bit 19 is 0 0b000000 through 0b001111: 200 MHz (pll200) 0b010000: 200 MHz/2 0b010001: 200 MHz/4 0b010010: 200 MHz/8 .. 0b010110: 200 MHz/128 .. 0b011111: 200 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14 clk_xin frequency if bit 19 is 1 0b000000 through 0b001111: 100 MHz (refclk) 0b010000: 100 MHz/2 0b010001: 100 MHz/4 0b010010: 100 MHz/8 .. 0b011111: 100 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14
Card Reader clk_sleep force switch	6	SC	0x0	Clk_sleep clock switcher force switch

Name	*Bits*	*Access*	*Default Value*	*Description*
Card Reader clk_sleep frequency	5:0	R/W	0x19	clk_sleep frequency 0b000000 through 0b001111: 25 MHz (xtal) 0b010000: 25 MHz/2 0b010001: 25 MHz/4 0b010010: 25 MHz/8 .. 0b011001: 25 MHz/1024 .. 0b011111: 25 MHz/65536

Card Reader Clock Status Register (offset: 0x36E8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	R/O	0x0	–
Status of Card Reader clk_xin stop	25	R/O	0x0	–
Status of Card Reader clk_sleep stop	24	R/O	0x0	–
Card Reader clk_xin status from CCLK Block	23:22	R/O	–	clk_xin_200_25 clock switcher current status bit 22 = pipe_pllclk_200 (200 MHz)/ pipe_RefclkOut (100 MHz) is active bit 23 = xtali (25 MHz) is active
Card Reader clk_sleep status from CCLK block	21:20	R/O	–	clk_sleep_25_out clock switcher current status bit 20 = xtali (25 MHz) is active bit 21 = cr_clk_sleep_out (variable) is active
Card Reader clk_xin select	19	R/O	0x0	Select between pll200 or refclk for clk_xin 0x0: pll200 0x1:refclk
Card Reader clk_xin 25mhz divider 1 divisor	18:14	R/O	0x0	Clk_xin frequency if bit 12 is 1 0b00000 through 0b01111: 25 MHz (xtal) 0b10000: 25 MHz/2 0b10001: 25 MHz/4 0b10010: 25 MHz/8 .. 0b11111: 25 MHz/65536
Reserved	13	R/O	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Card Reader clk_xin frequency	12:7	R/O	0x0	clk_xin frequency if bit 19 is 0 0b000000 through 0b001111: 200 MHz (pll200) 0b010000: 200 MHz/2 0b010001: 200 MHz/4 0b010010: 200 MHz/8 .. 0b010110: 200 MHz/128 .. 0b011111: 200 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14 clk_xin frequency if bit 19 is 1 0b000000 through 0b001111: 100 MHz (refclk) 0b010000: 100 MHz/2 0b010001: 100 MHz/4 0b010010: 100 MHz/8 .. 0b011111: 100 MHz/65536 0b100000 through 0b111111: controlled by bit 18:14
reserved	6	R/O	–	–
Status of Card Reader clk_sleep_sw	5:0	R/O	–	clk_sleep frequency 0b000000 through 0b001111: 25 MHz (xtal) 0b010000: 25 MHz/2 0b010001: 25 MHz/4 0b010010: 25 MHz/8 .. 0b011001: 25 MHz/1024 .. 0b011111: 25 MHz/65536 0b100000 through 0b111111: clk_sleep_out (controlled by card reader)

PLL Control1 Register (offset: 0x36F0)

Name	*Bits*	*Access*	*Default Value*	*Description*
pll_ch0_mdiv	31:24	RW	0x19	This controls the pll channel0 mdiv; 0x19 or 25 will produce 48Mhz Output Clock for the USB Core Clock
pll_ch1_mdiv	23:16	RW	0x9	This controls the pll channel1 mdiv; 0x9 will produce 133Mhz Output Clock for the core_clock/2
pll_ch2_mdiv	15:8	RW	0x9	This controls the pll channel2 mdiv; 0x9 will produce 133Mhz Output Clock for the APE Core Clock
Reserved	7:3	RW	0	Reserved
Pll_ch2_load_en	2	RW	0	This PLL Channel2 Load Enable is tied directly to the PLL Channel 2 load_en pin during functional mode
Pll_ch1_load_en	1	RW	0	This PLL Channel1 Load Enable is tied directly to the PLL Channel 1 load_en pin during functional mode
Pll_ch0_load_en	0	RW	0	This PLL Channel0 Load Enable is tied directly to the PLL Channel 0 load_en pin during functional mode

PLL Control2 Register (offset: 0x36F4)

Name	*Bits*	*Access*	*Default Value*	*Description*
pll_ch3_mdiv	31:24	RW	0x13	This controls the pll channel0 mdiv; 0x13 or 19 will produce 63.1Mhz Output Clock for the TL Clock
pll_ch4_mdiv	23:16	RW	0x18	This controls the pll channel4 mdiv; 0x18will produce 50Mhz Output Clock for the NCSI Clock
pll_ch5_mdiv	15:8	RW	0x6	This controls the pll channel5 mdiv; 0x6 will produce 200Mhz Output Clock for the CR Clock
Reserved	7:3	RW	0	Reserved
Pll_ch5_load_en	2	RW	0	This PLL Channel5 Load Enable is tied directly to the PLL Channel 5 load_en pin during functional mode
Pll_ch4_load_en	1	RW	0	This PLL Channel4 Load Enable is tied directly to the PLL Channel 4 load_en pin during functional mode
Pll_ch3_load_en	0	RW	0	This PLL Channel3 Load Enable is tied directly to the PLL Channel 3 load_en pin during functional mode

PLL Control3 Register (offset: 0x36F8)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RW	0	Reserved

Name	*Bits*	*Access*	*Default Value*	*Description*
Pll_ndiv_int	13:4	RW	0x30	This parameter controls the pll ndiv_int input. Setting it to 0x30 or 48 will configure the PLL to run at 1200MHz internal as the VCO Frequency.
Reserved	3	RW	0	Reserved
Pll_pdiv_int	2:0	RW	0x1	This parameter controls the pll pdiv_int input.

Clock Generator Control Register (offset: 0x36FC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RW	0	Reserved
Reserved	15	RW	0x0	–
Pclk_Block TLP_HCLK Secondary Clock Switcher ClkSel Override Enable	14	RW	0	0 – Disable Secondary tlp_hclk Clock Switcher CLKSEL Override Mode 1 – Enable Secondary tlp_hclk Clock Switcher CLKSEL Override Mode from SW Bit 12
Pclk_Block TLP_HCLK Secondary Clock Switcher Force_Switch	13	RW	0x0	0 – No force switch Pulse – enable Secondary Force Switch to the i_tlp_hclk_pri_clk_switcher
Pclk_Block TLP_HCLK Secondary Clock Switcher ClkSel	12	RW	0	1 – Select ape_hclk as the clock source for the Secondary TLP Clock 0 – Select the output from the TLP Clock Switcher as an intermediate source for the Secondary TLP Clock
Reserved	11	RW	0x0	–
Pclk_Block TLP_HCLK Primary Clock Switcher ClkSel Override Enable	10	RW	0	0 – Disable Primary tlp_hclk Clock Switcher CLKSEL Override Mode 1 – Enable Primary tlp_hclk Clock Switcher CLKSEL Override Mode from SW Bit 8
Pclk_Block TLP_HCLK Primary Clock Switcher Force_Switch	9	RW	0x0	0 – No force switch Pulse – enable Primary Force Switch to the i_tlp_hclk_pri_clk_switcher
Pclk_Block TLP_HCLK Primary Clock Switcher ClkSel	8	RW	0	1 – Select ape_hclk as the clock source for the Primary TLP Clock 0 – Select the output from the TLP Clock Switcher as an intermediate source for the Primary TLP Clock
Pclk_Block TLP Clock Switcher Force_Switch override Enable	7	RW	0x0	0 – Disable tlp_clk_switcher Force Switch Override Mode 1 – Enable tlp_clk_switcher Force Switch Override Control from SW bit 5
Pclk_Block TLP Clock Switcher ClkSel Override Enable	6	RW	0	0 – Disable TLP Clock Switcher CLKSEL Override Mode 1 – Enable TLP Clock Switcher CLKSEL Override Mode from SW Bit 4

Name	*Bits*	*Access*	*Default Value*	*Description*
Pclk_Block TLP Clock Switcher Force_Switch	5	RW	0x0	0 – No force switch Pulse – enable Force Switch to the i_tlp_clk_switcher
Pclk_Block TLP Clock Switcher ClkSel	4	RW	0	1 – Select CK25 as the clock source for the TLP Clock 0 – Select the output from the TLP Backup Clock Switcher as an intermediate source for the Primary TLP Clock
Pclk_Block TLP Back-Up Clock Switcher Force_Switch	3	RW	0x0	0 – No force switch Pulse – enable Force Switch to the i_tlp_backup_clk_switcher
Pclk_Block TLP Back-Up Clock Switcher ClkSel	2	RW	0	1 – Select PCIE Serdes CLK/4 or 62.5MHz 0 – Select Device PLL_TL_CLK or 63.15Mhz)
Godchildren Back-Up Core Clock Switcher Force_Switch	1	RW	0	0 – No force switch Pulse – enable Force Switch to the i_backup_core_clk_switcher
Godchildren Back-Up Core Clock Switcher CLKSEL	0	RW	0x1	0 – Select GPHY CK125 1 – Select PLL 133MHz

Power Gating Mask Register (offset: 0x3700)

This register is used to control whether to gate power to power domains in low power IDDQ mode.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	Not Used
HW IDDQ power gating	0	RW	0	1 – Enable power gating to Logan core in HW IDDQ mode.

Power Gating Mask Register for APE (offset: 0x3704)

This register is used to control whether to gate power to various memories inside APE.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RW	0	Not Used
Gate power to Offline Storage buffer memory	16	RW	0	1: Power down Offline storage memory
Gate power to APE_ATB_MEM	15	RW	0	1: Power down APE_ATB memory
Gate power to APE_SHR_MEM	14	RW	0	1: Power down APE shared memory
Gate power to APE_ARB_MEM	13	RW	0	1: Power down APE_ARB memory

Name	*Bits*	*Access*	*Default Value*	*Description*
Gate power to NVRAM sector memory	12	RW	0	1: Power down sector memory in NVRAM controller
Gate power to APE scratch-pad memory	11:4	RW	0	1: Power down APE scratch pad memory bank 7-0

Power Gating Control Register (offset: 0x3708)

RXCPU/APE can write and read this register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	Not Used
Link aware mode MA memory control	13	RW	0	1: gate off power to MA memory in link aware mode. (TX and RX Mbuf)
Reserved	12:7	RW	0	Not used
Power gating for RDMA_DR_FIFO	6	RW	0	1: Gate off power to RDMA_DR_FIFO
Power gating for MA_TX_MBUF memory	5	RW	0	1: Gate off power to MA_TX_MBUF memory
Power gating for MA_RX_MBUF memory	4	RW	0	1: Gate off power to MA_RX_MBUF memory
Power gating for CPU_SCRATCHPAD	3	RW	0	1:Gate off power to CPU_SCRATCHPAD
Power gating for SD 4.0 card reader memory	2	RW	0	1: Gate off power to SD4.0 card reader memory
–	1	RW	0	–
Power gating for Logan core	0	RW	0	1: Gate off power to Logan core

Power Good Delay Register (offset: 0x370c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31	RO	0	Not Used
power good delay register	15:0	RW	0x1388	Power good delay after power up event. The unit is 1 us, default is 5 ms. It is shared by all the power up events. RXCPU should be aware of this power time when bringing up Logan core from power down. This value should be re-characterized after the real silicon has been tested.

Power Gating Status (offset: 0x3710)

Name	*Bits*	*Access*	*Default Value*	*Description*
Power good status	31	RO	0	1: means the power is good for the last power up event
Reserved	30:27	RO	0	–
Misc small memories in Logan core	26	RO	1	1: Power is up
SD4.0 memory	25	RO	1	1: Power is up
APE Offline Storage Memory	24	RO	1	1: Power is up
APE_ARB_MEM	23	RO	1	1: Power is up
APE_SHR_MEM	22	RO	1	1: Power is up
APE_ATB_MEM	21	RO	1	1: Power is up
NVRAM sector memory	20	RO	1	1: Power is up
APE memory	19:12	RO	0xFF	1: Power is up
Reserved	11:6	RO	1	Not used
RDMA_DR_FIFO	5	RO	1	1: Power is up
MA_TX_MBUF memory	4	RO	1	1: Power is up
MA_RX_MBUF memory	3	RO	1	1: Power is up
CPU_SCRATCHPAD	2	RO	1	1: Power is up
Logan core power status	0	RO	1	1: Power is up

Power Gating Debug Register (offset: 0x3718)

This register allows Logan core or memories to be forced on. It is created for debugging purposes.

Note: When Logan core is power gated, all the memories will be power gated regardless of this register setting.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Not Used
SD4.0 card reader buffer memory	29	RW	0	Power up SD4.0 card reader memory
Offline Storage buffer memory	28	RW	0	Power up OLS buffer memory
Reserved	27:22	RO	0	Reserved
RDMA_DR_FIFO	21	RW	0	Power up RDMA FIFO

Name	*Bits*	*Access*	*Default Value*	*Description*
MA_TX_MBUF memory	20	RW	0	Power up MA TX MBUF memory
MA_RX_MBUF memory	19	RW	0	Power up MA RX MBUF memory
CPU_SCRATCHPAD memory	18	RW	0	Power up RXCPU scratch pad memory
APE_SHR_MEM	17	RW	0	Power up APE shared memory
APE_ATB memory	16	RW	0	Power up APE_ATB memory
APE_ARB memory	15	RW	–	Power up APE_ARB memory
NVRAM sector memory	14	RW	0	Power up NVRAM sector memory
APE scratch-pad memory power control	13:6	RW	00	Power up APE memory bank 0-7
Force Logan core to be always on	1	RW	0	1: Force Logan core to be on regardless of IDDQ or Runtime D3 status
Reserved	0	RW	0	–

Memory Power Up/Down Wait Register (offset: 0x371C)

Each memory can be powered down by changing its psm_vdd[1:0] pin. When coming out of power down mode, the daughter switch must be turned on first, then the mother switch. This register defines the wakeup time.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
Power up wait for memory mother switch	13:10	RW	4	The number of additional clock cycles it takes for mother switch to wake up. Each unit is 16 cycles, default is 64 cycles.
Power up wait for memory daughter switch	9:0	RW	0xDC	The number of clock cycles it takes for daughter switch to wake up. Unit is 1 cycle. Default is 220, which is 220 cycles.

Power Management Communication Channels (offset: 0x3720)

RXCPU/APE can deposit current status into this register. It can only be reset by POR. The default is 0. When RXCPU/APE is powered up, a non-zero value indicates RXCPU/APE has deposited a status signature before it got powered down. This is open protocol and can be defined later.

Name	*Bits*	*Access*	*Default Value*	*Description*
Power island communication channel between CPMU and APE	31:16	RW	0	Open protocol for APE, so it knows its previous state after waking up. Only writable by APE. Readable by both APE and RXCPU.
Power island communication channel between CPMU and RXCPU	15:0	RW	0	Open protocol for RXCPU so it knows its previous state after waking up. Only writable by RXCPU. Readable by both APE and RXCPU.

Runtime D3 Control and Status Register (offset: 0x3724)

Note: Runtime D3 enable is controlled by PCIE configuration register 0x4C[8] of function 0 and function 1. The value of these register bits are snooped when the PCIE IP first indicates that it’s entering D3 cold state.

Name	*Bits*	*Access*	*Default Value*	*Description*
CPMU Runtime D3 Status	31	RO	0	CPMU status bit indicating whether the chip is in Runtime D3 mode

CPMU Runtime D3 Wake State	30:29	RO	0	0x0: No internal wake;
				0x1: Wake by GPHY;
				0x2: Wake by card event;
				0x3: Wake by both;

Reserved	28:13	RO	0	Reserved
PCIE User PME Wake for Card Reader Function	12	RW	0	This bit is set and cleared by FW to wake up PCIE bus from D3 cold after exiting Runtime D3 mode
PCIE User PME Wake for LAN Function	11	RW	0	This bit is set and cleared by FW to wake up PCIE bus from D3 cold after exiting Runtime D3 mode
Release PERST_L to PCIE	10	RW	0	This bit is set by FW to release PERST_L to PCIE after exiting from Runtime D3 mode. Read value of this bit is always 1’b0.
SD4.0 TLP Clock Tree Enable	9	RO	0	0: Enable SD4.0 TLP clock tree 1: Stop SD4.0 TLP clock tree
FW Runtime D3 State	8	RW	0	This bit is set and cleared by FW to indicate Runtime D3 state. FW uses this bit to distinguish whether it’s awake from low power IDDQ mode or Runtime D3 mode.

Name	*Bits*	*Access*	*Default Value*	*Description*
Runtime D3 Hard Reset Delay Control	7:5	RW	0x2	0x0: No Delay 0x1: Delay hard reset by 10 us 0x2: Delay hard reset by 20 us 0x3: Delay hard reset by 30 us 0x4: Delay hard reset by 40 us 0x5-0x7: Reserved decoded as no delay.
Runtime D3 Wake by Card Removal	4	RW	0	1: Wake up PCIE bus from D3 cold when card removal is detected. 0: Do not wake up PCIE bus when card is removed.
Runtime D3 Wake by Card Insertion	3	RW	0	1: Wake up PCIE bus from D3 cold when card insertion is detected. 0: Do not wake up PCIE bus when card is inserted.
Runtime D3 Clock and Power Gate	2	RW	0	1: Gate off both the clock and power to Logan Core in Runtime D3 mode. 0: Do not gate off clock and power to Logan core in Runtime D3 mode.
Runtime D3 Clock Stop	1	RW	0	1: Turn off clocks inside Logan core in Runtime D3 mode. 0: Leave Logan core clocks running in Runtime D3 mode.
Reserved	0	RO	0	–

Fast Boot Register (offset: 0x3728)

Name	*Bits*	*Access*	*Default Value*	*Description*
Fast Boot Enable	31	RW	0	FW controlled fast enable 1: enabled 0: disabled
RXCPU Program Counter	30:0	RW	0	RXCPU program counter

Switching Regulator Control 4 Register (offset: 0x372C)

Name	*Bits*	*Access*	*Default Value*	*Description*
LNR2 Power Down Enable	31	RW	0	LNR2 Power Down Enable
Reserved	30	RO	0	Reserved
LNR2 i_vregcntl_en	29	RW	0	LNR2 i_vregcntl_en
LNR2's i_vregcntl[12:0]	28:16	RW	0	LNR2's i_vregcntl[12:0]
Reserved	15:14	RO	0	Reserved

Name	Bits	Access	Default Value	Description
LNR1 i_vregcntl_en	13	RW	0	LNR1 i_vregcntl_en
LNR1's i_vregcntl[12:0]	12:0	RW	0	LNR1's i_vregcntl[12:0]

Runtime D3 PCIE Configuration Register 0-6 (offset: 0x373C-0x3754)

Name	Bits	Access	Default Value	Description
Runtime D3 PCIE Config Register	31:0	RW	0	These registers are used for boot code to storage PCIE configuration registers before powering down Logan core in runtime D3 mode.

Switching Regulator Status Register (offset: 0x3758)

Name	Bits	Access	Default Value	Description
Switching Regulator Status Register	31:0	RO	TBD	Switching regulator status register.

Host Coalescing Control Registers

The Host Coalescing Control Registers are responsible for pacing the rate at which the NIC updates the host's transmit and receive buffer descriptor ring indices. Although the host produces and receives frames in one or more buffer descriptors, the Host Coalescing state machine always updates the host on frame boundaries. Additionally, the Host Coalescing state machine regulates the rate at which the statistics are updated in host memory.

All registers reset are core reset unless specified.

Host Coalescing Mode Register (offset: 0x3C00)

Name	*Bits*	*Access*	*Default Value*	*Description*
During Interrupt Frame Counter Fix Disable	31	RW	0x0	This bit disables the fix made to the frame count tracking logic for during ISR. 0x0: Status block is only updated when the frame counter exceeds the HC parameter while interrupt is being serviced. 0x1: Fix disabled. Status block is updated when the frame counter exceeds the HC parameter regardless the status of ISR.
End of RX Stream Detector Fires ALL MSI-X Vectors	30	RW	0x0	Write 1 to fire ALL MSI-X Vectors when an End of RX Stream is detected. Write 0 to fire only MSI-X Vector#0 when an End of RX Stream is detected.
Enable End of RX Stream Interrupt	29	RW	0x0	Write 1 to enable the End of RX Stream Interrupt
Enable ATTN Interrupt Fix	28	RW	0x0	Write 1 to enable the logic that fixes extra interrupt generated by assertion of ATTN bit from various block (BCM5725/BCM5762/BCM57767 CQ43520).
Reserved	27:18	RO	0	–
Coalesce Now[5:1]	17:13	RW	0	For MSI-X vectors 1-5. If set, Host Coalescing updates the Status Block immediately and sends an interrupt to host. This is a self-clearing bit. (For debug purpose only.)
No Interrupt on Force update	12	RW	–	When set, writing the Coalesce Now bit will cause a status without a corresponding interrupt event.
No Interrupt on DMAD Force	11	RW	–	When set, the COAL_NOW bit of the buffer descriptor may be set to force a status block update without a corresponding interrupt
Reserved	10	RO	0	–
Clear Ticks Mode on RX	9	RW	–	When set, the RX Host Coalescing Tick counter initializes to the idle state and begins counting only after a receive BD event is detected.

Name	*Bits*	*Access*	*Default Value*	*Description*
Status Block Size	8:7	RW	–	Status Block Size for partial status block updates 00: Full status block 01: 64 byte 10: 32 byte 11: Undefined
MSI Bits	6:4	RW	1	The least significant MSI 16-bit word is overwritten by these bits. Defaults to 0.
Coalesce Now	3	RW	0	If set, Host Coalescing updates the Status Block immediately and sends an interrupt to host. This is a self-clearing bit. (For debug purpose only.)
Attn Enable	2	RW	–	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	–	This bit control whether the Host Coalescing state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	–	When this bit is set to 1, the Host Coalescing state machine is reset. This is a self-clearing bit.

Host Coalescing Status Register (offset: 0x3C04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	RO	0	–
Error	2	RO	–	Host Coalescing Error Status
Reserved	1:0	RO	0	–

Receive Coalescing Ticks Register (offset: 0x3C08)

The value in this register can be used to control how often the status block is updated (and how often interrupts are generated) due to receiving packets. The value in this register controls how many ticks, in units of 1 µs each, get loaded in an internal receive tick timer register. The 1 uS tick is controlled by the GRC Timer Prescaler parameter from the GRC Block. The timer will be reset to the value of this register and will start counting down after every status block update (regardless of the reason for the status block update). The timer is only reset after status block updates, and is not reset after any given packet is received. When the timer reaches 0, it will be considered to be in the expired state. Once the counter is in the expired state, a status block update will occur if a packet had been received and copied to host memory (via DMA) since the last status block update.

This register must be initialized by host software. A value of 0 in this register disables the receive tick coalescing logic. In this case, status block updates will occur for receive event only if the Receive Max Coalesced BD value is reached. Of course, status block updates for other reasons (e.g., transmit events) will also include any updates to the receive indices. By setting the value in this register to a high number, a software device driver can reduce the number of status block updates and interrupts that occur due to receiving packets. This will generally increase performance in hosts that are under a high degree of stress and whose RISCs are saturated due to handling a large number of interrupts from the network controller. For host environments where receive interrupt latency needs to be very low, and the host is not close to be saturated, it is recommended that this register be set to 1.

Note: EAV-Mode or RSS-Mode is enabled:

Receive Coalescing Ticks Register for VRQ 0 => 0x3C08
Receive Coalescing Ticks Register for VRQ 1 => 0x3D80
Receive Coalescing Ticks Register for VRQ 2 => 0x3D98
Receive Coalescing Ticks Register for VRQ 3 => 0x3DB0
Receive Coalescing Ticks Register for VRQ 4 => 0x3DC8

Send Coalescing Ticks Register (offset: 0x3C0C)

The value in this register can be used to control how often the status block is updated (and how often interrupts are generated) according to the completion of transmit events. The value in this register controls how many ticks, in units of 1 µs each, get loaded in an internal transmit tick timer register. The 1 uS tick is controlled by the GRC Timer Prescaler parameter from the GRC Block. The timer will be reset to the value of this register and will start counting down, after every status block update (regardless of the reason for the status block update). The timer is only reset after status block updates, and is not reset after a transmit event completes. When the timer reaches 0, it will be considered to be in the expired state. Once the counter is in the expired state, a status block update will occur if a transmit event has occurred since the last status block update. In this case, a transmit event is defined by an update to one of the device's Send BD Consumer Indices. It should be noted that a Send Consumer Index increments whenever the data associated with a particular packet has been successfully moved (via DMA) across the bus, rather than when the packet is actually transmitted over the Ethernet wire.

This register must be initialized by host software. A value of 0 in this register disables the transmit tick coalescing logic. In this case, status block updates will occur for transmit events only if the Send Max Coalesced BD value is reached, or if the BD_FLAG_COAL_NOW bit is set in a send BD. Status block updates for other reasons (e.g., receive events) will also include any updates to the send indices. By setting the value in this register to a high number, a software device driver can reduce the number of status block updates, and interrupts, that occur due to transmit completions. This will generally increase performance in hosts that do not require their send buffers to be freed quickly. For host environments that do require their send buffers to be recovered quickly, it is recommended that this register be set to 0.

Note: EAV-Mode is enabled:

Send Coalescing Ticks Register for TXQ 0 => 0x3C0C
Send Coalescing Ticks Register for TXQ 1 => 0x3D84

Receive Max Coalesced BD Count Register (offset: 0x3C10)

This register contains the maximum number of receive return ring BDs that must filled in by the device before the device will update the status block due to a receive event. Whenever the device completes the reception of a packet, it will fill in a receive return ring BD, and then increment an internal receive coalesce BD counter. When this internal counter reaches the value in this register, a status block update will occur. This counter will be reset (i.e., zeroed) whenever a status block update occurs regardless of the reason for the status block update. This register must be initialized by host software. A value of 0 in this register disables the receive max BD coalescing logic. In this case, status block updates will occur for receive packets only via the Receive Coalescing Ticks mechanism. Status block updates for other reasons (e.g., transmit events) will also include any updates to the receive indices. For simplicity, if a host wanted to get a status block update for every received packet, the host driver should just set this register to a value of 1. On the other hand, by setting the value in this register to a high number, a software device driver can reduce the number of status block updates and interrupts that occur due to receiving packets. This can increase performance in hosts that are under a high degree of stress and whose RISCs are saturated due to handling a large number of interrupts from the network controller. However, in lower traffic environments, there is no guarantee that consecutive packets will be received in a timely manner. Therefore, for those environments, it is recommended that the Receive Coalescing Ticks register are used to make sure that status block updates due to receiving packets are not delayed for an infinite amount of time.

Note: EAV-Mode or RSS-Mode is enabled:

Receive Max Coalesced BD Count Register for VRQ 0 => 0x3C10
Receive Max Coalesced BD Count Register for VRQ 1 => 0x3D88
Receive Max Coalesced BD Count Register for VRQ 2 => 0x3DA0
Receive Max Coalesced BD Count Register for VRQ 3 => 0x3DB8
Receive Max Coalesced BD Count Register for VRQ 4 => 0x3DD0

Send Max Coalesced BD Count Register (offset: 0x3C14)

This register contains the maximum number of send BDs that must be processed by the device before the device will update the status block due to the transmission of packets. Whenever the device completes the DMA of transmit packet buffer, it increments an internal send coalesce BD counter. When this internal counter reaches the value in this register, a status block update will occur. This counter will be reset (i.e. zeroed) whenever a status block update occurs regardless of the reason for the status block update. This register must be initialized by host software. A value of 0 in this register disables the send max BD coalescing logic. In this case, status block updates will occur for receive packets only via the Send Coalescing Ticks mechanism. Of course,

status block updates for other reasons (e.g., receive events) will also include any updates to the send indices. For simplicity, if a host wanted to get a status block update for every transmitted packet, the host driver could just set this register to a value of 1. On the other hand, by setting the value in this register to a high number, a software device driver can reduce the number of status block updates and interrupts that occur due to transmitting packets. This can increase performance in hosts that are under a high degree of stress and whose RISCs are saturated due to handling a large number of interrupts from the network controller. However, in lower traffic environments, there is no guarantee that consecutive packets will be transmitted in a timely manner. Therefore, for those environments, it is recommended that the Send Coalescing Ticks register are used to make sure that status block updates due to transmitting packets are not delayed for an infinite amount of time.

Note: When EAV-Mode is enabled:

Send Max Coalesced BD Count Register for TXQ 0 => 0x3C14
Send Max Coalesced BD Count Register for TXQ 1 => 0x3D8C

Receive Max Coalesced BD Count During Interrupt (offset 0x3C18)

This register is similar to "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 446, but it is used instead when the host is considered to be in its interrupt service routine (ISR). In this case, the NIC considers the host to be in its ISR whenever "Interrupt Mailbox 0 Register (Offset 0x200-0x207)" for host standard and flat modes is set to a nonzero value or the Mask Interrupt bit is set.

Note: When EAV-Mode or RSS-Mode is enabled:

Receive Max Coalesced BD Count During Interrupt Register for VRQ 0 => 0x3C18
Receive Max Coalesced BD Count During Interrupt Register for VRQ 1 => 0x3D90
Receive Max Coalesced BD Count During Interrupt Register for VRQ 2 => 0x3DA8
Receive Max Coalesced BD Count During Interrupt Register for VRQ 3 => 0x3DC0
Receive Max Coalesced BD Count During Interrupt Register for VRQ 4 => 0x3DD8

Send Max Coalesced BD Count During Interrupt (offset 0x3C1C)

This register is similar to "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 446, but this register is used instead when the host is considered to be in its ISR. In this case, the NIC considers the host to be in its ISR whenever "Interrupt Mailbox 0 Register (Offset 0x200-0x207)" for host standard and flat modes is set to a nonzero value or when the Mask Interrupt bit is set.

Note: When EAV-Mode is enabled:

Send Max Coalesced BD Count During Interrupt Register for TXQ 0 => 0x3C1C
Send Max Coalesced BD Count During Interrupt Register for TXQ 1 => 0x3D94

Status Block Host Address Register (offset: 0x3C38)

This 64-bit register is in host address format and tells the NIC where to DMA the status block.

Status Block Base Address Register (offset: 0x3C44)

This 32-bit register is the location of the status block structure in NIC memory.

Flow Attention Register (offset: 0x3C48)

The Flow attention register reports attentions from the various transmit and receive state machines, flow-through queues and the MBUF allocator. Whenever one of these blocks detects an attention situation, it sets the appropriate bit in the Flow attention register. Refer to the state machine causing the attention to determine the exact cause. The attention bits are cleared by writing a one to the bit (W2C). If a bit is marked as fatal, it means that the associated state machine is halted, and that corrective action must be taken by a CPU.

Name	*Bits*	*Access*	*Default Value*	*Description*
Send BD Initiator	31	W2C	–	The Send BD Initiator state machine has caused an attention.
Send BD Completion	30	W2C	–	The Send BD Completion state machine has caused an attention.
Send BD Ring Selector	29	W2C	–	The Send BD Ring Selector state machine has caused an attention.
Send Data Initiator	28	W2C	–	The Send Data Initiator state machine has caused an attention.
Send Data Completion	27	W2C	–	The Send Data Completion state machine has caused an attention.
Reserved	26:24	RO	0	–
Recv BD Initiator	23	W2C	–	The Recv BD Initiator state machine has caused an attention.
Recv BD Completion	22	W2C	–	The Recv BD Completion state machine has caused an attention.
Recv List Placement	21	W2C	–	The Recv List Placement state machine has caused an attention.
Recv List Selector	20	W2C	–	The Recv List Selector state machine has caused an attention.
Recv Data and Recv BD Initiator	19	W2C	–	The Recv Data and Recv BD Initiator state machine has caused an attention.
Recv Data Completion	18	W2C	–	The Recv Data Completion state machine has caused an attention.
RCB Incorrectly Configured	17	W2C	–	Set if one of the RCBs is incorrectly configured based on the whole configuration.
DMA Completion Discard	16	W2C	–	The DMA Completion Discard state machine has caused an attention.
Host Coalescing	15	W2C	–	The Host Coalescing state machine has caused an attention.
Reserved	14:8	RO	0	–
Memory Arbiter	7	W2C	–	The Memory Arbiter has caused an attention.
MBUF Low Water	6	W2C	–	The MBUF allocation state machine has reached the mbuf low water threshold.
Reserved	5:0	RO	0	–

NIC Receive BD Consumer Index Register (offset: 0x3C50-0x3C58)

These three registers are shared by the Receive BD Completion and the Receive Data and Receive BD Initiator state machines. They are used to keep track of the receive BDs that have been DMAed to the NIC.

NIC Diagnostic Return Ring 0 Producer Index Register (offset: 0x3C80)

This register contains NIC Diagnostic Return Ring 0 Producer Index.

Name	Bits	Access	Default Value	Description
Reserved	31:9	RO	0	–
NIC Return Ring Producer Index	8:0	RW	–	NIC Return Ring 0 Producer Index

NIC Diagnostic Return Ring 1 Producer Index Register (offset: 0x3C84)

This register contains the Receive Return Ring 1 Producer Index.

Name	Bits	Access	Default Value	Description
Reserved	31:9	RO	0	–
NIC Return Ring Producer Index	8:0	RW	–	NIC Return Ring 1 Producer Index

NIC Diagnostic Return Ring 2 Producer Index Register (offset: 0x3C88)

This register contains the Receive Return Ring 2 Producer Index.

Name	Bits	Access	Default Value	Description
Reserved	31:9	RO	0	–
NIC Return Ring Producer Index	8:0	RW	–	NIC Return Ring 2 Producer Index

NIC Diagnostic Return Ring 3 Producer Index Register (offset: 0x3C8C)

This register contains the Receive Return Ring 3 Producer Index.

Name	Bits	Access	Default Value	Description
Reserved	31:9	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
NIC Return Ring Producer Index	8:0	RW	–	NIC Return Ring 3 Producer Index

NIC Diagnostic Send BD Consumer Index Register (offset: 0x3CC0)

The register keeps track of the NIC local copy of the send BD ring consumer (not the host copy which is DMAed by the Host Coalescing engine to the host). It is shared between the Send BD Initiator and the Host Coalescing state machines.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
NIC Send BD Consumer Index	8:0	RW	–	NIC Send BD Consumer Index

Memory Arbiter Control Registers

All registers reset are core reset unless specified.

Memory Arbiter Mode Register (offset: 0x4000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	–
CPU pipeline Request Disable	29	RW	0	When set to 1, the write/read requests from the internal CPU will be processed sequentially
Low Latency Enable	28	RW	0	When set to 1, the read from the CPU to the RXMBUF will take the original MA protocol, where data_rd_valid always goes after cmd_ack. If set to 0, the data_rd_valid overlaps at the same clock cycle as the cmd_ack.
Fast Path Read Disable	27	RW	0	Fast Path Read Disable. When set to 1, the read from the CPU to the RXMBUF will take the slow path that goes through the original memory arbitration logic
Reserved	26:21	RO	0	–
DMAW2 Addr Trap	20	RW	0	DMA Write 2 Memory Arbiter request trap enable
Reserved	19:17	RO	0	–
SDI Addr Trap Enable	16	RW	0	Send Data Initiator Memory Arbiter request trap enable
Reserved	15:13	RO	0	–
RDI2 Addr Trap Enable	12	RW	0	Receive Data Initiator 2 Memory Arbiter request trap enable
RDI1 Addr Trap Enable	11	RW	0	Receive Data Initiator 1 Memory Arbiter request trap enable
RQ Addr Trap Enable	10	RW	0	Receive List Placement Memory Arbiter request trap enable
Reserved	9	RO	0	–
PCI Addr Trap Enable	8	RW	0	PCI Memory Arbiter request trap enable
Reserved	7	RO	0	–
RX RISC Addr Trap Enable	6	RW	0	RX RISC Memory Arbiter request trap enable
DMAR1 Addr Trap Enable	5	RW	0	DMA Read 1 Memory Arbiter request trap enable
DMAW1 Addr Trap Enable	4	RW	0	DMA Write 1 Memory Arbiter request trap enable
RX-MAC Addr Trap Enable	3	RW	0	Receive MAC Memory Arbiter request trap enable
TX-MAC Addr Trap Enable	2	RW	0	Transmit MAC Memory Arbiter request trap enable

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable	1	RW	0	This bit controls whether the Memory Arbiter is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read.
Reset	0	RW	0	When this bit is set to 1, the Memory Arbiter state machine is reset. This is a self-clearing bit.

Memory Arbiter Status Register (offset: 0x4004)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
DMAW 2 Addr Trap	20	W2C	0	DMA Write 2 Memory Arbiter request trap
Reserved	19:17	RO	0	–
SDI Addr Trap	16	W2C	0	Send Data Initiator Memory Arbiter request trap
Reserved	15:13	RO	0	–
RDI2 Addr Trap	12	W2C	0	Receive Data Initiator 2 Memory Arbiter request trap
RDI1 Addr Trap	11	W2C	0	Receive Data Initiator 1 Memory Arbiter request trap
RQ Addr Trap	10	W2C	0	Receive List Placement Memory Arbiter request trap
Reserved	9	RO	0	–
PCI Addr Trap	8	W2C	0	PCI Memory Arbiter request trap
Reserved	7	RO	0	–
RX RISC Addr Trap	6	W2C	0	RX RISC Memory Arbiter request trap
DMAR1 Addr Trap	5	W2C	0	DMA Read 1 Memory Arbiter request trap
DMAW1 Addr Trap	4	W2C	0	DMA Write 1 Memory Arbiter request trap
RX-MAC Addr Trap	3	W2C	0	Receive MAC Memory Arbiter request trap
TX-MAC Addr Trap	2	W2C	0	Transmit MAC Memory Arbiter request trap
Reserved	1:0	RO	0	–

Memory Arbiter Trap Address Low Register (offset: 0x4008)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
MA Trap Addr Low	20:0	RW	–	Memory Arbiter Trap Address Low

Memory Arbiter Trap Address High Register (offset: 0x400C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
MA Trap Addr High	20:0	RW	–	Memory Arbiter Trap Address High

Buffer Manager Registers

All registers reset are core reset unless specified.

Buffer Manager Mode Register (offset: 0x4400)

Name	*Bits*	*Access*	*Default Value*	*Description*
TXFIFO Underrun Prevention Enable	31	RW	0x1	1: Enable the EMAC TXFIFO underrun prevention during LSO offload operation. It will change the arbitration algorithm of TXMBUF read requests to round-robin among CPU, PCIe, RDMA and TXMAC. When TXFIFO is almost empty, RDMA will hold its request till TXFIFO is not almost empty. 0: Disable the EMAC TXFIFO underrun prevention during LSO offload operation. The arbitration algorithm of TXMBUF read requests will be priority-based among CPU, PCIe, RDMA and TXMAC. RDMA will ignore TXFIFO almost empty alert.
Reserved	30:6	RO	0	–
Reset RXMBUF Pointer	5	R/WC	0	When this bit is set, it will cause the RXMBUF allocation and deallocation pointer to reset back to the RXMBUF base. It will also cause the RXMAC to drop the preallocated MBUF and request a new one.
MBUF Low Attn Enable	4	RW	0	MBUF Low Attn Enable MBUF low attention enable.
BM Test Mode	3	RW	0	Buffer Manager Test Mode. Must be set to 0 for normal operation.
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	1	This bit controls whether the Buffer Manager is active or not. When set to 0 it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Buffer Manager state machine is reset. This is a self-clearing bit.

Buffer Manager Status Register (offset: 0x4404)

Name	*Bits*	*Access*	*Default Value*	*Description*
BM Test Mode	31:5	RO	-	–
MBUF Low Attention	4	RO	-	MBUF Low Attention Status
Reserved	3	RO	0	–
Error	2	RO	-	Buffer Manager Error Status
Reserved	1:0	RO	0	–

MBUF Pool Base Address Register (offset: 0x4408)

The MBUF Pool Base Address specifies the beginning of the MBUF.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:23	RO	0	–
MBUF Base Address	22:0	RW	10000h	Specifies beginning of the MBUF for receive packet. The base address will ignore the lower seven bits, thus aligning the beginning of the MBUF pool on a 128-byte boundary.

MBUF Pool Length Register (offset: 0x440C)

The register specifies the length of MBUF.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:23	RO	0	–
MBUF Length	22:0	RW	4000h	Specifies the length of MBUF assigned for receive packet. The default is 16 KB. The lower seven bits should be ignored to align the MBUF pool on a 128-byte boundary.

Read DMA MBUF Low Watermark Register (offset: 0x4410)

This 6-bit register indicates the number of free MBUFs that must be available for the Read DMA Engine to dequeue a descriptor from the normal priority FTQ. If the free MBUF count drops below this mark, it must go above the high watermark to resume normal operation.

DMA MBUF Low Watermark Register (offset: 0x4414)

This 9-bit register indicates the number of free MBUFs that must be available for the RX MAC to accept a frame. If the free MBUF count drops below this mark, it must go above the high watermark to resume normal operation.

MBUF High Watermark Register (offset: 0x4418)

This 9-bit register indicates the number of free MBUFs that must be available before normal operation is restored to the Read DMA Engine and/or the RX MAC.

RX RISC MBUF Cluster Allocation Request Register (offset: 0x441C)

The RX RISC MBUF Cluster Allocation Request register contains two fields:

- A requested size field which can be up to 64 KB long
- An allocation bit

The allocation bit is used to control the access to the response register. Use this register to set the size and allocation bit and then poll the register until the allocation bit is cleared. When the allocation bit is cleared, it is safe to read from the RX RISC MBUF Cluster Allocation Response register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Allocation Bit	31	RW	0	Set this bit to 1 to request for the MBUF. When this bit is read as 0, then read the MBUF location Response register for the TXMBUF pointer.
Reserved	30:0	RO	0	–

RX RISC MBUF Allocation Response Register (offset: 0x4420)

This register returns the MBUF cluster pointer of the specified size when the Allocation bit is cleared. If a second MBUF cluster allocation request is made before this register is read, an MBUF memory leak may occur.

This register is hardwired to 61, or 0x0000003D. The TXMBUF that is dedicated for ASF is the uppermost 384 bytes. The CPU should use 0x00009E80 as the starting address for ASF.

BM Hardware Diagnostic 1 Register (offset: 0x444C)

This 32-bit register provides debug information on the TXMBUF pointer.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Last TXMBUF Deallocation Head Pointer	25:20	RO	0	Captures the last deallocation head pointer of the TXMBUF
Reserved	19:16	RO	0	–
Last TXMBUF Deallocation Tail Pointer	15:10	RO	0	Captures the last deallocation tail pointer of the TXMBUF
Reserved	9:6	RO	0	–
Next TXMBUF Allocation Pointer	5:0	RO	0	The value of the next TXMBUF allocation pointer (should be between 0 and 60)

BM Hardware Diagnostic 2 Register (offset: 0x4450)

This 32-bit register provides debug information on the TXMBUF and RXMBUF counts.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:25	RO	0	–
RXMBUF Count	24:16	RO	0	The number of RXMBUFs that were allocated
Reserved	15	RO	0	–
TXMBUF Count	14:9	RO	0	The number of TXMBUFs that were allocated
RXMBUF Left	8:0	RO	0	The number of free RXMBUFs

BM Hardware Diagnostic 3 Register (offset: 0x4454)

This 32-bit register provides debug information on the RXMBUF pointer.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:25	RO	0	–
Next RXMBUF Deallocation pointer	24:16	RO	0	The next RXMBUF that is to be deallocated
Reserved	15:9	RO	0	–
Next RXMBUF Allocation pointer	14:9	RO	0	The next RXMBUF that is to be allocated

Receive Flow Threshold Register (offset: 0x4458)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
MBUF Threshold	8:0	RW	0	Defines the integer number of MBUFs remaining before the receive MAC will drop received frames.

Real-Time Buffer Manager Registers

All registers reset are core reset unless specified.

RT Buffer Manager Mode Register (offset: 0x4600)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
MBUF Low Attn Enable	4	RW	0	MBUF Low Attn Enable MBUF low attention enable.
BM Test Mode	3	RW	0	Buffer Manager Test Mode. Must be set to 0 for normal operation.
Attention Enable	2	RW	0	When this bit is set to 1, an internal attention is generated when an error occurs.
Enable	1	RW	1	This bit controls whether the Buffer Manager is active or not. When set to 0 it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reserved	0	RO	0	–

RT Buffer Manager Status Register (offset: 0x4604)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

RT MBUF Pool Base Address Register (offset: 0x4608)

The MBUF Pool Base Address specifies the beginning of the RX MBUF.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

RT MBUF Pool Length Register (offset: 0x460C)

The register specifies the length of RX MBUF.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

RT Read DMA MBUF Low Watermark Register (offset: 0x4610)

This 6-bit register indicates the number of free MBUFs that must be available for the Read DMA Engine to dequeue a descriptor from the normal priority FTQ. If the free MBUF count drops below this mark, it must go above the high watermark to resume normal operation.

RT DMA MBUF Low Watermark Register (offset: 0x4614)

This 9-bit register indicates the number of free MBUFs that must be available for the RX MAC to accept a frame. If the free MBUF count drops below this mark, it must go above the high watermark to resume normal operation.

RT MBUF High Watermark Register (offset: 0x4618)

This 9-bit register indicates the number of free MBUFs that must be available before normal operation is restored to the Read DMA Engine and/or the RX MAC.

RT RX RISC MBUF Cluster Allocation Request Register (offset: 0x461C)

The RX RISC MBUF Cluster Allocation Request register contains two fields:

- A requested size field which can be up to 64 KB long
- An allocation bit

The allocation bit is used to control the access to the response register. Use this register to set the size and allocation bit and then poll the register until the allocation bit is cleared. When the allocation bit is cleared, it is safe to read from the RX RISC MBUF Cluster Allocation Response register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Allocation Bit	31	RW	0	Set this bit to 1 to request for the MBUF. When this bit is read as 0, then read the MBUF location Response register for the TXMBUF pointer.
Reserved	30:0	RO	0	–

RT RX RISC MBUF Allocation Response Register (offset: 0x4620)

This register returns the MBUF cluster pointer of the specified size when the Allocation bit is cleared. If a second MBUF cluster allocation request is made before this register is read, an MBUF memory leak may occur.

This register is hardwired to 61, or 0x0000003D. The TXMBUF that is dedicated for ASF is the uppermost 384 bytes. The CPU should use 0x00009E80 as the starting address for ASF.

RT BM Hardware Diagnostic 1 Register (offset: 0x464C)

This 32-bit register provides debug information on the TXMBUF pointer.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	–
Last TXMBUF Deallocation Head Pointer	25:20	RO	0	Captures the last deallocation head pointer of the TXMBUF
Reserved	19:16	RO	0	–
Last TXMBUF Deallocation Tail Pointer	15:10	RO	0	Captures the last deallocation tail pointer of the TXMBUF
Reserved	9:6	RO	0	–
Next TXMBUF Allocation Pointer	5:0	RO	0	The value of the next TXMBUF allocation pointer (should be between 0 and 60)

RT BM Hardware Diagnostic 2 Register (offset: 0x4650)

This 32-bit register provides debug information on the TXMBUF and RXMBUF counts.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:15	RO	0	–
TXMBUF Count	14:9	RO	0	The number of TXMBUFs that were allocated
Reserved	8:0	RO	0	–

RT BM Hardware Diagnostic 3 Register (offset: 0x4654)

This 32-bit register provides debug information on the RXMBUF pointer.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	0	–

RT Receive Flow Threshold Register (offset: 0x4658)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	–
MBUF Threshold	8:0	RW	0	Defines the integer number of MBUFs remaining before the receive MAC will drop received frames.

BD RDMA Registers

All registers reset are core reset unless specified.

BD Read DMA Miscellaneous Register (offset: 0x4700)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	–
Address Overflow Error Logging Enable	25	RW	0	This bit when set, enables the address overflow error to be generated when the DMA Read Engine performs a DMA operation that crosses a 4G boundary. This error is reported in bit 3 of the DMA Read Status Register. Subsequently, this will generate an internal event to interrupt the internal CPU and the DMA Read Engine will lock up after detecting this error. So it's recommended that this bit should not be set by firmware or software. 1: Enable Address Overflow Error Logging 0: Disable Address Overflow Error Logging.
Reserved	24:18	RO	0	–
PCI Request Burst Length	17:16	RW	11	The two bits define the burst length that the RDMA read engine would request to the PCI block. • 00 = 128B • 01 = 256B • 10 = 512B • 11 = 4KB
Reserved	15:14	RO	0	–
Attention Enables	13:2	RO	0	Enable read DMA PCI-X split transaction timeout expired attention.
Enable	1	RO	0	This bit controls whether the Read DMA state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read. This bit is derived from Register 4800[1]
Reserved	0	RO	0	Reserved

BD Read DMA Miscellaneous Register (offset: 0x4704)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
Malformed TLP or Poison TLP Error Detected	10	W2C	0	Malformed TLP or Poisoned TLP Occurs from BD DMA Read Engine Read Request

Name	Bits	Access	Default Value	Description
Read DMA Local Memory Write Longer Than DMA Length Error	9	W2C	0	Read DMA Local Memory Write Longer Than DMA Length Error.
Read DMA PCI FIFO Overread Error	8	W2C	0	Read DMA PCI FIFO Overread Error (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Error	7	W2C	0	Read DMA PCI FIFO Underrun Error
Read DMA PCI FIFO Overrun Error	6	W2C	0	Read DMA PCI FIFO Overrun Error
Read DMA PCI Host Address Overflow Error	5	W2C	0	Read DMA PCI Host Address Overflow Error. A host address overflow occurs when a single DMA read begins at an address below a multiple of 4 GB and ends at an address above the same multiple of 4 GB (i.e., the host memory address transitions from 0xXXXXXXXX_FFFFFFFF to 0xYYYYYYYY_00000000 in a single read). This is a fatal error.
DMA Read Completion Timeout	4	W2C	0	Completion Timeout Occurs from BD DMA Read Engine Read Request
Completion Abort Error	3	W2C	0	Completion Abort Occurs from BD DMA Read Engine Read Request
Unsupported Request Error Detected	2	W2C	0	Unsupported Request Occurs from BD DMA Read Engine Read Request
Reserved	1:0	RO	0	Reserved

BD Read DMA Length Debug Register (offset: 0x470C)

Name	Bits	Access	Default Value	Description
rdmad_length_b_2	31:16	RO	0	DMAD Length Read Request Size from holding register 2
rdmad_length_b_1	15:0	RO	0	DMAD Length Read Request Size from holding register 1

BD Read DMA Rstates Debug Register (offset: 0x470C)

Name	Bits	Access	Default Value	Description
rbdi_cnt	31:16	R/W	0	Counting the number of RBDI Requests
Reserved	15:2	RO	0	Reserved bits

Name	Bits	Access	Default Value	Description

Rstate1	1:0	RO	0	parameter idle1_st = 2'b 000;
				parameter ftq_b_req_st = 2'b 01;
				parameter dmad_b_rd_req_st = 2'b 10;
				parameter wait_b_st = 2'b 11;

BD Read Rstate2 Debug Register (offset: 0x4710)

Name	Bits	Access	Default Value	Description
Host Address	31:4	R/W	0	Internal Host Address

Rstate2	3:0	RO	0	parameter idle2_st = 4'b0000;
				parameter non_op_st = 4'b0001;
				parameter wait2_st = 4'b0010;
				parameter wr_mem_st = 4'b0011;
				parameter pre_wr_ftq_st = 4'b0100;
				parameter intr_st = 4'b0101;
				parameter rdr_err_st = 4'b1010;
				parameter wait_st = 4'b0111;
				parameter wr_mis_bd_req_st = 4'b1000;
				parameter wr_mbuf0_req_st = 4'b1001;

BD Read DMA BD Status Debug Register (offset: 0x4714)

Name	Bits	Access	Default Value	Description
rlctrl	31:23	R/W	0	Amount of available free space in the DMA Read FIFO: Write Pointer – Read Pointer
Dmad_load_and_mem_ok	22	R/W	0	Internal state of Dmad_load_and_mem_ok signal
Int_rh_dmad_load	21	R/W	0	Internal state of Int_rh_dmad_load signal
Rh_dmad_load_fst	20	R/W	0	Internal state of Rh_dmad_load_fst signal
Rh_dmad_done_syn3	19	R/W	0	Internal state of Rh_dmad_done_syn3signal
Rh_dmad_load_en	18	R/W	0	Internal state of Rh_dmad_load_en signal
Rh_dmad_no_empty	17	R/W	0	Internal state of Rh_dmad_no_empty signal
Hold_dmad_n_empty	16	R/W	0	Internal state of Hold_dmad_n_empty signal
Rwr_ptr	15:14	R/W	0	DMA Read Write FIFO Pointer
Rrd_ptr	13:12	R/W	0	DMA Read Read FIFO Pointer

Name	Bits	Access	Default Value	Description
Dmad_pnt2	11:10	R/W	0	DMADPointer2
Dmad_pnt1	9:8	R/W	0	DMAD Pointer1
Dmad_pnt0	6:7	R/W	0	DMAD Pointer0
Dmad_pnt	5:4	R/W	0	Final DMAD Pointer
Reserved	3	R/W	0	Reserved
Bd_non_Mbuf	2	RO	0	1: BD Request 0: Mbuf Request
Fst_bd_mbuf	1	RO	0	1: First BD_Mbuf Request 0: Not First_BD_Mbuf_Request
Lst_bd_mbuf	0	RO	0	1: Last BD_Mbuf Request 0: Not Last_BD_Mbuf_Request

BD Read DMA Request Pointer Debug Register (offset: 0x4718)

Name	Bits	Access	Default Value	Description
ih_dmad_length	31:16	R/W	0	Dmad Length request size
Reserved	15:13	RO	0	Reserved
Txmbuf_left	12:5	RO	0	Amount of TXMBuf Left
Rh_dmad_load_en	4	RO	0	State of rh_dmad_load_en signal
Rftq_d_dmad_pnt	3:2	RO	00	FTQ Data Dmad Pointer
Rftq_b_dmad_pnt	1:0	RO	0	Ftq BD DMAD Pointer

BD Read DMA hold_d_dmad Debug Register (offset: 0x471C)

Name	Bits	Access	Default Value	Description
Reserved	31:4	RO	0	Reserved
Rhold_b_dmad	3:2	RO	0	Number of outstanding BD dmad request
Reserved	1:0	RO	0	Not Used

BD Read DMA Length and Address Index Debug Register (offset: 0x4720)

Name	Bits	Access	Default Value	Description
Rdma_rd_length	31:16	R/W	0	Dmad Length request size
Reserved	15:0	RO	0	Reserved

BD Read DMA Address Index Debug Register (offset: 0x4724)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	R/W	0	Reserved
H_host_addr_i	4:0	RO	0	Internal Host Address

BD Read DMA PCIE Debug Status Register (offset: 0x4728)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_term	31:28	RO	0	Latched Termination Code
Reserved	27	RO	0	Reserved
Lt_too_lg	26	RO	0	DMA Too Large Error
Lt_dma_reload	25	RO	0	State of dmad Load signal
Lt_dma_good	24			State of dmad good signal
Cur_trans_active	23			State of cur_trans_active signal
DRPCIREQ	22			State of DR PCI Request signal
Dr_pci_word_swap	21			State of dr_pci_word_swap signal
Dr_pci_byte_swap	20			State of dr_pci_byte_swap signal
New_slow_core_clk_mode	19			State of gated new_slow_core_clock mode signal which is a function of the following input: new_slow_core_clk_mode = SLOW_CORE_CLK_MODE | slow_down_request_syn2 | rbd_non_mbuf & cq25155_fix_enable;
Rbd_non_mbuf	18			State of bd_non_mbuf signal
Rfst_bd_mbuf	17			State of first_bd_Mbuf signal
Rlst_bd_mbuf	16			State of last BD_Mbuf Signal
Dr_pci_len	15:0			Indicates the amount of dma read request

BD Read DMA PCIE DMA Read Request Address Debug Register (offset: 0x472C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Dr_pci_ad31:16	31:16	RO	0	Upper 16-bit of dma read request address
Dr_pci_ad	15:0	RO	0	Lower 16-bit of dma read request address

BD Read DMA PCIE DMA Request Length Debug Register (offset: 0x4730)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	Reserved
Rdma_len	31:0	RO	0	DMA Read Request Length to Host

BD Read DMA FIFO1 Debug Register (offset: 0x4734)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	Upper 16-bit of dma read request address
C_write_addr	8:0	RO	0	Core Clock Write Address Pointer

BD Read DMA FIFO2 Debug Register (offset: 0x4738)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:18	RO	0	Reserved
Rlctrl_in	17:9	RO	0	Amount of available free space in the DMA Read FIFO: Write Pointer – Read Pointer
C_read_addr	8:0	RO	0	Core Clock Read Address Pointer

BD Read DMA Reserved Control Register (offset: 0x4770)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RW	0	Reserved
Select fed Enable_bd	20	RW	0	Ensure only 1 request is generated upon any condition where the core clock is switching from slow to fast or vice-versa. ***Note:*** In Soledad this field is mapped to bit-0 of this register.
FIFO High Mark_bd	19:12	RW	0xC0	These bits control the FIFO HI Mark[7:0]. It is concatenated with 4778[3], 4774[3], and 4770[19:12] to form the FIFO_HI_MARK[9:0]. Default value is 00_1100_0000.
FIFO Low Mark_bd	11:4	RW	0x40	These bits control the FIFO LO Mark[7:0]. This parameter is formed by concatenating the 4778[2], 4774[2], and 4770[11:4] to form the FIFO_LO_MARK[9:0]. Default value is 00_0100_0000.

Name	*Bits*	*Access*	*Default Value*	*Description*
Slow Clock Fix Disable_bd	3	RW	0	When cleared, it enables the fix to cover a corner case in the link idle mode to allow the DMA Read request to be generated when the core clock transitions from slow to fast
Enable hardware fix 25155 CQ25155 fix enable	2	RW	1	When set, enables hardware fix 25155. When set, this bit enables the fix for CQ25155, where a DMA FIFO overrun occurs if a large number of Rx BDs are fetched while the Tx MBUF is full and the Read DMA FIFO is empty.
Reserved	1:0	RW	0	Reserved

BD Read DMA Flow Reserved Control Register (offset: 0x4774)

Name	*Bits*	*Access*	*Default Value*	*Description*
FIFO_threshold_bd_req	31:24	RW	0	This register contains various controls to configure hardware fix 27862. This register contains various controls to fix CQ27862.
Reserved	23:7	RW	0	Reserved
FIFO Available Mode_bd	6	RW	0	BD DMA Read Engine FIFO Available Mode
Reserved	5:4	RW	0	Reserved
FIFO Hi Mark MSB_bd[8]	3	RW	0x0	This bit controls the FIFO HI Mark[8]. It is concatenate with 4778[3], 4774[3], and 4770[19:12] to form the FIFO_HI_MARK_BD[9:0]. Default value is 00_1100_0000.
FIFO Low Mark MSB_bd[8]	2	RW	0x0	This bit controls the FIFO LO Mark[8]. This parameter is formed by concatenating the 4778[2], 4774[2], and 4770[11:4] to form the FIFO_LO_MARK[9:0]. Default value is 00_0100_0000.
FIFO_threshold_bd_req [8]	1	RW	0x1	This bit controls the FIFO Threshold BD Request[8]. This parameter is formed by concatenating the 4778[1], 4774[1] and 4774[31:24] to form the fifo_threshold_bd_req[9:0]. Default value is 01_0000_0000. Only when the amount of data in the FIFO is less than this parameter, then a BD Fetch request is allowed to go upstream.
Reserved	0	RW	0x0	Reserved

BD Read DMA Corruption Enable Control Register (offset: 0x4778)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:4	RW	0	Reserved
FIFO Hi Mark MSB_bd	3	RW	0x0	This bit controls the FIFO HI Mark[9]. It is concatenate with 4778[3], 4774[3], and 4770[19:12] to form the FIFO_HI_MARK_BD[9:0] Default value is 00_1100_0000
FIFO Low Mark MSB_bd	2	RW	0x0	This bit controls the FIFO LO Mark[9]. This parameter is formed by concatenating the 4778[2], 4774[2], and 4770[11:4] to form the FIFO_LO_MARK[9:0] Default value is 00_0100_0000
FIFO_threshold_bd_ req msb	1	RW	0x0	This bit controls the FIFO Threshold BD Request[9]. This parameter is formed by concatenating the 4778[1], 4774[1] and 4774[31:24] to form the fifo_threshold_bd_req[9:0] Default value is 01_0000_0000 Only when the amount of data in the FIFO is less than this parameter, then a BD Fetch request is allowed to go upstream.
Reserved	0	RW	0	Reserved

RDMA Registers

All registers reset are core reset unless specified.

Read DMA Mode Register (offset: 0x4800)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	–
In_band_vtag_enable	29	RW	1	In-Band VLAN tag enable
Hardware IPv6 Post-DMA Processing Enable	28	RW	1	Enables hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets. This bit when clear disables IPV6 Processing.
Hardware IPv4 Post-DMA Processing Enable	27	RW	0	Enables hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets. The functionality of this bit in the BCM5725/ BCM5762/BCM57767 is identical to previous generation controllers. This bit is tcp Segmentation Enable bit
Post-DMA Debug Enable	26	RW	0	When this bit is set, the Send Data Completion State Machine will be halted if the Post-DMA bit of the Send BD is set
Address Overflow Error Logging Enable	25	RW	0	This bit when set, enables the address overflow error to be generated when the DMA Read Engine performs a DMA operation that crosses a 4G boundary. This error is reported in bit 3 of the DMA Read Status Register. Subsequently, this will generate an internal event to interrupt the internal CPU and the DMA Read Engine will lock up after detecting this error. So it's recommended that this bit should not be set by firmware or software. 1: Enable Address Overflow Error Logging 0: Disable Address Overflow Error Logging.
Mmrr_disable	24	RW	0	MMRR Disable 1 = Disable 0 = Enable
Jumbo_2k_mmrrr_mode	23	RW	0	Jumbo 2KB MMRR Mode 1 = Enable 0 = Disable For this mode to work, the driver needs to follow the following rules: **1.** Jumbo Packets with 1 SBD can have up to 9.6 KB as specified in the MRD for Jumbo Packet support. **2.** For jumbo packets with multiple SBDs, each SBD must be 2 KB or less.
Reserved	22:18	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Request Burst Length	17:16	RW	3	The two bits define the burst length that the RDMA read engine would request to the PCI block. Set to 256B if slow core clock is enabled (See Clock Control Register 0x74 bit-19). 00 = 128B 01 = 256B 10 = 512B 11 = 4 KB
Reserved	15:14	RO	0	–
MBUF SBD Corruption Attn Enable	13	RW	0	–
MBUF RBD Corruption Attn Enable	12	RW	0	–
BD SBD Corruption Attn Enable	11	RW	0	–
Reserved	12:11	RO	0	–
Read DMA PCI-X Split Transaction Timeout Expired Attention Enable	10	RW	0	Enable read DMA PCI-X split transaction timeout expired attention.
Read DMA Local Memory Write Longer Than DMA Length Attention Enable	9	RW	0	Enable Read DMA Local Memory Write Longer Than DMA Length Attention.
Read DMA PCI FIFO Overread Attention Enable	8	RW	0	Enable Read DMA PCI FIFO Overread Attention (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Attention Enable	7	RW	0	Enable Read DMA PCI FIFO Underrun Attention
Read DMA PCI FIFO Overrun Attention Enable	6	RW	0	Enable Read DMA PCI FIFO Overrun Attention
Read DMA PCI Host Address Overflow Error Attention Enable	5	RW	0	Enable Read DMA PCI Host Address Overflow Error Attention. A host address overflow occurs when a single DMA read begins at an address below 4 GB and ends on an address above 4 GB. This is a fatal error.
Read DMA PCI Parity Error Attention Enable	4	RW	0	Enable Read DMA PCI Parity Error Attention
Read DMA PCI Master Abort Attention Enable	3	RW	0	Enable Read DMA PCI Master Abort Attention
Read DMA PCI Target Abort Attention Enable	2	RW	0	Enable Read DMA PCI Target Abort Attention

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable	1	RW	1	This bit controls whether the Read DMA state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Read DMA state machine is reset. This is a self-clearing bit.

Read DMA Status Register (offset: 0x4804)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:11	RO	0	–
MBUF SBD Corruption Attn Status	13	W2C	0	–
MBUF RBD Corruption Attn Status	12	W2C	0	–
BD SBD Corruption Attn Status	11	W2C	0	–
Read DMA PCI-X Split Transaction Timeout Expired	10	W2C	0	Read DMA PCI-X split transaction timeout expired.
Read DMA Local Memory Write Longer Than DMA Length Error	9	W2C	0	Read DMA Local Memory Write Longer Than DMA Length Error.
Read DMA PCI FIFO Overread Error	8	W2C	0	Read DMA PCI FIFO Overread Error (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Error	7	W2C	0	Read DMA PCI FIFO Underrun Error
Read DMA PCI FIFO Overrun Error	6	W2C	0	Read DMA PCI FIFO Overrun Error
Read DMA PCI Host Address Overflow Error	5	W2C	0	Read DMA PCI Host Address Overflow Error. A host address overflow occurs when a single DMA read begins at an address below a multiple of 4 GB and ends at an address above the same multiple of 4 GB (i.e., the host memory address transitions from 0xXXXXXXXX_FFFFFFFF to 0xYYYYYYYY_00000000 in a single read). This is a fatal error.
Read DMA Completion Timer Timeout	4	W2C	0	Read DMA PCIe Completion Timer Timeout
Read DMA Completer Abort	3	W2C	0	Read DMA PCIe Completer Abort
Read DMA Unsupported Request	2	W2C	0	Read DMA PCIe Unsupported Request
Reserved	1:0	RO	0	–

Read DMA Programmable IPv6 Extension Header Register (offset: 0x4808)

Name	*Bits*	*Access*	*Default Value*	*Description*
Programmable Extension Header Type #2 Enable	31	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [15:8] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [15:8].
Programmable Extension Header Type #1 Enable	30	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [7:0] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [7:0].
Reserved	29:16	RO	0	Reserved bits
Programmable Extension Header Type #2	15:8	R/W	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	R/W	0	These bits contain the programmable extension header value for programmable header #1.

Read DMA Rstates Debug Register (offset: 0x480c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	R/W	0	Reserved
Rstate3	5:4	RO	00	00: idle 01: Mbuf Request State 10: Mbuf Ready State
Reserved	3	RO	0	Reserved bits

Rstate1	2:0	RO	0	parameter idle1_st = 3’b 000;
				parameter ftq_b_req_st = 3’b 001;
				parameter ftq_d_req_st = 3’b 010;
				parameter dmad_b_rd_req_st = 3’b 011;
				parameter dmad_d_rd_req_st = 3’b 100;
				parameter dmad_wr_req_st = 3’b 101;
				parameter wait_b_st = 3’b 110;
				parameter wait_d_st = 3’b 111;

Read DMA Rstate2 Debug Register (offset: 0x4810)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	R/W	0	Reserved

Rstate2	4:0	RO	0	parameter idle2_st = 5'b 00000;
				parameter non_op_st = 5'b 00001;
				parameter wait2_st = 5'b 00010;
				parameter wr_mem_req0_pre_st = 5'b 00011;
				parameter wr_mem_req0_st = 5'b 00100;
				parameter wr_mem_req1_st = 5'b 00101;
				parameter wr_mem_req2_st = 5'b 00110;
				parameter wr_mem_req3_st = 5'b 00111;
				parameter wr_mem_req4_st = 5'b 01000;
				parameter wr_mem_st = 5'b 01001;
				parameter wr_mbuf_hdr_req_st = 5'b 01010;
				parameter wr_mbuf_hdr_st = 5'b 01011;
				parameter mbuf_get_st = 5'b 01100;
				parameter wr_fd_req_st = 5'b 01101;
				parameter wr_fd_st = 5'b 01110;
				parameter pre_wr_ftq_st = 5'b 01111;
				parameter intr_st = 5'b 10000;
				parameter rdr_err_st = 5'b 10001;
				parameter wait_st = 5'b 10010;
				parameter wr_mem_req5_st = 5'b 10011;
				parameter chk_hst_addr_st = 5'b 10100;
				parameter wr_full_mbuf_hdr_req_st = 5'b 10101;
				parameter wr_full_mbuf_done_st = 5'b 10111;
				parameter wr_mis_bd_req_st = 5'b 11000;
				parameter wr_mem_req_st = 5'b 11001;
				parameter wr_mbuf0_req_st = 5'b 11010;
				parameter wait4_proc_done_st = 5'b 11011;
				parameter wr_mbuf1_req_st = 5'b 11100;
				parameter get_more_mbuf_st = 5'b 11101;

Read DMA BD Status Debug Register (offset: 0x4814)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	R/W	0	Reserved
Bd_non_Mbuf	2	RO	0	1: BD Request 0: Mbuf Request

Name	Bits	Access	Default Value	Description
Fst_bd_mbuf	1	RO	0	1: First BD_MBuf Request 0: Not First_BD_Mbuf_Request
Lst_bd_mbuf	0	RO	0	1: Last BD_MBuf Request 0: Not Last_BD_Mbuf_Request

Read DMA Request Pointer Debug Register (offset: 0x4818)

Name	Bits	Access	Default Value	Description
ih_dmad_length	31:16	R/W	0	Dmad Length request size
Reserved	15:10	RO	0	Reserved
Txmbuf_left	9:4	RO	0	Amount of TXMBuf Left
Rh_dmad_load_en	3	RO	0	State of rh_dmad_load_en signal
Rftq_d_dmad_pnt	2:1	RO	00	FTQ Data Dmad Pointer
Rftq_b_dmad_pnt	0	RO	0	Ftq BD DMAD Pointer

Read DMA hold_d_dmad Debug Register (offset: 0x481c)

Name	Bits	Access	Default Value	Description
Reserved	31:2	RO	0	Reserved
Rhold_d_dmad	1:0	RO	0	Number of outstanding data dmad request

Read DMA Length and Address Index Debug Register (offset: 0x4820)

Name	Bits	Access	Default Value	Description
Rdma_rd_length	31:16	R/W	0	Dmad Length request size
Mbuf_addr_idx	15:0	RO	0	6-bit allocation tx mbuf index; upper bits are zeroes

Read DMA Mbuf Byte Count Debug Register (offset: 0x4824)

Name	Bits	Access	Default Value	Description
Reserved	31:4	RO	0	Reserved
Rmbuf_byte_cnt	3:0	RO	0	Mbuf Byte Count

Read DMA PCIe Mbuf Byte Count Debug Register (offset: 0x4828)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_term	31:28	RO	0	Latched Termination Code
Reserved	27	RO	0	Reserved
Lt_too_lg	26	RO	0	DMA Too Large Error
Lt_dma_reload	25	RO	0	State of dmad Load signal
Lt_dma_good	24	–	–	State of dmad good signal
Cur_trans_active	23	–	–	State of cur_trans_active signal
DRPCIREQ	22	–	–	State of DR PCI Request signal
Dr_pci_word_swap	21	–	–	State of dr_pci_word_swap signal
Dr_pci_byte_swap	20	–	–	State of dr_pci_byte_swap signal
New_slow_core_clk_mode	19	–	–	State of gated new_slow_core_clock mode signal which is a function of the following input: new_slow_core_clk_mode = SLOW_CORE_CLK_MODE | slow_down_request_syn2 | rbd_non_mbuf and cq25155_fix_enable;
Rbd_non_mbuf	18	–	–	State of bd_non_mbuf signal
Rfst_bd_mbuf	17	–	–	State of first_bd_Mbuf signal
Rlst_bd_mbuf	16	–	–	State of last BD_Mbuf Signal
Dr_pci_len	15:0	–	–	Indicates the amount of DMA read request

Read DMA PCIe DMA Read Request Address Debug Register (offset: 0x482c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Dr_pci_ad31:16	31:16	RO	0	Upper 16-bit of DMA read request address
Dr_pci_ad	15:0	RO	0	Lower 16-bit of DMA read request address

Read DMA PCIe DMA Request Length Debug Register (offset: 0x4830)

Name	*Bits*	*Access*	*Default Value*	*Description*
Rdma_len	31:0	RO	0	DMA Read Request Length to Host

Read DMA FIFO1 Debug Register (offset: 0x4834)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0	Upper 16-bit of DMA read request address
C_write_addr	7:0	RO	0	Core Clock Write Address Pointer

Read DMA FIFO2 Debug Register (offset: 0x4838)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	Upper 16-bit of DMA read request address
Rlctrl_in	15:8	RO	0	Amount of available free space in the DMA Read FIFO: Write Pointer – Read Pointer
C_read_addr	7:0	RO	0	Core Clock Read Address Pointer

Read DMA Packet Request Debug Register (offset: 0x4840)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0	Reserved
Pkt_req_cnt(7:0)	7:0	RO	0	Request Counts

Read DMA Packet Request Debug Register (offset: 0x4844)

Name	*Bits*	*Access*	*Default Value*	*Description*
Sdc_ack_cnt	31:0	RO	0	Send Data Completion Ack. Counts

Read DMA Packet Request Debug Register (offset: 0x4848)

Name	*Bits*	*Access*	*Default Value*	*Description*

Cs	31:28	RO	0	parameter IDLE = 4'b 0000;
				parameter MEM_RD_REQ0 = 4'b 0011;
				parameter MEM_RD_REQ1 = 4'b 0100;
				parameter MEM_WR_REQ0 = 4'b 0101;
				parameter MEM_WR_REQ1 = 4'b 0110;
				parameter WR_MBUF0 = 4'b 0111;
				parameter WR_MBUF1 = 4'b 1000;
				parameter FD_REQ = 4'b 1001;
				parameter FD_RD = 4'b 1010;
				parameter SDC_WR = 4'b 1011;
				parameter CHK_FD_FLAGS = 4'b 1100;

Reserved	27:26	RO	0	Reserved
Lt_fst_seg	25	RO	0	Indicates First Segment
Lt_lst_seg	24	RO	0	Indicates Last Segment
Lt_mem_ip_hdr_ofst	23:16	RO	0	IP Header Offset
Lt_mem_tcp_hdr_ofst	15:8	RO	0	TCP Header Offset
Reserved	7	RO	0	Reserved
Pre_sdcq_pkt_cnt	6:0	RO	0	Pre-Send Data Completion Queue Packet Count

Read DMA TCP Checksum Debug Register (offset: 0x484c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Reserved
Fd_addr_req(9:4)	29:24	RO	0	Holding the content of Frame Descriptor Address Register
Lt_mem_data_ofst	23:16	RO	0	Indicates First Segment
Lt_mem_tcp_chksum	15:0	RO	0	Hold the current value of the Tcp_checksum calculation

Read DMA Ip/TCP Header Checksum Debug Register (offset: 0x4850)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_mem_ip_chksum	31:16	RO	0	Hold the current value of the ip Checksum Calculation
Lt_mem_tcphdr_chksum	15:0	RO	0	Hold the current value of the Tcp Header_checksum calculation

Read DMA Pseudo Checksum Debug Register (offset: 0x4854)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_mem_pse_chksum_no_tcplen	31:16	RO	0	Hold the current value of the Pseudo-Checksum Calculation
Lt_mem_pkt_len	15:0	RO	0	Hold the current value of the packet length

Read DMA Mbuf Address Debug Register (offset: 0x4858)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Reserved
Mbuf1_addr(9:4)	29:24	RO	0	Mbuf1 Address Pointer
Reserved	23:22	RO	0	Reserved
Mbuf0_addr(9:4)	21:16	RO	0	Mbuf0 Address Pointer
Reserved	15:14	RO	0	Reserved
Pre_Mbuf1_addr(9:4)	13:8	RO	0	Pre_Mbuf1 Address Pointer
Reserved	7:6	RO	0	Reserved
Pre_Mbuf0_addr(9:4)	5:0	RO	0	Pre_Mbuf0 Address Pointer

Registers for the BCM5725/BCM5762/BCM57767

Read DMA Reserved Control1 Register- Debug Controls (offset: 0x4890)

Name	*Bits*	*Access*	*Default Value*	*Description*
Txmbuf_margin	31:21	R/W	0x140	Fix for CQ27862: This parameter is used to calculate how much TXMBUF Memory is available for storing the payload data.
FIFO Available Mode	20	RW	1	LSO/Jumbo DMA Read FIFO Available Mode.
FIFO High Mark	19:12	RW	0x90	These bits control the FIFO HI Mark[7:0]. It is concatenate with 48a0[3], 4894[3], and 4890[19:12] to form the FIFO_HI_MARK[9:0] Default value is 00_1001_0000
FIFO Low Mark	11:4	RW	0x40	These bits control the FIFO LO Mark[7:0]. This parameter is formed by concatenating the 48a3[2], 4894[2], and 4890[11:4] to form the FIFO_LO_MARK[9:0]. Default value is 00_0100_0000.
Slow Clock Fix Disable	3	RW	0	When cleared, it enables the fix to cover a corner case in the link idle mode to allow the DMA Read request to be generated when the core clock is transitioning from slow to fast.
Enable hardware fix 25155 CQ25155 fix enable	2	RW	0	When set, enables hardware fix 25155 When set, this bit enables the fix for CQ25155, where a DMA FIFO overrun occurs if a large number of Rx BDs are fetched while the Tx MBUF is full and the Read DMA Fifo is empty. This bit is NOT applicable in the LSO DMA Read Engine with MMRR architecture. ***Note:*** For the BCM5725/BCM5762/BCM57767-PG106, there is no need to touch this bit in the driver; keep the default value.
Late Collision Fix Enable	1	RW	1	0 means disable fix 1 means enable fix
Select fed Enable	0	RW	0	Ensure only 1 request is generated upon any condition where the core clock is switching from slow to fast or vice-versa. Note; in Soledad, this field is mapped to bit-0 of this register.

Read DMA Flow_Reserved Control2 Register (offset: 0x4894)

Name	*Bits*	*Access*	*Default Value*	*Description*
FIFO_threshold_bd_req	31:24	RW	0	This register contains various controls to configure hardware fix 27862 This register contains various controls to fix CQ27862 This is not applicable to the LSO/Jumbo DMA Read Engine in the MMRR Architecture
FIFO_threshold_mbuf_req	23:16	RW	0x30	This parameter is compared with the FIFO to determine whether or not we have enough space to submit a request to the PCIE Host.
MBUF Threshold MBUF Request	15:8	RW	0x54	This parameter is compared with the TXMBUF to determine whether we have enough mbuf to submit a request to the PCIE Host.
Sb_ooo_fix_enable	7	RW	0	Status Block Out-of-Order Fix Enable 0 means fix enabled 1 means fix disabled
Reserved	6:4	RW	0	Reserved
FIFO Hi Mark MSB	3	RW	0x0	This bit controls the FIFO HI Mark[8]. It is concatenate with 48a0[3], 4894[3], and 4890[19:12] to form the FIFO_HI_MARK[9:0] Default value is 00_1001_0000
FIFO Low Mark MSB	2	RW	0x0	This bit controls the FIFO LO Mark[8]. This parameter is formed by concatenating the 48a3[2], 4894[2], and 4890[11:4] to form the FIFO_LO_MARK[9:0] Default value is 00_0100_0000
FIFO_threshold_bd_req msb	1	RW	0x0	This bit controls the FIFO Threshold BD Request[8]. This parameter is formed by concatenating the 48a0[1], 4894[1] and 4894[31:24] to form the fifo_threshold_bd_req[9:0] Default value is 00_0000_0000 This parameter is not used in the LSO DMA Read Engine with MMRR Architecture
FIFO_threshold_mbuf_req msb	0	RW	0x0	This bit controls the FIFO Threshold Mbuf Request[8]. This parameter is formed by concatenating the 48a0[0], 4894[0] and 4894[23:16] to form the fifo_threshold_mbuf_req[9:0] Default value is 00_0011_0000

Read DMA Corruption Enable Register – Miscellaneous Control (offset: 0x48a0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lcrc_dr_fix_enable	31	RW	0	1: enable fix 0: disable fix
Lcrc_dr_fix_enable2	30	RW	0	1: enable fix 0: disable fix
Reserved	29:8	RW	0xF00	–
CQ35774_Fix_Enable	7	RW	0	This control bit is used to enable the fix for CQ35774 1: Disable fix 0: enable fix
Reserved	6	RW	0	Reserved
CQ33951_Fix_Disable	5	RW	0	This control bit is used to enable the fix for CQ33951 where the IP Header Checksum Corruption occurs when an IPV4 payload contents match an IPV6 Header Type. 1: Disable fix 0: enable fix
Reserved	4	RW	1	Reserved
FIFO Hi Mark[9]	3	RW	0	This bit controls the FIFO HI Mark[9]. It is concatenate with 48a0[3], 4894[3], and 4890[19:12] to form the FIFO_HI_MARK[9:0]. Default value is 00_1001_0000.
FIFO LO Mark[9]	2	RW	0	This bit controls the FIFO LO Mark[9]. This parameter is formed by concatenating the 48a3[2], 4894[2], and 4890[11:4] to form the FIFO_LO_MARK[9:0]. Default value is 00_0100_0000.
FIFO Threshold BD Request [9]	1	RW	0	This bit controls the FIFO Threshold BD Request[9]. This parameter is formed by concatenating the 48a0[1], 4894[1] and 4894[31:24] to form the fifo_threshold_bd_req[9:0]. Default value is 00_0000_0000. This parameter is not used in the LSO DMA Read Engine with MMRR Architecture.
FIFO Threshold Mbuf Request [9]	0	RW	0	This bit controls the FIFO Threshold Mbuf Request[9]. This parameter is formed by concatenating the 48a0[0], 4894[0] and 4894[23:16] to form the fifo_threshold_mbuf_req[9:0]. Default value is 00_0011_0000.

Non_LSO DMA Read Engine

Non_LSO Read DMA Mode Register (offset: 0x4900)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	–
Address Overflow Error Logging Enable	25	RW	0	This bit when set, enables the address overflow error to be generated when the DMA Read Engine performs a DMA operation that crosses a 4G boundary. This error is reported in bit 3 of the DMA Read Status Register. Subsequently, this will generate an internal event to interrupt the internal CPU and the DMA Read Engine will lock up after detecting this error. So it's recommended that this bit should not be set by firmware or software. 1: Enable Address Overflow Error Logging 0: Disable Address Overflow Error Logging.
Reserved	24:18	RO	0	–
PCI Request Burst Length	17:16	RW	11	The two bits define the burst length that the RDMA read engine would request to the PCI block. • 00 = 128B • 01 = 256B • 10 = 512B • 11 = 4KB
Reserved	15:14	RO	0	–
Attention Enables	13:2	RO	0	Enable read DMA PCI-X split transaction timeout expired attention.
Enable	1	RO	0	This bit controls whether the Read DMA state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read. This bit is derived from Register 4800[1]
Reset	0	RO	0	When this bit is set to 1, the Read DMA state machine is reset. This is a self-clearing bit. This bit is derived from Register 4800[0]

Non_LSO Read DMA Status Register (offset: 0x4904)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:11	RO	0	–
Malformed TLP or Poison TLP Error Detected	10	W2C	0	Malformed TLP or Poisoned TLP Occurs from BD DMA Read Engine Read Request

Name	Bits	Access	Default Value	Description
Read DMA Local Memory Write Longer Than DMA Length Error	9	W2C	0	Non-LSO Read DMA Local Memory Write Longer Than DMA Length Error.
Read DMA PCI FIFO Overread Error	8	W2C	0	Non-LSO Read DMA PCI FIFO Overread Error (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Error	7	W2C	0	Non-LSO Read DMA PCI FIFO Underrun Error
Read DMA PCI FIFO Overrun Error	6	W2C	0	Non-LSO Read DMA PCI FIFO Overrun Error
Read DMA PCI Host Address Overflow Error	5	W2C	0	Non-LSO Read DMA PCI Host Address Overflow Error. A host address overflow occurs when a single DMA read begins at an address below a multiple of 4 GB and ends at an address above the same multiple of 4 GB (i.e., the host memory address transitions from 0xXXXXXXXX_FFFFFFFF to 0xYYYYYYYY_00000000 in a single read). This is a fatal error.
DMA Read Completion Timeout	4	W2C	0	Completion Timeout Occurs from Non-LSO DMA Read Engine Read Request
Completion Abort Error	3	W2C	0	Completion Abort Occurs from Non-LSO DMA Read Engine Read Request
Unsupported Request Error Detected	2	W2C	0	Unsupported Request Occurs from Non-LSO DMA Read Engine Read Request
Reserved	1:0	RO	0	–

Non_LSO Read DMA Programmable IPv6 Extension Header Register (offset: 0x4908)

Name	Bits	Access	Default Value	Description
Programmable Extension Header Type #2 Enable	31	RO	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [15:8] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [15:8].
Programmable Extension Header Type #1 Enable	30	RO	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [7:0] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [7:0].
Reserved	29:16	RO	0	Reserved bits
Programmable Extension Header Type #2	15:8	RO	0	These bits contain the programmable extension header value for programmable header #2.

Name	***Bits***	***Access***	***Default Value***	***Description***
Programmable Extension Header Type #1	7:0	RO	0	These bits contain the programmable extension header value for programmable header #1.

Non_LSO Read DMA Rstates Debug Register (offset: 0x490C)

Name	***Bits***	***Access***	***Default Value***	***Description***
Reserved	31:1 1	R/W	0	Reserved
SDI_DR_WR	10	RO	00	0: SDI DR_Write Request Low 1: SDI_DR Write Request High
DR_SDI_WR_ACK	9	RO	00	0: DR_SDI Write Not Acknowledged 1: DR_SDI_Write Acknowledged
Non_LSO_Select	8	RO	00	0: Select LSO Request 1: Select Non_LSO Request
Non-LSO_Queue_Full	7	RO	00	0: Non-LSO Queue Not Full 1: Non-LSO Queue Full
Non-LSO_busy	6	RO	00	0: Non-LSO Not Busy 1: Non-LSO Busy
Rstate3	5:4	RO	00	00: idle 01: Mbuf Request State 10: Mbuf Ready State
Reserved	3	RO	0	Reserved bits

Rstate1	2:0	RO	0	parameter idle1_st = 3'b 000;
				parameter ftq_b_req_st = 3'b 001;
				parameter ftq_d_req_st = 3'b 010;
				parameter dmad_b_rd_req_st = 3'b 011;
				parameter dmad_d_rd_req_st = 3'b 100;
				parameter dmad_wr_req_st = 3'b 101;
				parameter wait_b_st = 3'b 110;
				parameter wait_d_st = 3'b 111;

Non_LSO Read DMA Rstate2 Debug Register (offset: 0x4910)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	R/W	0	Reserved

Rstate2	4:0	RO	0	parameter idle2_st = 5'b 00000;
				parameter non_op_st = 5'b 00001;
				parameter wait2_st = 5'b 00010;
				parameter wr_mem_req0_pre_st = 5'b 00011;
				parameter wr_mem_req0_st = 5'b 00100;
				parameter wr_mem_req1_st = 5'b 00101;
				parameter wr_mem_req2_st = 5'b 00110;
				parameter wr_mem_req3_st = 5'b 00111;
				parameter wr_mem_req4_st = 5'b 01000;
				parameter wr_mem_st = 5'b 01001;
				parameter wr_mbuf_hdr_req_st = 5'b 01010;
				parameter wr_mbuf_hdr_st = 5'b 01011;
				parameter mbuf_get_st = 5'b 01100;
				parameter wr_fd_req_st = 5'b 01101;
				parameter wr_fd_st = 5'b 01110;
				parameter pre_wr_ftq_st = 5'b 01111;
				parameter intr_st = 5'b 10000;
				parameter rdr_err_st = 5'b 10001;
				parameter wait_st = 5'b 10010;
				parameter wr_mem_req5_st = 5'b 10011;
				parameter chk_hst_addr_st = 5'b 10100;
				parameter wr_full_mbuf_hdr_req_st = 5'b 10101;
				parameter wr_full_mbuf_done_st = 5'b 10111;
				parameter wr_mis_bd_req_st = 5'b 11000;
				parameter wr_mem_req_st = 5'b 11001;
				parameter wr_mbuf0_req_st = 5'b 11010;
				parameter wait4_proc_done_st = 5'b 11011;
				parameter wr_mbuf1_req_st = 5'b 11100;
				parameter get_more_mbuf_st = 5'b 11101;

Non_LSO Read DMA BD Status Debug Register (offset: 0x4914)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:3	R/W	0	Reserved
Bd_non_Mbuf	2	RO	0	1: BD Request 0: Mbuf Request
Fst_bd_mbuf	1	RO	0	1: First BD_Mbuf Request 0: Not First_BD_Mbuf_Request
Lst_bd_mbuf	0	RO	0	1: Last BD_Mbuf Request 0: Not Last_BD_Mbuf_Request

Non_LSO Read DMA Request Pointer Debug Register (offset: 0x4918)

Name	*Bits*	*Access*	*Default Value*	*Description*
ih_dmad_length	31:16	R/W	0	Dmad Length request size
Reserved	15:13	RO	3'b110	Reserved
Txmbuf_left	12:5	RO	0	Amount of TXMBuf Left
Rh_dmad_load_en	4	RO	0	State of rh_dmad_load_en signal
Rftq_d_dmad_pnt	3:2	RO	00	FTQ Data Dmad Pointer
Reserved	1:0	RO	0	Not Used

Non_LSO Read DMA hold_d_dmad Debug Register (offset: 0x491C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:2	RO	0	Reserved
Rhold_d_dmad	1:0	RO	0	Number of outstanding data dmad request

Non_LSO Read DMA Length & Address Debug Register (offset: 0x4920)

Name	*Bits*	*Access*	*Default Value*	*Description*
Rdma_rd_length	31:16	R/W	0	Dmad Length request size
Mbuf_addr_idx	15:0	RO	0	6-bit allocation tx mbuf index; upper bits are zeroes

Non_LSO Read DMA Mbuf Byte Count Debug Register (offset: 0x4924)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:4	RO	0	Reserved
Rmbuf_byte_cnt	3:0	RO	0	Mbuf Byte Count

Non_LSO Read DMA PCIE Debug Status Register (offset: 0x4928)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_term	31:28	RO	0	Latched Termination Code
Reserved	27	RO	0	Reserved
Lt_too_lg	26	RO	0	DMA Too Large Error
Lt_dma_reload	25	RO	0	State of dmad Load signal
Lt_dma_good	24	–	–	State of dmad good signal
Cur_trans_active	23	–	–	State of cur_trans_active signal
DRPCIREQ	22	–	–	State of DR PCI Request signal
Dr_pci_word_swap	21	–	–	State of dr_pci_word_swap signal
Dr_pci_byte_swap	20	–	–	State of dr_pci_byte_swap signal
New_slow_core_clk_mode	19	–	–	State of gated new_slow_core_clock mode signal which is a function of the following input: new_slow_core_clk_mode = SLOW_CORE_CLK_MODE | slow_down_request_syn2 | rbd_non_mbuf & cq25155_fix_enable;
Rbd_non_mbuf	18	–	–	State of bd_non_mbuf signal
Rfst_bd_mbuf	17	–	–	State of first_bd_Mbuf signal
Rlst_bd_mbuf	16	–	–	State of last BD_Mbuf Signal
Dr_pci_len	15:0	–	–	Indicates the amount of DMA read request

Non_LSO Read DMA PCIE DMA Read Request Debug Register (offset: 0x492C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Dr_pci_ad31:16	31:16	RO	0	Upper 16-bit of dma read request address
Dr_pci_ad	15:0	RO	0	Lower 16-bit of dma read request address

Non_LSO Read DMA PCIE DMA Request Length Debug Register (offset: 0x4930)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	Reserved
Rdma_len	31:0	RO	0	DMA Read Request Length to Host

Non_LSO Read DMA FIFO1 Debug Register (offset: 0x4934)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:9	RO	0	Upper 16-bit of dma read request address
C_write_addr	8:0	RO	0	Core Clock Write Address Pointer

Non_LSO Read DMA FIFO2 Debug Register (offset: 0x4938)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:18	RO	0	Reserved
Rlctrl_in	17:9	RO	0	Amount of available free space in the DMA Read FIFO: Write Pointer – Read Pointer
C_read_addr	8:0	RO	0	Core Clock Read Address Pointer

Non_LSO Read DMA Post Processing Module Packet Request Count Register (offset: 0x4940)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0	Reserved
Pkt_req_cnt(7:0)	7:0	RO	0	Request Counts

Non_LSO Read DMA Mbuf Address Debug Register (offset: 0x4960)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	0	Reserved
MACTQ_FULL	25	RO	0	MACTQ Full
TXFIFO_Almost_Underrun	24	RO	0	TX FIFO Almost Underflow
TDE FIFO Entry(7:0)	23:16	RO	0	TCE FIFO Entry = {3'b0, TCE FIFO_EMPTY_WR}
RCMP_Head	15:0	RO	0	Pre_Mbuf0 Address Pointer

Non_LSO Read DMA TCE Debug1 Register (offset: 0x4964)

Name	Bits	Access	Default Value	Description
ODI State_out(3:0)	31:28	RO	0	ODI Output State
ODI State_in(3:0)	27:24	RO	0	ODI Input State
FIFO_ODI_DATA_Ccode	23:22	RO	0	ODI FIFO Data Code
FIFO_ODI_DATA(21:0)	21:0	RO	0	ODI FIFO Data Data

Non_LSO Read DMA TCE Debug2 Register (offset: 0x4968)

Name	Bits	Access	Default Value	Description
Det_Abort_Cnt(7:0)	31:24	RO	0	TCE Detect Abort Count
Reserved	23:0	RO	0	Not Used

Non_LSO Read DMA TCE Debug3 Register (offset: 0x496C)

Name	Bits	Access	Default Value	Description
Reserved	31:28	RO	0	Not Used
TX_Pkt_Cnt(7:0)	27:20	RO	0	TX Packet Count(7:0)
Reserved	19:17	RO	0	Not Used
TCE_MA_REQ	16	RO	0	TCE MA Request
TCE_MA_CMD_LEN(3:0)	15:12	RO	0	TCE MA Request Length
Reserved	11:0	RO	0	Not Used

Non_LSO Read DMA Reserved Control Register (offset: 0x4970)

Name	Bits	Access	Default Value	Description
Txmbuf_margin_nlso	31:21	RW	0	Fix for CQ27862: Non-LSO DMA Read TXMbuf Margin
Reserved	20	RW	0	Not Used
FIFO High Mark	19:12	RW	0x28	Non-LSO DMA Read FIFO Hi Mark
FIFO Low Mark	11:4	RW	0x20	Non-LSO DMA Read Low Mark
Slow Clock Fix Disable	3	RW	0	When cleared, it enables the fix to cover a corner case in the link idle mode to allow the DMA Read request to be generated when the core clock is transitioning from slow to fast
Enable hardware fix 25155 CQ25155 fix enable	2	RW	0	When set, enables hardware fix 25155 When set, this bit enables the fix for CQ25155, where a DMA FIFO overrun occurs if a large number of Rx BDs are fetched while the Tx MBUF is full and the Read DMA Fifo is empty.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	1	RW	1	Not Used
Select FED Enable	0	RW	0	Ensure only one request is generated upon any condition where the core clock is switching from slow to fast or vice-versa. ***Note:*** In Soledad this field is mapped to bit-0 of this register.

Non_LSO Read DMA Flow Reserved Control Register (offset: 0x4974)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RW	0x0	Reserved
FIFO_threshold_mbuf_req msb	23:16	RW	0x20	LSO/Jumbo FIFO Threshold Mbuf Request MSB
MBUF Threshold MBUF Request	15:8	RW	0x54	–
Reserved	7:4	RW	0	Reserved
FIFO High Mark[8]	3	RW	0	–
FIFO Low Mark[8]	2	RW	0	–
Reserved	1	RW	0	–
FIFO_threshold_mbuf_req msb[8]	0	RW	0	–

Non_LSO Read DMA Corruption Enable Control Register (offset: 0x4978)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lcrc_dr_fix_enable	31	RW	0	0 = Fix Disable 1 = Fix Enable
New_length_fix_enable	30	RW	0	0 – Fix Enable 1 – Fix Disable
Reserved	29:22	RO	0	
Cq51816_non_lso_fix_enable	21	RW	0	0 – Fix Enable 1 – Fix Disable
Cq51036_non_lso_fix_enable	20	RW	0	0 – Fix Enable 1 – Fix Disable
Reserved	19:15	RW	0	0 – Fix Enable
Sbd_8b_less_fix_enable3	14	RW	0	0 – Fix Enable
Sbd_8b_less_fix_enable2	13	RW	0	0 = Fix Enable
Mem_too_large_fix_enable2	12	RW	0	0 – Fix Enable
Mem_too_large_fix_enable1	11	RW	0	0 – Fix Enable

Name	***Bits***	***Access***	***Default Value***	***Description***
Mem_too_large_fix_enable	10	RW	0	0 – Fix Enable
Sbd_8b_less_fix_enable_fast_return	9	RW	0	0 – Fix Enable
Sbd_8b_less_fix_enable	8		0	0 – Fix Enable
Enable hardware fix for CQ35774	7	RW	0	CQ35774 involves wiping out the tcp_checksum calculation 0: Enable Fix 1: Disable Fix
Reserved	6:0	RO	0	–

Real-TimeRDMA Registers

All registers reset are core reset unless specified.

Real-TimeRead DMA Mode Register (offset: 0x4A00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
Hardware IPv6 Post-DMA Processing Enable	28	RW	1	Enables hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets. This bit when clear disables IPV6 Processing This bit was not used for projects before BCM5725/ BCM5762/BCM57767
Hardware IPv4 Post-DMA Processing Enable	27	RW	0	Enables hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets. The functionality of this bit in BCM5725/ BCM5762/BCM57767 is identical to previous generation controllers. This bit is tcp Segmentation Enable bit
Post-DMA Debug Enable	26	RW	0	When this bit is set, the Send Data Completion State Machine will be halted if the Post-DMA bit of the Send BD is set
Address Overflow Error Logging Enable	25	RW	0	This bit when set, enables the address overflow error to be generated when the DMA Read Engine performs a DMA operation that crosses a 4G boundary. This error is reported in bit 3 of the DMA Read Status Register. Subsequently, this will generate an internal event to interrupt the internal CPU and the DMA Read Engine will lock up after detecting this error. So it’s recommended that this bit should not be set by firmware or software. 1: Enable Address Overflow Error Logging 0: Disable Address Overflow Error Logging.
Reserved	24:18	RO	0	–
PCI Request Burst Length	17:16	RW	3	The two bits define the burst length that the RDMA read engine would request to the PCI block. Set to 256B if slow core clock is enabled (See Clock Control Register 0x74 bit-19). 00 = 128B 01 = 256B 10 = 512B 11 = 4 KB
Reserved	15:14	RO	0	–
MBUF SBD Corruption Attn Enable	13	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
MBUF RBD Corruption Attn Enable	12	RW	0	–
BD SBD Corruption Attn Enable	11	RW	0	–
Reserved	12:11	RO	0	–
Read DMA PCI-X Split Transaction Timeout Expired Attention Enable	10	RW	0	Enable read DMA PCI-X split transaction timeout expired attention.
Read DMA Local Memory Write Longer Than DMA Length Attention Enable	9	RW	0	Enable Read DMA Local Memory Write Longer Than DMA Length Attention.
Read DMA PCI FIFO Overread Attention Enable	8	RW	0	Enable Read DMA PCI FIFO Overread Attention (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Attention Enable	7	RW	0	Enable Read DMA PCI FIFO Underrun Attention
Read DMA PCI FIFO Overrun Attention Enable	6	RW	0	Enable Read DMA PCI FIFO Overrun Attention
Read DMA PCI Host Address Overflow Error Attention Enable	5	RW	0	Enable Read DMA PCI Host Address Overflow Error Attention. A host address overflow occurs when a single DMA read begins at an address below 4 GB and ends on an address above 4 GB. This is a fatal error.
Read DMA PCI Parity Error Attention Enable	4	RW	0	Enable Read DMA PCI Parity Error Attention
Read DMA PCI Master Abort Attention Enable	3	RW	0	Enable Read DMA PCI Master Abort Attention
Read DMA PCI Target Abort Attention Enable	2	RW	0	Enable Read DMA PCI Target Abort Attention
Enable	1	RW	1	This bit controls whether the Read DMA state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the Read DMA state machine is reset. This is a self-clearing bit.

Real-TimeRead DMA Status Register (offset: 0x4A04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:11	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
MBUF SBD Corruption Attn Status	13	W2C	0	–
MBUF RBD Corruption Attn Status	12	W2C	0	–
BD SBD Corruption Attn Status	11	W2C	0	–
Read DMA PCI-X Split Transaction Timeout Expired	10	W2C	0	Read DMA PCI-X split transaction timeout expired.
Read DMA Local Memory Write Longer Than DMA Length Error	9	W2C	0	Read DMA Local Memory Write Longer Than DMA Length Error.
Read DMA PCI FIFO Overread Error	8	W2C	0	Read DMA PCI FIFO Overread Error (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Error	7	W2C	0	Read DMA PCI FIFO Underrun Error
Read DMA PCI FIFO Overrun Error	6	W2C	0	Read DMA PCI FIFO Overrun Error
Read DMA PCI Host Address Overflow Error	5	W2C	0	Read DMA PCI Host Address Overflow Error. A host address overflow occurs when a single DMA read begins at an address below a multiple of 4 GB and ends at an address above the same multiple of 4 GB (i.e., the host memory address transitions from 0xXXXXXXXX_FFFFFFFF to 0xYYYYYYYY_00000000 in a single read). This is a fatal error.
Read DMA Completion Timer Timeout	4	W2C	0	Read DMA PCIe Completion Timer Timeout
Read DMA Completer Abort	3	W2C	0	Read DMA PCIe Completer Abort
Read DMA Unsupported Request	2	W2C	0	Read DMA PCIe Unsupported Request
Reserved	1:0	RO	0	–

Real-TimeREAD Programmable IPv6 Extension Header Register (offset: 0x4A08)

Name	*Bits*	*Access*	*Default Value*	*Description*
Programmable Extension Header Type #2 Enable	31	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [15:8] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [15:8].

Name	Bits	Access	Default Value	Description
Programmable Extension Header Type #1 Enable	30	R/W	0	This bit enables programmable extension header #1. If this bit is clear, then the value programmed in bits [7:0] of this register will be ignored. If this bit is set, then extension headers will be checked for a type matching the value in bits [7:0].
Reserved	29:16	RO	0	Reserved bits
Programmable Extension Header Type #2	15:8	R/W	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	R/W	0	These bits contain the programmable extension header value for programmable header #1.

Real-TimeREAD Rstates Debug Register (offset: 0x4A0c)

Name	Bits	Access	Default Value	Description
Reserved	31:6	R/W	0	Reserved
Rstate3	5:4	RO	00	00: idle 01: Mbuf Request State 10: Mbuf Ready State
Reserved	3	RO	0	Reserved bits

Rstate1	2:0	RO	0	parameter idle1_st = 3'b 000;
				parameter ftq_b_req_st = 3'b 001;
				parameter ftq_d_req_st = 3'b 010;
				parameter dmad_b_rd_req_st = 3'b 011;
				parameter dmad_d_rd_req_st = 3'b 100;
				parameter dmad_wr_req_st = 3'b 101;
				parameter wait_b_st = 3'b 110;
				parameter wait_d_st = 3'b 111;

Real-TimeREAD Rstate2 Debug Register (offset: 0x4A10)

Name	Bits	Access	Default Value	Description
Reserved	31:5	R/W	0	Reserved

Name	Bits	Access	Default Value	Description

Rstate2	4:0	RO	0	parameter idle2_st = 5'b 00000;
				parameter non_op_st = 5'b 00001;
				parameter wait2_st = 5'b 00010;
				parameter wr_mem_req0_pre_st = 5'b 00011;
				parameter wr_mem_req0_st = 5'b 00100;
				parameter wr_mem_req1_st = 5'b 00101;
				parameter wr_mem_req2_st = 5'b 00110;
				parameter wr_mem_req3_st = 5'b 00111;
				parameter wr_mem_req4_st = 5'b 01000;
				parameter wr_mem_st = 5'b 01001;
				parameter wr_mbuf_hdr_req_st = 5'b 01010;
				parameter wr_mbuf_hdr_st = 5'b 01011;
				parameter mbuf_get_st = 5'b 01100;
				parameter wr_fd_req_st = 5'b 01101;
				parameter wr_fd_st = 5'b 01110;
				parameter pre_wr_ftq_st = 5'b 01111;
				parameter intr_st = 5'b 10000;
				parameter rdr_err_st = 5'b 10001;
				parameter wait_st = 5'b 10010;
				parameter wr_mem_req5_st = 5'b 10011;
				parameter chk_hst_addr_st = 5'b 10100;
				parameter wr_full_mbuf_hdr_req_st = 5'b 10101;
				parameter wr_full_mbuf_done_st = 5'b 10111;
				parameter wr_mis_bd_req_st = 5'b 11000;
				parameter wr_mem_req_st = 5'b 11001;
				parameter wr_mbuf0_req_st = 5'b 11010;
				parameter wait4_proc_done_st = 5'b 11011;
				parameter wr_mbuf1_req_st = 5'b 11100;
				parameter get_more_mbuf_st = 5'b 11101;

Real-TimeREAD BD Status Debug Register (offset: 0x4A14)

Name	Bits	Access	Default Value	Description
Reserved	31:3	R/W	0	Reserved
Bd_non_Mbuf	2	RO	0	1: BD Request 0: Mbuf Request
Fst_bd_mbuf	1	RO	0	1: First BD_MBuf Request 0: Not First_BD_Mbuf_Request
Lst_bd_mbuf	0	RO	0	1: Last BD_MBuf Request 0: Not Last_BD_Mbuf_Request

Real-TimeREAD Request Pointer Debug Register (offset: 0x4A18)

Name	*Bits*	*Access*	*Default Value*	*Description*
ih_dmad_length	31:16	R/W	0	Dmad Length request size
Reserved	15:10	RO	0	Reserved
Txmbuf_left	9:4	RO	0	Amount of TXMBuf Left
Rh_dmad_load_en	3	RO	0	State of rh_dmad_load_en signal
Rftq_d_dmad_pnt	2:1	RO	00	FTQ Data Dmad Pointer
Rftq_b_dmad_pnt	0	RO	0	Ftq BD DMAD Pointer

Real-TimeREAD hold_d_dmad Debug Register (offset: 0x4A1c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:2	RO	0	Reserved
Rhold_d_dmad	1:0	RO	0	Number of outstanding data dmad request

Real-TimeREAD Length and Address Index Debug Register (offset: 0x4A20)

Name	*Bits*	*Access*	*Default Value*	*Description*
Rdma_rd_length	31:16	R/W	0	Dmad Length request size
Mbuf_addr_idx	15:0	RO	0	6-bit allocation tx mbuf index; upper bits are zeroes

Real-TimeREAD Mbuf Byte Count Debug Register (offset: 0x4A24)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:4	RO	0	Reserved
Rmbuf_byte_cnt	3:0	RO	0	Mbuf Byte Count

Real-TimeREAD PCIe Mbuf Byte Count Debug Register (offset: 0x4A28)

Name	Bits	Access	Default Value	Description
Lt_term	31:28	RO	0	Latched Termination Code
Reserved	27	RO	0	Reserved
Lt_too_lg	26	RO	0	DMA Too Large Error
Lt_dma_reload	25	RO	0	State of dmad Load signal
Lt_dma_good	24	–	–	State of dmad good signal
Cur_trans_active	23	–	–	State of cur_trans_active signal
DRPCIREQ	22	–	–	State of DR PCI Request signal
Dr_pci_word_swap	21	–	–	State of dr_pci_word_swap signal
Dr_pci_byte_swap	20	–	–	State of dr_pci_byte_swap signal
New_slow_core_clk_mode	19	–	–	State of gated new_slow_core_clock mode signal which is a function of the following input: new_slow_core_clk_mode = SLOW_CORE_CLK_MODE | slow_down_request_syn2 | rbd_non_mbuf and cq25155_fix_enable;
Rbd_non_mbuf	18	–	–	State of bd_non_mbuf signal
Rfst_bd_mbuf	17	–	–	State of first_bd_Mbuf signal
Rlst_bd_mbuf	16	–	–	State of last BD_Mbuf Signal
Dr_pci_len	15:0	–	–	Indicates the amount of DMA read request

Real-TimeREAD PCIe DMA Read Request Address Debug Register (offset: 0x4A2c)

Name	Bits	Access	Default Value	Description
Dr_pci_ad31:16	31:16	RO	0	Upper 16-bit of DMA read request address
Dr_pci_ad	15:0	RO	0	Lower 16-bit of DMA read request address

Real-TimeREAD PCIe DMA Request Length Debug Register (offset: 0x4A30)

Name	Bits	Access	Default Value	Description
Rdma_len	31:0	RO	0	DMA Read Request Length to Host

Real-TimeREAD FIFO1 Debug Register (offset: 0x4A34)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0	Upper 16-bit of DMA read request address
C_write_addr	7:0	RO	0	Core Clock Write Address Pointer

Real-TimeREAD FIFO2 Debug Register (offset: 0x4A38)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	Upper 16-bit of DMA read request address
Rlctrl_in	15:8	RO	0	Amount of available free space in the DMA Read FIFO: Write Pointer – Read Pointer
C_read_addr	7:0	RO	0	Core Clock Read Address Pointer

Real-TimeREAD Packet Request Debug Register (offset: 0x4A40)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0	Reserved
Pkt_req_cnt(7:0)	7:0	RO	0	Request Counts

Real-TimeREAD Packet Request Debug Register (offset: 0x4A44)

Name	*Bits*	*Access*	*Default Value*	*Description*
Sdc_ack_cnt	31:0	RO	0	Send Data Completion Ack. Counts

Real-TimeREAD Packet Request Debug Register (offset: 0x4A48)

Name	*Bits*	*Access*	*Default Value*	*Description*

Cs	31:28	RO	0	Parameter IDLE = 4'b 0000;
				Parameter MEM_RD_REQ0 = 4'b 0011;
				Parameter MEM_RD_REQ1 = 4'b 0100;
				Parameter MEM_WR_REQ0 = 4'b 0101;
				Parameter MEM_WR_REQ1 = 4'b 0110;
				Parameter WR_MBUF0 = 4'b 0111;
				Parameter WR_MBUF1 = 4'b 1000;
				Parameter FD_REQ = 4'b 1001;
				Parameter FD_RD = 4'b 1010;
				Parameter SDC_WR = 4'b 1011;
				Parameter CHK_FD_FLAGS = 4'b 1100;

Reserved	27:26	RO	0	Reserved
Lt_fst_seg	25	RO	0	Indicates First Segment
Lt_lst_seg	24	RO	0	Indicates Last Segment
Lt_mem_ip_hdr_ofst	23:16	RO	0	IP Header Offset
Lt_mem_tcp_hdr_ofst	15:8	RO	0	TCP Header Offset
Reserved	7	RO	0	Reserved
Pre_sdcq_pkt_cnt	6:0	RO	0	Pre-Send Data Completion Queue Packet Count

Real-TimeREAD TCP Checksum Debug Register (offset: 0x4A4c)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Reserved
Fd_addr_req(9:4)	29:24	RO	0	Holding the content of Frame Descriptor Address Register
Lt_mem_data_ofst	23:16	RO	0	Indicates First Segment
Lt_mem_tcp_chksum	15:0	RO	0	Hold the current value of the Tcp_checksum calculation

Real-TimeREAD Ip/tcp Header Checksum Debug Register (offset: 0x4A50)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_mem_ip_chksum	31:16	RO	0	Hold the current value of the ip Checksum Calculation
Lt_mem_tcphdr_chksum	15:0	RO	0	Hold the current value of the Tcp Header_checksum calculation

Real-TimeREAD Pseudo Checksum Debug Register (offset: 0x4A54)

Name	*Bits*	*Access*	*Default Value*	*Description*
Lt_mem_pse_chksum_no_tcplen	31:16	RO	0	Hold the current value of the Pseudo-Checksum Calculation
Lt_mem_pkt_len	15:0	RO	0	Hold the current value of the packet length

Real-TimeREAD Mbuf Address Debug Register (offset: 0x4A58)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	Reserved
Mbuf1_addr(9:4)	29:24	RO	0	Mbuf1 Address Pointer
Reserved	23:22	RO	0	Reserved
Mbuf0_addr(9:4)	21:16	RO	0	Mbuf0 Address Pointer
Reserved	15:14	RO	0	Reserved
Pre_Mbuf1_addr(9:4)	13:8	RO	0	Pre_Mbuf1 Address Pointer
Reserved	7:6	RO	0	Reserved
Pre_Mbuf0_addr(9:4)	5:0	RO	0	Pre_Mbuf0 Address Pointer

Real-TimeREAD Miscellaneous Control1 Register- Debug Controls (offset: 0x4B00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Txmbuf_margin	31:21	R/W	0	Fix for CQ27862
Select fed Enable	20	RW	0	Ensure only 1 request is generated upon any condition where the core clock is switching from slow to fast or vice-versa. ***Note:*** In some controllers this field is mapped to bit-0 of this register.
FIFO High Mark	19:12	RW	0x4A	–
FIFO Low Mark	11:4	RW	0x20	–
Slow Clock Fix Disable	3	RW	0	When cleared, it enables the fix to cover a corner case in the link idle mode to allow the DMA Read request to be generated when the core clock is transitioning from slow to fast

Name	Bits	Access	Default Value	Description
Enable hardware fix 25155 CQ25155 fix enable	2	RW	0	When set, enables hardware fix 25155 When set, this bit enables the fix for CQ25155, where a DMA FIFO overrun occurs if a large number of RX BDs are fetched while the TX MBUF is full and the Real-TimeREAD FIFO is empty.
Late Collision Fix Enable	1	RW	1	–
SDI ShortQ Enable	0	RW	1	When set, this bit enables hardware fix 27862 When set, this bit enables the fix for CQ27862

Real-TimeREAD Miscellaneous Control2 Register (offset: 0x4B04)

)Name	Bits	Access	Default Value	Description
Fifo_threshold_bd_req	31:24	RW	0	This register contains various controls to configure hardware fix 27862. This register contains various controls to fix CQ27862.
Fifo_threshold_mbuf_req	23:16	RW	0x20	–
Reserved	15:14	RW	0	–
MBUF Threshold MBUF Request	13:8	RW	0xe	–
Reserved	7	RW	0	–
Clock Request Fix Enable	6	RW	0	–
MBUF Threshold Clk Req	5:0	RW	0x8	–

Real-TimeREAD Miscellaneous Control3 Register – Debug Control (offset: 0x4B10)

)Name	Bits	Access	Default Value	Description
Reserved	31:6	RW	0	–
CQ33951_Fix_Disable	5	RW	0	This control bit is used to enable the fix for CQ33951 where the IP Header Checksum Corruption occurs when an IPV4 payload contents match an IPV6 Header Type 1: Disable fix 0: enable fix

)Name	Bits	Access	Default Value	Description
Enable hardware fix 30888 CQ30888 Fix Enable	4	RW	1	Set to 1 to enable fix for clock request gap problem of TX Real-TimeREAD lock-up issue, CQ Cont00030888 ***Note:*** Increasing the ASPM L1 entry time to a value on the order of 1ms is recommended and may prevent this issue from occurring. See register 0x7d28.
Enable hardware fix 30888 CQ30888 Fix Enable	3	RW	1	Set to 1 to enable fix for clock switching problem of TX Real-TimeREAD lock-up issue, CQ Cont00030888 ***Note:*** Increasing the ASPM L1 entry time to a value on the order of 1ms is recommended and may prevent this issue from occurring. See register 0x7d28.
Enable hardware fix 30808 CQ30808 Fix Enable	2	RW	0	Set to 1 to enable fix for TX Real-TimeREAD lock-up issue, CQ Cont00030808 ***Note:*** Increasing the ASPM L1 entry time to a value on the order of 1ms is recommended and may prevent this issue from occurring. See register 0x7d28.
Reserved	1	RW	0	Reserved
Reserved	0	RW	0	Reserved

WDMA Registers

All registers reset are core reset unless specified.

Write DMA Mode Register (offset: 0x4C00)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	–
Status Tag (CQ12384) Fix Enable	29	RW	1	1: Enable CQ12384 fix. Device will send out Status Block before the interrupt message. 0: Disable CQ12384 fix. Device will send out the interrupt message before Status Block. This issue is worked around in the Broadcom driver so the default value is 0. ***Note:*** For the BCM5725/BCM5762/BCM57767-PG106, there is no need to touch this bit in the driver; keep the default value.
Reserved	28:19	RO	0	–
Swap Test Enable	18	RW	0	When this bit is set, swap test mode will be enabled and bits 17 to 12 can be used to test different byte/word swap settings.
HC Byte Swap	17	RW	0	Byte swap control for status words.
HC Word Swap	16	RW	0	Word swap control for status words.

Name	*Bits*	*Access*	*Default Value*	*Description*
BD Byte Swap	15	RW	0	Byte swap control for return BDs
BD Word Swap	14	RW	0	Word swap control for return BDs
Data Byte Swap	13	RW	0	Byte swap control for data
Data Word Swap	12	RW	0	Word swap control for data
Software Byte Swap Control	11	RW	0	To override byte enables with all 1's
Receive Accelerate Mode	10	RW	0	The write DMA-to-PCI request length is the available data size in the PCI RX FIFO. Set to 1: The write DMA-to-PCI request length is the maximum length of the current transaction, regardless of the available data size in PCI RX FIFO. This mode cannot be used in slow core clock environment. Disable this mode before switching to slow core clock mode.
Write DMA Local Memory	9	RW	0	Attention Enable. Enable Write DMA Local Memory Read Longer Than DMA Length Attention.
Write DMA PCI FIFO Overwrite Attention Enable	8	RW	0	Enable Write DMA PCI FIFO Overwrite Attention (PCI write longer than DMA length).
Write DMA PCI FIFO Underrun Attention Enable	7	RW	0	Enable Write DMA PCI FIFO Underrun Attention.
Write DMA PCI FIFO Overrun Attention Enable	6	RW	0	Enable Write DMA PCI FIFO Overrun Attention.
Write DMA PCI Host Address Overflow Error Attention Enable	5	RW	0	Enable Write DMA PCI Host Address Overflow Error Attention.
Write DMA PCI Parity Error Attention Enable	4	RW	0	Enable Write DMA PCI Parity Error Attention.
Write DMA PCI Master Abort Attention Enable	3	RW	0	Enable Write DMA PCI Master Abort Attention.
Write DMA PCI Target Abort Attention Enable	2	RW	0	Enable Write DMA PCI Target Abort Attention.
Enable	1	RW	1	This bit controls whether the Write DMA state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains 1 when read.
Reset	0	RW	0	When this bit is set to 1, the Write DMA state machine is reset. This is a self-clearing bit.

Write DMA Status Register (offset: 0x4C04)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
Write DMA Local Memory Read Longer than DMA Length Error	9	W2C	0	Write DMA Local Memory Read Longer Than DMA Length Error
Write DMA PCI FIFO Overwrite Error	8	W2C	0	Write DMA PCI FIFO Overwrite Error (PCI write longer than DMA length).
Write DMA PCI FIFO Underrun Error	7	W2C	0	Write DMA PCI FIFO Underrun Error.
Write DMA PCI FIFO Overrun Error	6	W2C	0	Write DMA PCI FIFO Overrun Error.
Write DMA PCI Host Address Overflow Error	5	W2C	0	Write DMA PCI Host Address Overflow Error.
Reserved	4	W2C	0	Reserved
Reserved	3	W2C	0	Reserved
Reserved	2	W2C	0	Reserved
Reserved	1:0	RO	0	–

RX-CPU Registers

All registers reset are core reset unless specified.

RX RISC Mode Register (offset: 0x5000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:15	RO	0	–
Enable register address trap halt	14	RW	0	When set, if the GRC raises the trap signal to this processor, it will halt. Cleared on reset and Watchdog interrupt.
Enable memory address trap halt	13	RW	0	When set, if the MA raises the trap signal to this processor, it will halt. Cleared on reset and Watchdog interrupt.
Enable Invalid Instruction Fetch halt	12	RW	0	When set, the condition that causes RX RISC state bit 6 to be set, also halts the RX RISC. Set by reset. Cleared by Watchdog interrupt.
Enable Invalid Data access halt	11	RW	0	When set, the condition that causes RX RISC state bit 5 to be set, also halts the RX RISC. Set by reset. Cleared by Watchdog interrupt.

Name	*Bits*	*Access*	*Default Value*	*Description*
Halt RX RISC	10	RW	0	Set by TX RISC or the host to halt the RX RISC.Cleared on reset and Watchdog interrupt.
Reserved	9	WO	0	Reserved
Reserved	8	RO	0	Reserved
Enable Watchdog Enable Interrupt	7	RW	0	Enables watchdog interrupt state machine. Used in conjunction with Watchdog Clear register, Watchdog Saved PC register and Watchdog Vector register. Cleared on reset and Watchdog interrupt. When this bit is set to 1, the interrupt is enabled. When this bit is zero, any interrupt will be ignored. This bit can also be set by writing the interrupt_enable register
ROM Fail	6	RW	1	Asserted on reset. Cleared by ROM code after it successfully loads code from NVRAM. Afterwards, this bit can be used by software for any purpose.
Reserved	5	RO	0	Reserved
Reserved	4	RO	0	Reserved
Enable Page 0 Instr Halt	3	RW	0	When set, instruction references to the first 256 bytes of SRAM force the RX RISC to halt and cause bit 4 in the RX RISC state register to be latched. Cleared on reset and Watchdog interrupt.
Enable Page 0 Data Halt	2	RW	0	When set, data references to the first 256 bytes of SRAM force the RX RISC to halt and cause bit 3 in the RX RISC state register to be latched. Cleared on reset and Watchdog interrupt.
Single-Step RX RISC	1	RW	0	Advances the RX RISC's PC for one cycle. If halting condition still exists, the RX RISC will again halt; otherwise, it will resume normal operation.
Reset RX RISC	0	WO	0	Self-clearing bit which resets only the RX RISC.

RX RISC Status Register (offset: 0x5004)

The RX RISC State register reports the current state of the RX RISC and, if halted, gives reasons for the halt. There are four categories of information; informational (read-only), informational (write-to-clear), disable-able halt conditions (write-to-clear), and nondisable-able halt conditions (write-to-clear).

Name	*Bits*	*Access*	*Default Value*	*Description*
Blocking Read	31	RO	0	A blocking data cache miss occurred, causing the RX RISC to stall while data is fetched from external (to the RX RISC) memory. This is intended as a debugging tool. No state is saved other than the fact that the miss occurred.
MA Request FIFO overflow	30	W2C	0	MA_req_FIFO overflowed. The RX RISC is halted on this condition.

Name	*Bits*	*Access*	*Default Value*	*Description*
MA data/bytemask FIFO overflow	29	W2C	0	MA_datamask_FIFO overflowed. The RX RISC is halted on this condition.
MA outstanding read FIFO overflow	28	W2C	0	MA_rd_FIFO overflowed. The RX RISC is halted on this condition.
MA outstanding write FIFO overflow	27	W2C	0	MA_wr_FIFO overflowed. The RX RISC is halted on this condition.
Reserved	26:16	RO	0	–
Instruction fetch stall		RO	0	The processor is currently stalled due to an instruction fetch.
Data access stall		RO	0	The processor is currently stalled due to a data access.
Reserved	13	RO	0	
Interrupt Received	12	RO	0	This bit is each time an interrupt input is asserted, regardless of the interrupt enable bit (bit 7, mode).
Reserved	11	RO	0	–
RX RISC Halted	10	RO	0	The RX RISC was explicitly halted via bit 10 in the RX RISC Mode register.
Register Address Trap	9	W2C	0	A signal was received from the Global Resources block indicating that this processor accessed a register location that triggered a software trap. The GRC registers are used to configure register address trapping.
Memory Address Trap	8	W2C	0	A signal was received from the Memory Arbiter indicating that some BCM5700 block, possibly this processor, accessed a memory location that triggered a software trap. The MA registers are used to configure memory address trapping.
Bad Memory Alignment	7	W2C	0	Load or Store instruction was executed with the least significant two address bits not valid for the width of the operation (e.g., Load word or Load Half-word from an odd byte address).
Invalid Instruction Fetch	6	W2C	0	Program Counter (PC) is set to invalid location in processor address space.
Invalid Data Access	5	W2C	0	Data reference to illegal location.
Page 0 Instruction Reference	4	W2C	0	When enabled in mode register, indicates the address in the PC is within the lower 256 bytes of SRAM.
Page 0 Data Reference	3	W2C	0	When enabled in mode register, indicates data reference within lower 256 bytes of SRAM.
Invalid Instruction	2	W2C	0	Invalid instruction fetched.
Halt Instruction Executed	1	W2C	0	A halt-type instruction was executed by the RX RISC.

Name	*Bits*	*Access*	*Default Value*	*Description*
Hardware Breakpoint	0	W2C	0	When enabled (in Hardware Breakpoint Register), indicates hardware breakpoint has been reached. One of the 3 actions below will clear this bit 1) Disable breakpoint 2) Reprogram the breakpoint addr, or 3) Reprogram the PC

Event Mask Register (offset: 0x5008)

This register provides one bit for each state register bit to enable it into the equation for generation the TX Processor Attention output. The reset value of 1 masks all halt conditions from generating an attention.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	13	RO	0	–
Interrupt mask	12	RW	0	This bit attention when bit 12 of the state register is set
Spad underflow mask	11	RW	0	The processor is currently stalled due to a data access.
Soft halted mask	10	RW	1	This bit enables soft halts to generate Attention output
Reserved	9	RW	0	–
FIO Abort mask	8	RW	1	This bit enables the attention output when bit 8 of the state register is set.
Align halted mask	7	RW	0	This bit enables alignment errors to generate Attention output
Bad pc halted mask	6	RW	0	This bit enables invalid PC values to generate Attention output
Bad data addr halted mask	5	RW	0	This bit enables invalid data addresses to generate Attention output.
Page 0 inst halted mask	4	RW	0	This bit enables page 0 instructions to generate Attention output
Page 0 data halted mask	3	RW	0	This bit enables page 0 data access to generate Attention output
Bad inst halted mask	2	RW	0	This bit enables invalid instruction decodes to generate Attention output
Reserved	1	RW	0	–
Breakpoint mask	0	RW	0	This bit enables breakpoints to generate Attention output.

Instruction Register (offset: 0x5020)

This register allows access instruction in the decode sate of the pipeline while the processor is halted. This register is only intended for debugging use. This register may be used to replace a halt instruction with some other instruction after the halt has been executed.

Interrupt Enable Register (offset: 0x5028)

Any write to this register will enable CPU Interrupts (set bit 7 in mode register). This register is intended to allow a way to return from an interrupt service routine (ISR) using only 2 general purpose registers. MIPS conventions reserve registers 26 and 27 (k0 and k1) for use by an interrupt handler. At the end of an ISR, k0 should be loaded with the return address from the CPU Interrupt Saved PC register. Then k1 should be loaded with the address of the CPU Interrupt Enable register. The last 2 instructions in the ISR should be a jump register (jr) to k0 followed immediately by a store word (sw) to k1. This ensures that we can't respond to another interrupt until we are safely out of the ISR. Interrupts can also be enabled through the CPU Mode Register. They can be disabled only through the CPU Mode Register. Each time this register is written, bit 7 of the **mode** register is set. The data value of the write is not used. The read value of this register is always zero.

Interrupt Vector Register (offset: 0x502C)

This register sets the program counter value that will be loaded when an interrupt is performed due to the interrupt input.

Interrupt Saved PC Register (offset: 0x5030)

This register reports the PC that was saved during the execution of an interrupt.

RX RISC Hardware Breakpoint Register (offset: 0x5034)

This register is used to set a hardware breakpoint based on the RISC's program counter (PC). If the PC equals the value in this register, and the hardware breakpoint is enabled, the RISC is halted and the appropriate stopping condition is indicated in the RISC State Register. To enable the hardware breakpoint, simply write the byte address of the instruction to break on and clear the Disable Hardware Breakpoint bit.

Name	*Bits*	*Access*	*Default Value*	*Description*
Hardware Breakpoint	31:2	RW	0	Word address to break on
Reserved	1	RO	0	–
Disable Hardware Breakpoint	0	RW	1	When this bit is set, the Hardware Breakpoint is disabled

Name	*Bits*	*Access*	*Default Value*	*Description*
Progress Code	–	–	–	Self-Boot Progress Code #define PROGRESS_CODE_STARTED 0x41 #define PROGRESS_CODE_ICP_INIT 0x45 #define PROGRESS_CODE_DONE_INIT 0x49 #define PROGRESS_CODE_SCP_INIT 0x4d #define PROGRESS_CODE_SERVICE 0x51 #define PROGRESS_CODE_NO_DATA 0x55 ROM Code Progress Code #define Reading Media Region 0x5 #define SEPROM Not Found 0x9 #define BAD Magic Value 0xd #define ROM CRC Media Region Failed 0x11 #define Reading Code 0x15 #define CRC Code Fail 0x19 #define CPU Running 0x1d #define Fast Boot 0x25 #define Self-Boot 0x29

Last Branch Address Register (offset: 0x5048)

This register indicates that address and branch type of the last branch that was taken. This register is for debug use only.

Name	*Bits*	*Access*	*Default Value*	*Description*
LBA	31:2	RO	0	This value indicates the address of the last branch that was taken. An offset as indicated by the type field must be subtracted from this value. This bit indicates the type of branch that * was last taken.
TYPE	1	RO	0	0x0: Jump: 4 bytes must be subtracted from the LBA value to determine the actual address of the branch instruction that caused this register to load. 0x1: Branch: 8 bytes must be subtracted from the LBA value to determine the actual address of the branch instruction that caused this register to load.
Reserved	0	RO	0	When this bit is set, the Hardware Breakpoint is disabled

EAV Filter Set Registers

There are 8 Filtering Elements and 8 Filtering Sets - two registers per Filtering Element and two registers per Filtering Set, in total 32.

FILT_ELEM_CFG[0:7] Register (offset: 0x5400-0x541C)

Each of these registers configures a Management Filtering Element. There are 8 such elements. The lowest register address offset associates with Element#0 and the highest address offset associates with Element#7.

Name	***Bits***	***Access***	***Default Value***	***Description***
Enable	31	R/W	0	1 == This element is enabled for use in Filtering Set-equations 0 = This element shall return a FALSE value if called into a Set-equation
Unused	30:25	RU	0xUU	Unpredictable read value.
Mask Mode	26	R/W	0	1 == Apply a 16-bit mask on 16-bits extracted from a packet and then operate with the 16-bit Element-Pattern 0 ==Apply the 32-bit Element-Pattern on 32-bits extracted from a packet See FILT_ELEM_PTTRN_REG ***Note:*** When [15:13] is set to **111**, the value of this bit is ignored and assumed to be a 1 by h/w.
Unused	25:18	RU	0xU	Unpredictable read value.
Element-Opcode	17:16	R/W	000	Comparison Operation to be performed on 32 or 16 bits extracted from "Element-Offset" relative to the "Element-Header"- 00: Equal 01: Not Equal 10: Greater Than 11: Less Than
Element-Header	15:13	R/W	000	H/w must detect this Header type in a packet before applying the filtering pattern: 000: Start of Frame 001: Start of IPv4/IPv6 Header 010: Start of TCP Header 011: Start of UDP Header 100: Reserved 101: Start of ICMPv4 Header 110: Start of ICMPv6 Header 111: Start of a VLAN TAG Control Field
Unused	12:8	RU	0xU	Unpredictable read value.

Name	*Bits*	*Access*	*Default Value*	*Description*
Element-Offset	7:0	R/W	0x00	Stride length, in number of bytes, relative to the first octet of the specified "Element-Header". ***Note:*** When [15:13] is set to **111**, the value of this field is ignored and assumed to be 0 by h/w.

FILT_ELEM_PTTRN[0:7] Register (offset: 0x5480-0x549C)

Each of these registers supplies a 32-bit Element-Pattern or a 16-bit Element-Pattern plus a 16-bit Element-Mask corresponding to each FILT_ELEM_CFG_REG. There are 8 of these registers. The lowest register address offset associates with Element#0 and the highest address offset associates with Element#7.

Name	*Bits*	*Access*	*Default Value*	*Description*
Element-Pattern Right	31:16	R/W	0x0000	This field supplies the left most 16-bits to be compared with 16-bits extracted from a packet. This value is in Network-Byte-Order.
Element-Mask or Element-Pattern Left	15:0	R/W	0x0000	This value is in Network-Byte-Order. If "Mask Mode" == 1 : This field is used as a mask to be applied on 16-bits extracted from a packet. The masked value is then compared with the field "Element-Pattern Left" If "Mask == 0" : This field supplies the right most 16-bits to be compared with 16-bits extracted from a packet.

Reserved [0-7] Register (offset: 0x54A0-0x54BC)

There are no equivalents of Filtering Set Configuration Registers in this chip – the Set Actions are nonexistent, the Sets produce a Boolean True/False result.

FILT_SET_MSK_REG[0:7] Register (offset: 0x5500-0x551C)

Filtering Set Mask Registers. Each of these registers defines a Set-Equation. There are 8 Filtering Sets and hence 8 of these registers. The lowest register address offset associates with Set#0 and the highest address offset associates with Set#7.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0x0000	–

Name	***Bits***	***Access***	***Default Value***	***Description***
Set-Mask	7:0	RW	0x00	This 8-bit field determines which Filtering-Element shall participate in this Set equation. If a bit ==1, the corresponding Element participates. The Boolean results of all participating Elements are AND-ed to produce the final Boolean result of the Set.

Perfect Match Registers

Perfect_MATCH[4 – 5]_HIGH_REG (Offsets 0x5690, 0x5698)

There are total 4 Perfect (Destination Address) Match registers dedicated to RX-MAC. These registers hold the higher 2 octets of the matching address.

Name	***Bits***	***Access***	***Default Value***	***Description***
Reserved	31:29	RO	000	Reserved
MAC High Address	15:0	RW	0x0000	Upper 2-bytes [4 – 5]th perfect match address

Perfect_MATCH[4 – 5]_LOW_REG (Offsets 0x5694, 0x569C)

There are total 4 Perfect (Destination Address) Match registers dedicated to RX-MAC. These registers hold the lower 4 octets of the matching address.

Name	***Bits***	***Access***	***Default Value***	***Description***
MAC Low Address	31:0	RW	0x0000	Lower 4-bytes of [4 – 5]th perfect match address

Low Priority Mailboxes

This is a 512 byte region that contains 64 registers. These registers are called low-priority mailbox registers (or low-priority mailboxes). When a value is stored in the least significant 32 bits of these registers, an event (known as a Mailbox Event) is generated to the RX RISC.

Interrupt Mailbox 0 Register (offset: 0x5800)

This register is same as BCM5784M.

Other Interrupt Mailbox Register (offset: 0x5808 – 0x5818)

This register is same as BCM5784M.

General Mailbox Registers 1-8 (offset: 0x5820 – 0x5858)

This register is same as BCM5784M.

Receive BD Standard Producer Ring Index Register (offset: 0x5868)

This register is same as BCM5784M.

Receive BD Return Ring 0 Consumer Index (Low Priority Mailbox) Register (offset: 0x5880-0x5887)

This register is same as BCM5784M.

Receive BD Return Ring 1 Consumer Index (Low Priority Mailbox) Register (offset: 0x5888-0x588F)

The Receive BD Return Ring 1 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 1 that has been consumed. Host software writes this register whenever it updates the return ring 1. This register must be initialized to 0.

Receive BD Return Ring 2 Consumer Index (Low Priority Mailbox) Register (offset: 0x5890-0x5897)

The Receive BD Return Ring 2 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 2 that has been consumed. Host software writes this register whenever it updates the return ring 2. This register must be initialized to 0.

Receive BD Return Ring 3 Consumer Index (Low Priority Mailbox) Register (offset: 0x5898-0x589F)

The Receive BD Return Ring 3 Consumer Index Register contains the index of the last buffer descriptor for Receive Return Ring 3 that has been consumed. Host software writes this register whenever it updates the return ring 3. This register must be initialized to 0.

Send BD Ring Producer Index (Low Priority Mailbox) Register (offset: 0x5900)

This register is same as BCM5784M.

Flow Through Queues

All registers reset are core reset unless specified.

FTQ Reset Register (offset: 0x5C00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:17	RO	0	–
Reset Receive Data Completion FTQ	16	RW	0	Set this bit to reset the Receive Data Completion flow through queue. When set to 0, this FTQ is ready for use. This bit is self-clearing
Reserved	15	RO	0	–
Reset Receive List Placement FTQ	14	RW	0	Set this bit to reset the Receive List. This bit self-clearing placement flow through queue. When set to 0, this FTQ is ready for use. This bit is self-clearing
Reset Receive BD Complete FTQ	13	RW	0	Set this bit to reset the Receive BD Complete flow through queue. When set to 0, this FTQ is ready for use. This bit is self-clearing
Reserved	12	RO	0	–
Reset MAC TX FTQ	11	RW	0	Set this bit to reset the MAC TX flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reset Host Coalescing FTQ	10	RW	0	Set this bit to reset the Host Coalescing flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reset Send Data Completion FTQ	9	RW	0	Set this bit to reset the Send Data Completion flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reserved	8	RO	0	–
Reset DMA High Priority Write FTQ	7	RW	0	Set this bit to reset the DMA High Priority Write flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reset DMA Write FTQ	6	RW	0	Set this bit to reset the DMA Write flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reserved	5	RO	0	–
Reset Send BD Completion FTQ	4	RW	0	Set this bit to reset the Send BD Completion flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reserved	3	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Reset DMA High Priority Read FTQ	2	RW	0	Set this bit to reset the DMA High Priority Read flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reset DMA Read Queue FTQ	1	RW	0	Set this bit to reset the DMA Read Queue flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reserved	0	RO	0	–

EAV: Real-TimeFTQ Reset Register (offset: 0x5E00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:12	RO	0	–
Reset MAC TX FTQ	11	RW	0	Set this bit to reset the MAC TX flow through queue. When set to 0, this flow through queue is ready for use. This bit is self-clearing.
Reserved	10:0	RO	0	–

MAC TX FIFO Enqueue Register (offset: 0x5CB8)

A write to this register will add a transmit packet to the tail of the MACTQ FTQ. The host CPU uses this register to send an ASF message out.

Since the size of TXMBUF FIFO is only 64 entries and MACTQ is 12 bits wide:

- Bits 21:16 from this register are mapped to bits 11:6 of the MACTQ FTQ.
- Bits 5:0 from this register are mapped to bits 5:0 of the MACTQ FTQ.
- Bits 31:22 and 15:6 are ignored.

The TXMBUF cluster for the ASF message is defaulted to the uppermost three TXMBUFs.

Name	*Bits*	*Access*	*Default Value*	*Description*
Head TXMBUF Pointer	31:16	WO	003Dh	Specifies the first MBUF of the TXMBUF cluster for the transmit packet.
Tail TXMBUF Pointer	15:0	WO	003Fh	Specifies the last MBUF of the TXMBUF cluster for the transmit packet.

EAV: Real-TimeMAC TX FIFO Enqueue Register (offset: 0x5DB8)

A write to this register will add a transmit packet to the tail of the MACTQ FTQ. The host CPU uses this register to send an ASF message out.

Since the size of TXMBUF FIFO is only 64 entries and MACTQ is 12 bits wide:

- Bits 21:16 from this register are mapped to bits 11:6 of the MACTQ FTQ.
- Bits 5:0 from this register are mapped to bits 5:0 of the MACTQ FTQ.
- Bits 31:22 and 15:6 are ignored.

The TXMBUF cluster for the ASF message is defaulted to the uppermost three TXMBUFs.

Name	*Bits*	*Access*	*Default Value*	*Description*
Head TXMBUF Pointer	31:16	WO	003Dh	Specifies the first MBUF of the TXMBUF cluster for the transmit packet.
Tail TXMBUF Pointer	15:0	WO	003Fh	Specifies the last MBUF of the TXMBUF cluster for the transmit packet.

RXMBUF Cluster Free Enqueue Register (offset: 0x5CC8)

A write to this register will free a cluster of RXMBUFs. The host CPU uses this register to deallocate RXMBUFs after it has processed the received ASF message.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:18	RO	0	–
Head RXMBUF Pointer	17:9	WO	00h	Specifies the first MBUF of the RXMBUF cluster for the received packet to be freed.
Tail RXMBUF Pointer	8:0	WO	00h	Specifies the last MBUF of the TXMBUF cluster for the received packet to be freed.

RDIQ FTQ Write/Peak Register (offset: 0x5CFC)

The host CPU uses this register to get the RXMBUF cluster pointers if the received packet requires the attention of the This could be an ASF or ACPI packet.

- A write to this register will modify the head of the RDIQ FTQ entry.
- A read of this register will peek at the head of the RDIQ FTQ entry.
- When the Valid bit is 1 and the Pass bit is 0, the CPU can take the RXMBUF cluster pointers to access the received Packet.
- When the CPU writes a 1 to the Skip bit, the hardware will pop the head of the queue entry.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
Valid Bit	20	RW	0	Set only if the head of RDIQ entry is valid
Skip Bit	19	RW	0	If this bit iset, the head of RDIQ entry will be popped
Pass Bit	18	RW	0	This bit is 0 if RDIQ head entry is intended for the CPU. It prevents the entry to be serviced by WDMA
Head RXMBUF Pointer	17:9	RO	0	Specifies the first MBUF of the RXMBUF cluster for the received packet
Tail RXMBUF Pointer	8:0	RO	0	Specifies the last MBUF of the RXMBUF cluster for the received packet

Message Signaled Interrupt Registers

All registers reset are core reset unless specified.

MSI Mode Register (offset: 0x6000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Priority	31:30	RW	0	Sets the priority of the MSI engine relative to the DMA read engine and DMA Write engine. Equal settings result in fair round robin arbitration. 00: Lowest 01: Low 10: High 11: Highest
Reserved	29:24	RO	0	–
MSI ST Lower	23:16	RW	0x01	This 8-bit value is used as the TPH Steering Tag when TPH Interrupt Vector Mode is enabled and MSI Mode is enabled.
Reserved	28:11	RO	0	–
MSI Message	10:8	RW	0	This register sets the MSI message data bottom bits to the value programmed here. This register exists only for testing purposes and should always be programmed to zero.
MSIX Multi-Vector Mode	7	RW	0	Enable MSIX Multi-Vector Mode 0: Disable 1 : Enable
MSI Byte Swap Enable	6	RW	0	Enable MSI byte swap 0: Disable 1: Enable
MSI Single Shot Mode Disable	5	RW	0	Disable MSI Single Shot Mode 1: Disable One-Shot MSI mode 0;: Enable One-Shot MSI mode One-Shot MSI mode works in conjunction with Tagged Status mode. The Tagged Status mode is enabled by setting bit 9 of Register 0x68. When One-Shot MSI mode is enabled, the HW automatically auto-mask and auto-acknowledge when MSI interrupt is generated by the HW. SW needs to unmask the interrupt after it has serviced the current MSI interrupt to allow future interrupts to be generated by the HW. The mechanism for SW to unmask the interrupt is by writing to the mailbox register bits 31:24 with the same tag value that was DMAed to the host in the last status block update.
PCI Parity Error Attn	4	RW	0	PCI parity error attention enable.

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Master Abort Attn	3	RW	0	PCI master abort attention enable.
PCI Target Abort Attn	2	RW	0	PCI target abort attention enable.
Enable	1	RW	1	This bit controls whether the MSI state machine is active or not. When set to 0, it completes the current operation and cleanly halts. Until it is completely halted, it remains one when read.
Reset	0	RW	0	When this bit is set to 1, the MSI state machine is reset. This is a self-clearing bit.
Note: If not using Tagged Status mode, the driver should set 0x6000 bit 5.				

MSI Status Register (offset: 0x6004)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:5	RO	0	–
PCI Parity Error	4	W2C	0	PCI parity error status
PCI Master Abort	3	W2C	0	PCI master abort status
PCI Target Abort	2	W2C	0	PCI target abort status
Reserved	1	RO	0	–
MSI PCI Request	0	RW	0	Reading this bit returns the current status of the request to PCI to send an MSI. If a value of 1 is read, then the request is currently asserted. Writing this bit with a value of one will cause the request to be asserted. Writing this bit with a value of 0 has no effect.

GRC Register

All registers reset are core reset unless specified.

Mode Control Register (offset: 0x6800)

Name	*Bits*	*Access*	*Default Value*	*Description*
TLP Address 2	31	RW	0	TLP private register upper address 2 Bits [31][22][29] remap the PCIe core TL/DL/PL register to the GRC space from 0x6400 to 0x67ff [31]: 0: Selects the lower 1 KB of each TL/DL/PL. 1: Selects the higher 1 KB of each TL/DL/Pl. [22][29]: 00: Selects the TL register. 01: Selects the DL register. 10: Selects the PL register.
Multi_Cast Enable	30	RW	0	Multi-Cast Enable
TLP Address 0	29	RW	0	TLP private register upper address 0 Bits [31][22][29] remap the PCIe core TL/DL/PL register to the GRC space from 0x6400 to 0x67ff. [31]: 0: Selects the lower 1 KB of each TL/DL/PL. 1: Selects the higher 1 KB of each TL/DL/Pl. [22][29]: 00: Selects the TL register. 01: Selects the DL register. 10: Selects the PL register.
Interrupt on Flow Attention	28	RW	0	Cause a host interrupt when an enabled flow attention occurs
Interrupt on DMA Attention	27	RW	0	Cause a host interrupt when an enabled DMA attention occurs
Interrupt on MAC Attention	26	RW	0	Cause a host interrupt when an enabled MAC attention occurs
Interrupt on RX RISC Attention	25	RW	0	Cause a host interrupt when an enabled RX-RISC attention occurs
Reserved	24	RO	0	–
Receive No Pseudo-header checksum	23	RW	0	Do not include the pseudo-header in the TCP or UDP checksum calculations. To obtain the correct checksum, the driver must add the TCP/UDP checksum field to the pseudo-header checksum.

Name	*Bits*	*Access*	*Default Value*	*Description*
TLP Address 1	22	RW	0	TLP private register upper address 1 Bits [31][22][29] remap the PCIe core TL/DL/PL register to the GRC space from 0x6400 to 0x67ff [31]: 0: Selects the lower 1 KB of each TL/DL/PL. 1: Selects the higher 1KB of each TL/DL/Pl. [22][29]: 00: Selects the TL register. 01: Selects the DL register. 10: Selects the PL register.
NVRAM Write Enable	21	RW	0	The host must set this bit before attempting to update the Flash or SEEPROM
Send No Pseudo-header checksum	20	RW	0	Do not include the pseudo-header in the TCP or UDP checksum calculations. To obtain the correct checksum, the driver must seed the TCP/UDP checksum field with the pseudo-header checksum.
1588 TSYNC enable	19	RW	0	Enable 1588 TSYNC feature.
EAV Mode Enable	18	RW	0	Write 1 to this bit to enable EAV Mode. Enabling EAV Mode shall internally over-ride RSS Mode setting to a disable. Disabling EAV Mode shall hide Send Ring#2 and other EAV related registers noted in this section
Host Send BDs	17	RW	0	Use host-based BD rings instead of NIC-based BD rings.
Host Stack Up	16	RW	0	The host stack is ready to receive data from the NIC.
Reserved	15	RO	0	–
Don't Interrupt on Receives	14	RW	0	Never cause an interrupt on receive return ring producer updates.
Don't Interrupt on Sends	13	RW	0	Never cause an interrupt on send BD ring producer updates.
DMA Write System ATTN	12	RW	0	DMA Write System Attention Enable
Allow Bad Frames	11	RW	0	The RX MAC forwards illegal frames to the NIC and marks them as such instead of discarding them. The frames are queued based on default class and interrupt distribution queue number
No_CRC	10	RO	0	Used by EMAC

Name	*Bits*	*Access*	*Default Value*	*Description*
No Frame Cracking	9	RW	0	Turn off all frame cracking functionality in both the read DMA engine and the MAC receive engine. On receive, the TCP/ UDP checksum field is replaced by raw checksum for the whole frame except the Ethernet header. On transmit, IP and TCP/UDP checksum generation is always disabled when this bit is set. Also, the raw checksum is calculated over the entire frame except the Ethernet header and CRC.
Reserved	8	RO	0	–
CR_Func_Sel	7:6	RW	0	Used to select the Card-Reader Function Number when accessing the Card-Reader Register from the internal GRC Data Path 00: Function 0 – Not Applicable 01: Function 1 – Select SD/MMC Function 1 Register
Word Swap Data	5	Host-RW NIC-R	0	Word swap data when DMAing it across the PCIe bus.
Byte Swap Data	4	Host-RW NIC-R	0	Byte swap data when DMAing it across the PCIe bus.
Reserved	3	RO	0	–
Word Swap BD	2	Host-RW NIC-R	0	Word swap BD structure when DMAing them across the PCIe bus.
Byte Swap BD	1	Host-RW NIC-R	0	Byte swap BD structure when DMAing them across the PCIe bus.
Int_Send_Tick	0	RO	0	Used by Host Coalescing Block

Miscellaneous Configuration Register (offset: 0x6804)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31	RO	0	–
Reserved	30	RO	0	–
Disable GRC Reset on PCIE block	29	RW	0	Setting this bit will prevent reset to PCIe block
Reserved	28	RO	0	–
Reserved	27	RO	0	–
Reserved	26	RW	0	–
Reserved	25	RW	0	–
RAM Powerdown	24	RW	0	When this bit is set, all of the RAMs are powered down.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	23	RW	0	–
BIAS IDDQ	22	RW	0	When this bit is set, the BIAS will be powered down.
GPHY IDDQ	21	RW	0	When this bit is set, the GPHY will be powered down.
Reserved	20	RO	0	–
Vmain_prsnt State	19	RO	1	State of Vmain_prsnt input for this device.
Power State	18:17	RO	0	Indicates the current power state of the device. 00b: D0 01b: D1 02b: D2 03b: D3 This PowerState mirrors the PMSCR register
Reserved	16:13	RO	0	–
Reserved	12:8	RW	0x10	–
Timer Prescaler	7:1	RW	1111111b	Local Core clock frequency in MHz, minus 1, which should correspond to each advance of the timer. Reset to all 1. This timer Prescaler is also control the tick timers (receive and send tick timers) in the Host Coalesce Block. This parameter is no longer applicable in BCM5725/ BCM5762/BCM57767.
GRC Reset	0	RW	0	Write 1 to this bit resets the CORE_CLK blocks in the device. This is a self-clearing bit.

Miscellaneous Local Control Register (offset: 0x6808)

The Miscellaneous Local Control register is used to control various functions within the device. All bits are set to zero (i.e. disabled) during reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Wake On Link Up	31	RW	0	When set, the chip drives the PME when the link is up.
Enable Wake On Link Down	30	RW	0	When set, the chip drives the PME when the link is down.
Disable Traffic LED fix	29	RW	0	Set to t to disable Traffic LED Fix (CQ9609)
Reserved	28:27	RO	0	–
PME Assert	26	RW	0	When set, the PME Status bit in the PMSCR register is forced high. If PME Enable is also set, the PME signal will activate. This register bit is write-only and self-clearing after write.
Reserved	25:17	RO	0	–
GPIO(2:0) Output	16:14	RW	0	Outputs which are defined by board level design.

Name	*Bits*	*Access*	*Default Value*	*Description*
GPIO(2:0) Output Enable	13:11	RW	0	When asserted, the device drives miscellaneous pin outputs.
GPIO(2:0) Input	10:8	RO	0	Input from bidirectional miscellaneous pin. GPIO0 is accessible in 48-pin package GPIO0 and GPIO2 are accessible in 68-pin package
Reserved	7:6	RW	0	Reserved
Energy detection pin	5	RW	0	Input value of ENERGY_DET pin
UART Disable	4	RW	Inversion of OTP bit 141	Disable UART on SPD100LEDB/TRAFFICLEDB pin x0: TRAFFICLEDB pin is used as UART serial in SPEED100LEDB pin is used as UART serial out 0x1: TRAFFICLEDB/SPEED100LEDB pin is controlled by MAC This register bit resets by POR only.
Interrupt on Attention	3	RW	0	If set, the host will be interrupted when any of the attention bits in the CPU event register are asserted.
Set Interrupt	2	WO	0	If Interrupt Mailbox 0 contains a nonzero value, setting this bit does nothing. If Interrupt Mailbox 0 is zero, then setting this bit will cause the internal unmasked interrupt state to be asserted. The external interrupt state (INTA pin) will also be asserted immediately if interrupts are not masked by the Mask Interrupts bit. If interrupts are masked, INTA will be asserted once interrupts are unmasked, so long as interrupts are not first cleared. This bit is not operational in MSI mode.
Clear Interrupt	1	WO	0	This bit provides the same functionality as the Clear Interrupt bit in the Miscellaneous Host Control register. This bit is not operational in MSI mode
Interrupt State	0	RO	0	This bit reflects the state of the PCI INTA pin. This bit is not operational in MSI mode.

Timer Register (offset: 0x680C)

The Timer register is a 32-bit free-running counter. This counter increments when the Prescale Counter hits the Timer Prescaler limit as specified by the Miscellaneous Configuration register. This counter is used by the CPU to keep track of relative time in microseconds. A write to the Timer register will load the counter value written.

Name	*Bits*	*Access*	*Default Value*	*Description*
Timer Value	31:0	RW	0	32-bit free-running counter

RX-CPU Event Register (offset: 0x6810)

Name	*Bits*	*Access*	*Default Value*	*Description*
SW Event 13	31	RO	0	SW Event 13 is set; This bit is Flash Attention;
Reserved	30	RO	0	Reserved
Timer	29	RW	0	Timer Reference reached
SW Event 11	28	RW	0	SW Event 11 is set
Flow Attention	27	RO	0	Flow Attention
RX CPU Attention	26	RW	0	RX CPU needs attention
MAC Attention	25	RO	0	MAC needs attention
Reserved	24	RO	0	–
SW Event 10	23	RW	0	SW Event 10 is set
High Priority Mailbox	22	RO	0	First 32 Mailbox registers have been updated
Low Priority Mailbox	21	RO	0	Last 32 Mailbox registers have been updated
DMA Attention	20	RO	0	A DMA channel needs attention
SW Event 9	19	RW	0	SW Event 9 is set
High DMA RD	18	RO	0	High Priority DMA read FTQ has stalled
High DMA WR	17	RO	0	High Priority DMA write FTA has stalled
SW Event 8	16	RW	0	SW Event 8 is set
Host Coalescing	15	RO	0	The host coalescing FTQ has stalled
SW Event 7	14	RW	0	SW Event 7 is set
Receive Data Comp (Post DMA)	13	RO	0	Receive data completion FTQ has stalled
SW Event 6	12	RW	0	SW Event 6 is set
RX SW Queue Event	11	RO	0	Receive Software Queue Event
DMA RD	10	RO	0	Normal Priority DMA read FTQ has stalled
DMA WR	9	RO	0	Normal Priority DMA write FTQ has stalled
Read DMA Init (Pre DMA)	8	RO	0	Receive Data and Receive BD Initiator FTQ has stalled
SW Event 5	7	RW	0	SW Event 5 is set
Recv BD Comp	6	RO	0	Receive BD Completion FTQ has stalled
SW Event 4	5	RW	0	SW Event 4 is set
Recv List Selector	4	RO	0	Receive List Selector is nonzero
SW Event 3	3	RW	0	SW Event 3 is set
Recv List Placement	2	RO	0	Receive List Placement FTQ has stalled
SW Event 1	1	RW	0	SW Event 1 is set
SW Event 0	0	RW	0	SW Event 0 is set

RX-CPU Timer Reference Register (offset: 0x6814)

The Timer Reference register allows the RX-RISC to receive an event when the free-running Timer register counts up to this value.

Name	*Bits*	*Access*	*Default Value*	*Description*
RX-CPU Timer Reference	31:0	RW	0	RX-RISC Timer Event when time stamp = RX-RISC Timer Reference

RX-CPU Semaphore Register (offset: 0x6818)

The RX-RISC Semaphore register allows access to both internal RISC processors to a hardware semaphore mechanism. Writes to the register indicates the preference to toggle the own/not own states of a single semaphore bit. Reads of this register provide a 1 if that register owns the semaphore, and a 0 otherwise. To obtain the semaphore, the normal operation is a loop containing a write 0 followed by a read. Exit the loop when the read returns nonzero. To release the semaphore,

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	–
RX-CPU Semaphore	0	RW	0	RX-CPU Semaphore

PCIe Misc. Status Register (offset: 0x681C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	Unused
P1 PCIE Ack FIFO Underrun	23	RO	0	PCIe Ack FIFO Underrun Error Status for CR DMA Write
P0 PCIE Ack FIFO Underrun	22	RO	0	PCIe Ack FIFO Underrun Error Status for LAN DMA Write
P1 PCIE Ack FIFO Overrun	21	RO	0	PCIe Ack FIFO Overrun Error Status for CR DMA Write
P0 PCIE Ack FIFO Overrun	20	RO	0	PCIe Ack FIFO Overrun Error Status for LAN DMA Write
Reserved	19:17	RO	0	Unused
PCIE Link in L23	16	RO	0	PCIe Link In L23 State
F0_PCIE_PowerState	15:14	RO	0	PCIe Function 3 Power State Status
F1_PCIE_PowerState	13:12	RO	0	PCIe Function 3 Power State Status
F2_PCIE_PowerState	11:10	RO	0	PCIe Function 3 Power State Status
F3_PCIE_PowerState	9:8	RO	0	PCIe Function 3 Power State Status

Name	Bits	Access	Default Value	Description

PCIE Phy Attn	7:4	RO	0	PCIe Attention Error Status
				casex ({txintf_overflow_attn, rtag_val_unexp_attn,
				tx_tag_in_use_attn , unknowntype_err_attn,
				bridge_forward_err_attn, illegal_size_attn,
				ecrc_attn, rx_unsupport_attn, cpl_unexp_attn,
				cpl_timeout_attn, poison_attn, cpl_abrt_attn})
				
				12'bxxxx_xxxx_xxx1 : pcie_err_status = 4'h0;
				12'bxxxx_xxxx_xx10 : pcie_err_status = 4'h1;
				12'bxxxx_xxxx_x100 : pcie_err_status = 4'h2;
				12'bxxxx_xxxx_1000 : pcie_err_status = 4'h3;
				12'bxxxx_xxx1_0000 : pcie_err_status = 4'h4;
				12'bxxxx_xx10_0000 : pcie_err_status = 4'h5;
				12'bxxxx_x100_0000 : pcie_err_status = 4'h8;
				12'bxxxx_1000_0000 : pcie_err_status = 4'h9;
				12'bxxx1_0000_0000 : pcie_err_status = 4'ha;
				12'bxx10_0000_0000 : pcie_err_status = 4'hc;
				12'bx100_0000_0000 : pcie_err_status = 4'hd;
				12'b1000_0000_0000 : pcie_err_status = 4'he;
				default : pcie_err_status = 4'hf;

PCI_GRC_INTB_F3	3	RO	0	PCIe Function 3 Int-B Status
PCI_GRC_INTB_F2	2	RO	0	PCIe Function 2 Int-B Status
PCI_GRC_INTB_F1	1	RO	0	PCIe Function 1 Int-B Status
PCI_GRC_INTA	0	RO	0	PCIe Function 0 Int-A Status

Card-Reader DMA Read Policy Register (offset: 0x6820)

Name	Bits	Access	Default Value	Description
Reserved	31:17	RO	0x0	–
Deprecate DMAR Channel#2	16	RW	0x0	When this bit is 0, the LAN DMAR Channel#2 (RTT) gets the higher (fixed) priority over the winner of LAN DMAR Channel#1 versus CR DMAR. When this bit is 1, the LAN DMAR Channel#2 (RTT) AND the winner of LAN DMAR Channel#1 versus CR DMAR are arbitrated in a simple Round Robin manner.
Reserved	15:11	RO	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Max Credit	10:8	RW	0x0	The Max Credit value of the DMAR arbiter can be set by programming the field. 000 == 2 Credits 001 == 2 Credits 010 == 2 Credits 011 == 3 Credits 100 == 4 Credits 101 == 5 Credits 110 == 6 Credits 111 == 7 Credits This field is meaningful only when the field [2 :0] is 101 or 110, else this field is ignored.
Reserved	7:3	RO	0x0	–
DMAR Arbitration Policy	2:0	RW	0x0	This field chooses the arbitration policies: 000 == Simple Round Robin 001 == Fixed Priority, LAN Highest 010 == Fixed Priority, CR Highest 011 == Reserved 101 == Credit Based – LAN Bias 110 == Credit Based – CR Bias 111 == Reserved

Card-Reader DMA Write Policy Register (offset: 0x6824)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:11	RO	0x0	–
Max Credit	10:8	RW	0x0	The Max Credit value of the DMAR arbiter can be set by programming the field. 000 == 2 Credits 001 == 2 Credits 010 == 2 Credits 011 == 3 Credits 100 == 4 Credits 101 == 5 Credits 110 == 6 Credits 111 == 7 Credits This field is meaningful only when the field [2 :0] is 101 or 110, else this field is ignored.
Reserved	7:3	RO	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
DMAW Arbitration Policy	2:0	RW	0x0	This field chooses the arbitration policies: 000 == Simple Round Robin 001 == Fixed Priority, LAN Highest 010 == Fixed Priority, CR Highest 011 == Reserved 101 == Credit Based – LAN Bias 110 == Credit Based – CR Bias 111 == Reserved

RX CPU Event Enable Register (offset: 0x684C)

Setting a bit in this register enables an interrupt to the CPU or the Event.

Name	*Bits*	*Access*	*Default Value*	*Description*
Flash	31	RW	0	–
Reserved	30	RW	0	–
Timer Reference Reached	29	RW	0	–
ROM	28	RW	0	–
HC Module	27	RW	0	–
RX CPU Module	26	RW	0	–
EMAC Module	25	RW	0	–
Memory Map Enable Bit	24	RW	0	Set by HW, Cleared by SW
Reserved	23	RW	0	–
High Priority Mailbox	22	RW	0	–
Low Priority Mailbox	21	RW	0	–
DMA	20	RW	0	–
Reserved	19	RW	0	–
Reserved	18	RW	0	–
Reserved	17	RW	0	–
ASF Location 15	16	RW	0	–
TPM Interrupt Enable	15	RW	0	–
ASF Location 14	14	RW	0	–
Reserved	13	RW	0	–
ASF Location 13	12	RW	0	–
Unused SDI	11	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
SDC (Post TCP segmentation)	10	RW	0	–
SDI (Pre TCP segmentation)	9	RW	0	–
RDIQ FTQ (Received an ASF)	8	RW	0	–
ASF Location 12	7	RW	0	–
Reserved	6	RW	0	–
ASF Location 11	5	RW	0	–
Reserved	4	RW	0	–
ASF Location 10	3	RW	0	–
Reserved	2	RW	0	–
ASF Location 9	1	RW	0	–
ASF Location 8	0	RW	0	–

Secfg Config 1 Register (offset: 0x6880)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
CR_VDDIO 3.0V regulator output adjustment	31:28	RW	7	CR_VDDIO regulator output adjustment at 3.0V (approximately, output maxed-out at 3.0V) 0000: +14% 0001: +12% 0010: +10% 0011: +8% 0100: +6% 0101: +4% 0110: +2% 0111: 0% 1000: –2% 1001: –4% 1010: –6% 1011: –8% 1100: –10% 1101: –12% 1110: –14% 1111: –16%

Name	*Bits*	*Access*	*Default Value*	*Description*
CR_VDDIO 1.8V regulator output adjustment	27:24	RW	7	CR_VDDIO regulator output adjustment at 1.8V (approximately) 0000: +14% 0001: +12% 0010: +10% 0011: +8% 0100: +6% 0101: +4% 0110: +2% 0111: 0% 1000: –2% 1001: –4% 1010: –6% 1011: –8% 1100: –10% 1101: –12% 1110: –14% 1111: –16%
SI/EEDATA Pin Strength Control	23:21	RW	Inversion of OTP bit 159:157	Pad Strength Control 111 = TH16 110 = TH12 011 = TH8 001 = TH4
SO Pin Strength Control	20:18	RW	Inversion of OTP bit 156:154	Pad Strength Control 111 = TH16 110 = TH12 011 = TH8 001 = TH4
SCLK Pin Strength Control	17:15	RW	Inversion of OTP bit 153:151	Pad Strength Control 111 = TH16 110 = TH12 011 = TH8 001 = TH4
SO Pin Strength Control	14:12	RW	Inversion of OTP bit 150:148	Pad Pin Strength Control 111 = TH16 110 = TH12 011 = TH8 001 = TH4
Flash/LED Pin Sharing Control	11	RW	OTP bit 131	Disables Flash/LED Pin sharing 0: Enable Flash/LED Pin sharing 1: Disable Flash/LED Pin sharing; Flash pins will only be used for flash purpose

Name	*Bits*	*Access*	*Default Value*	*Description*
SD_CLK pull-up Control	10	RW	0	Enables internal chip pull-up on CR_CLK pin when SD card is detected 0: Disable internal chip pull-up 1: Enable internal chip pull-up
xD R_B_N pull-up Control	9	RW	0	Enables internal chip pull-up on CR_CLK (R_B_N) pin when xD card is detected 0: Disable internal chip pull-up 1: Enable internal chip pull-up
GPIO0/SD Bus_Pow Control	8	RW	0	Selects between GPIO0 function or SD Bus_Pow function for GPIO0 pin 0: Use GPIO0 pin as GPIO0 function 1: Reserved
SD Bus_Pow/LED Control	7	RW	0	Selects between SD Bus_Pow function or LED function for CR_LED pin (non-xD sku only) 0: Use CR_LED pin as SD Bus_Pow function 1: Use CR_LED pin as SD LED function
SD LED Output Mode Control	6:5	RW	1	SD LED Output Mode Control 00: Active High 01: Active Low 10: Open Drain 11: Open Collector
SD Bus_Pow Output Polarity Control	4	RW	0	SD Bus_Pow Output Polarity Control 0: Active High 1: Active Low
SD Write Protect Polarity Control	3	RW	0	SD Write Protect Polarity Control 0: Active High (internal chip pull-down is activated) 1: Active Low (internal chip pull-up is activated)
SD/MMC Card Detect Polarity/Reserved	2	RW	0	SD/MMC Card Detect Polarity Control — Not supported in A0 but supported in B0 0: Active Low (internal chip pull-up is activated) 1: Active High (internal chip pull-down is activated)

Secfg Config 2 Register (offset: 0x6884)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RW	0	–
SD Write Protect Internal Chip Pull-up/ Pull-down Override Control	7:6	RW	0	SD Write Protect internal chip pull-up/pull-down override control 0: Disable Override 1: Activates Pull-up and Deactivates Pull-down 2: Activates Pull-down and Deactivate Pull-down 3: Deactivates Pull-up and Pull-down
Reserved	5:4	RO	0	–

Bond ID Register (offset: 0x6888)

Name	*Bits*	*Access*	*Default Value*	*Description*
Serdes L0 exit latency select	31:30	RO	0	Value of OTP bit 144:143 SerDes L0 exit latency select (f/w use) 00 = slow 01 = normal 10 = fast 11 = reserved
UMC BG WA (f/w use)	29	RO	0	Value of OTP bit 142 UMC BG WA (f/w use) 0 = WA disabled 1 = WA enabled
UART enable	28	RO	0	Value of OTP bit 141 UART enable (reset value of inversion of 0x6808.4) 1 = UART 0 = LED
eAV Disable	27	RO	0	Value of OTP bit 140 eAV Disable 1 = eAV Disable 0 = eAV Enable
SEDATA OE control	26	RO	0	Value of OTP bit 139 SEDATA OE control 0 = i2c 1 = legacy (reset value of inversion of 0x7024.3)

Name	*Bits*	*Access*	*Default Value*	*Description*
Disable auto eeprom reset	25	RO	0	Value of OTP bit 138 Disable auto eeprom reset 0 = enable 1 = disable (reset value of 0x7024.4)
EEE LPI Enable H/W Default	24	RO	0	Value of OTP bit 137 EEE LPI Enable H/W Default 1 = enable EEE LPI 0 = disable EEE LPI
PCIE Gen2 mode	23	RO	0	Value of OTP bit 136 PCIe Gen2 mode 1 = Gen2 0 = Gen1
Vaux _prsnt	22:21	RO	0	Value of OTP bit 135:134 Vaux _prsnt 11 = 1 10 = 0 01 = 0 00 = 1
Non-CR sku	20	RO	0	Value of OTP bit 133 Non-CR sku 1 = Non-CR sku (48-pin package) 0 = CR sku (68-pin package)
Disable Gigabit	19	RO	0	Value of OTP bit 132 Disable Gigabit 1 = disable gigabit 0 = enable gigabit
Disable LED pin sharing	18	RO	0	Value of OTP bit 131 Disable LED pin sharing 1 = disable LED's on SCLK and SO pin 0 = enable
CR regulator power down	17	RO	0	Value of OTP bit 130 CR regulator power down 1 = power down 0 = power up
Bond ID	16:0	RO	0	H/w Bond ID BCM57781: 0x016B1 or 0x116B1 BCM5725/BCM5762/BCM57767: 0x016B5 BCM5725/BCM5762/BCM57767X: 0x116B5 BCM57761: 0x016B0 or 0x116B0 BCM57765: 0x016B4 BCM57765X: 0x116B4 BCM57791: 0x016B2 or 0x116B2 BCM57795: 0x016B6 BCM57795X: 0x116B6

Clock Control Register (offset: 0x688C)

Name	*Bits*	*Access*	*Default Value*	*Description*
PL Clock Disable	31	R/W	0	When this bit is set to 1, PCI Express Physical Layer Clock is disabled
DLL Clock Disable	30	R/W	0	When this bit is set to 1, PCI Express Data Link Layer Clock is disabled
TL Clock Disable	29	R/W	0	When this bit is set to 1, PCI Express Transaction Layer Clock is disabled Once this bit is set, the FW can no longer access the PCI Config registers (including the clock control register), so it should be set only when the FW has no need to access the clock control and other PCI config registers until next power cycle. N/A for PCI Device*
PCI Express Clock to Core Clock	28	R/W	0	When this bit is 1, the source of internal PCI Express Clock is CORE_CLK
Reserved	27	R/W	0	–
Reserved	26	R/W	0	–
Reserved	25	R/W	0	–
Reserved	24	R/W	0	–
Reserved	23	R/W	0	–
Reserved	22	R/W	0	–
Reserved	21	R/W	0	–
Select Final Alt Clock Source: 0 = Alt Clock Source 1, 1 = 6.25 MHz	20	R/W	0	Select the 6.25 MHz clock as the alternate clock (use in Airplane Mode). If this bit is 0, the alternate clock will be selected by bit 13.
Slow Core Clock Mode	19	R/W	0	Set this bit to 1 when running a 10:1 PCI to Core clock ratio. For engineering debug only.
LED polarity	18	R/W	0	When set to 1, polarity of the 4 LEDs is inverted
BIST function control	17	R/W	0	–
Asynchronous BIST Reset	16	R/W	0	–
Reserved	15:14	–	–	–
Select Alt Clock Source 1: 0 = ck25 (XTAL_IN)/2, 1 = MII_CLK/2	13	R/W	0	Use the MII CLK input as the alternate clock for the internal clocks, rather than the Xtal CK25 input as the alternate clock.
Select Alt Clock	12	R/W	0	Use the alternate clock as the clock reference for the internal clocks, rather than the 62.5 MHz.
Reserved	11:10	–	–	–
Core Clock Disable	9	R/W	0	Disable the CORE CLK to all blocks and shut down all ram's except bd and txmbuf memories
Reserved	8	–	–	–

Name	Bits	Access	Default Value	Description
Reserved	7	–	–	–
Reserved	6:5	–	–	–
Reserved	4:0	–	–	–

Miscellaneous Control Registers

Miscellaneous Control Register (offset: 0x6890)

Name	Bits	Access	Default Value	Description
Done_dr_fix_enable4	31	RW	0	Enable Bug Fix4 for Done Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix
Done_dr_fix_enable3	30	RW	0	Enable Bug Fix3 for Done Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix
Done_dr_fix_enable2	29	RW	0	Enable Bug Fix2 for Done Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix
Done_dr_fix_enable1	28	RW	0	Enable Bug Fix1 for Done Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix
Clkreq_Delay_Disable	27	RW	0	When set, gate off the DMA Write Request When clear, enable the DMA Write Request to pass to the PCIe Block. This bit was introduced to address CQ14159 1: Disable Fix 0: Enable Fix This bit is reset by Hard_Reset (POR, Exit Low Power Mode, Lost of Vmain while in D0)
Lcrc_dr_fix_enable2	26	RW	0	Enable Bug Fix2 for LCRC Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix
Lcrc_dr_fix_enable	25	RW	0	Enable Bug Fix1 for LCRC Logic in DMA Read Module 1: Enable Bug Fix 0: Disable Bug Fix

Name	*Bits*	*Access*	*Default Value*	*Description*
Chksum_fix_enable	24	RW	0	Enable Checksum Bug Fix in DMA Read Module 1 – Enable Bug Fix 0 – Disable Bug Fix
Ma_addr_fix_enable	23	RW	1	Enable Bug Fix in Memory Arbiter to prevent address from incrementing 1: Enable Bug Fix 0: Disable Bug Fix
Ma_prior_enable	22	RW	0	This bit control the priority of the Memory Arbiter Requests between RT and LSO requests when both requests occur simultaneously 1: Select RT Request over LSO Request 0: Select LSO Request over RT Request
Underrun_Fix_Enable	21	RW	0	This bit enable the PCIe ACK FIFO Underrun when Watermark is less than MPS 1: Enable Bug Fix 0: Disable Bug Fix
Underrun Clear	20	RW	0	Clear PCIe Ack Underrun Error Status 1 – Clear Underrun
Overrun Clear	19	RW	0	Clear PCIe Ack Overrun Error Status 1: Clear Overrun
Reserved	18:0	RO	0	–

Fast Boot Program Counter Register (offset: 0x6894)

Name	*Bits*	*Access*	*Default Value*	*Description*
Fastboot Enable	31	RW	0	This bit is used by the CPU to keep track of whether or not there is valid phase 1 boot code stored in the RX MBUF. If the bit is set, then RXMBUF contains valid boot code. Otherwise, it is assumed that RXMBUF does not contain valid boot code. This bit is reset only by a power-on reset. The state of this bit has no effect on state machines within the device. It is used by the CPU to track boot code status.
Fastboot Program Counter	30:0	RW	0	This field is used by the CPU to keep track of the location of the phase 1 boot code in RX MBUF. These bits behave identical to bit 31 in that they have no effect on state machine operation and they are cleared only by a power-on reset.

Power Management Debug Register – Debug Control (offset: 0x68A4)

Some of the bit in this register is initialized by POR only.

<table>
<tr><th>Name</th><th>Bits</th><th>Access</th><th>Default Value</th><th>Description</th></tr>
<tr><td>Reserved</td><td>31:28</td><td>R/W</td><td>0</td><td>Reserved</td></tr>
<tr><td>Tl_fsw_perst_dis</td><td>27</td><td>R/W</td><td>0</td><td>There are 3 conditions that are used to force the pcie tlclk to CK25 Mhz.:
1. PERST_L is low AND
2. PCIE Link is NOT in L23 AND
3. VMAIN is NOT present
Each of these three conditions have a corresponding control bit to enable/disable the condition in determining the clock switching.
Bit 27 of this register is used to override the PERST_L checking from the clock switching Logic
1: Disable PERST_L deassertion checking from the clock switching logic
0: Enable PERST_L deassertion checking for the clock switching Logic
assign pcie_clk_switcher_force_switch_condition = (
(~top_din_perst_l | tl_fsw_perst_dis) &   << Bit 27
(~pcie_link_in_l23 | tl_fsw_l23ready_dis) & << Bit 26
(~vddo_pci_rsb | ~tl_fsw_vmain_ena) << Bit 25
) & pcie_clk_switcher_clksel_scan & tlp_clk_switcher_clk0_enabled;</td></tr>
<tr><td>tl_fsw_l23ready_dis</td><td>26</td><td>R/W</td><td>0</td><td>There are 3 conditions that are used to force the pcie tlclk to CK25 Mhz.
1. PERST_L is low AND
2. PCIE Link is NOT in L23 AND
3. VMAIN is NOT present
Each of these three conditions have a corresponding control bit to enable/disable the condition in determining the clock switching
Bit 26 of this register is used to override the Link_in_L23 condition from the clock switching Logic
1: Disable l23_ready deassertion checking from the clock switching logic
0: Enable l23_ready deassertion checking for the clock switching Logic
assign pcie_clk_switcher_force_switch_condition = (
(~top_din_perst_l | tl_fsw_perst_dis) &   << Bit 27
(~pcie_link_in_l23 | tl_fsw_l23ready_dis) & << Bit 26
(~vddo_pci_rsb | ~tl_fsw_vmain_ena) << Bit 25
) & pcie_clk_switcher_clksel_scan & tlp_clk_switcher_clk0_enabled;</td></tr>
</table>

<table>
<tr><th>Name</th><th>Bits</th><th>Access</th><th>Default Value</th><th>Description</th></tr>
<tr><td>tl_fsw_vmain_ena</td><td>25</td><td>R/W</td><td>0</td><td>There are 3 conditions that are used to force the pcie tlclk to CK25 Mhz.
1. PERST_L is low AND
2. PCIE Link is NOT in L23 AND
3. VMAIN is NOT present
Each of these three conditions have a corresponding control bit to enable/disable the condition in determining the clock switching.
Bit 26 of this register is used to override the Link_in_L23 condition from the clock switching Logic.
1: Disable Vmain Present deassertion checking from the clock switching logic
0: Enable Vmain_Present deassertion checking for the clock switching Logic
`  assign pcie_clk_switcher_force_switch_condition = (`
`        (~top_din_perst_l  | tl_fsw_perst_dis) &   << Bit 27`
`          (~pcie_link_in_l23 | tl_fsw_l23ready_dis) & << Bit 26`
`        (~vddo_pci_rsb    | ~tl_fsw_vmain_ena)  << Bit 25`
`       ) & pcie_clk_switcher_clksel_scan & tlp_clk_switcher_clk0_enabled;`</td></tr>
<tr><td>tl_sw_perst_dis
PCLK Switcher Selection Condition A checking disable</td><td>24</td><td>R/W</td><td>0</td><td>When 36FC[6] is 0 this register bit disables perst_l checking
1: disable perst_l assertion checking when asserting force_switch
0: enable perst_l assertion checking when asserting force_switch</td></tr>
<tr><td>tl_sw_l23ready_dis
PCLK Switcher Selection Condition B checking disable</td><td>23</td><td>R/W</td><td>0</td><td>When 36FC[6] is 0 this register bit disables l23_ready checking
1: disable l23_ready assertion checking when asserting force_switch
0: enable l23_ready assertion checking when asserting force_switch</td></tr>
<tr><td>tl_sw_vmain_dis</td><td>22</td><td>R/W</td><td>0</td><td>When 36FC[6] is 0 this register bit disables vmain checking
1: disable vmain deassertion checking when asserting force_switch
0: enable vmain deassertion checking when asserting force_switch</td></tr>
<tr><td>Reserved</td><td>21:17</td><td>R/W</td><td>0</td><td>–</td></tr>
<tr><td>PERST Override</td><td>16</td><td>R/W</td><td>0</td><td>This bit is used to override the PERSTN so that the internal cpu can access the PCIe register when perstn is asserted
1: Override Perstn Reset
0: No Override
Reset by Hard Reset</td></tr>
<tr><td>Reserved</td><td>15:6</td><td>RO</td><td>0</td><td>–</td></tr>
<tr><td>pip_clkreq_serdes</td><td>5</td><td>RO</td><td>X</td><td>PCIe SerDes pipe clkreq status</td></tr>
</table>

Name	*Bits*	*Access*	*Default Value*	*Description*
pipe_aux_power_down	4	RO	X	PCIe SerDes pipe aux power down status
ipllpowerdown	3	RO	X	PLL Power Down Status
Iclkreq_oe_l	2	RO	X	Clock Request Output Enable Status
Reserved	1	RO	X	–
PLLisUp	0	RO	X	PCIe PLL Status

Miscellaneous Control 2 Register – Debug Control (offset: 0x68B0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:19	RW	0	–
EMAC Legacy Bug Fix	18	RW	1	This bit when set enable the EMAC to receive packet without dropping it after receiving a short invalid packet size (CQ25266) 1 : Enable Bug Fix 0: Disable Bug Fix
Reserved	17:2	RW	0	Reserved
Vmail Present Decay Debounce Period	1:0	RW	0	Debounce period for CQ23039 00- 640 us 01– 8 ms 10 – 20 ms 11 – 80 ms

TPH ISO Control Register (offset: 0x68C0)

Name		*Bits*	*Access*	*Default Value*	*Description*
Reserved		31:8	RO	–	–
ISO Send BD Read	Resvd	7	RO	0	–
	PH Value	6:5	RW	10	–
	TH Enable	4	RW	0	–
ISO Packet Data Read	Resvd	3	RO	0	–
	PH Value	1:0	RW	10	–
	TH Enable	0	RW	0	–

TPH Control Register (offset: 0x68FC)

Name		*Bits*	*Access*	*Default Value*	*Description*
Reserved		31:28	RO	–	–
MSI / MSI-X Vector Write	Resvd	27	RO	0	–
	PH Value	26:25	RW	10	–
	TH Enable	24	RW	0	–
Status Block Write	Resvd	23	RO	0	–
	PH Value	22:21	RW	10	–
	TH Enable	20	RW	0	–
Receive Return BD Write	Resvd	19	RO	0	–
	PH Value	18:17	RW	10	–
	TH Enable	16	RW	0	–
Packet Data Write	Resvd	15	RO	0	–
	PH Value	14:13	RW	10	–
	TH Enable	12	RW	0	–
Receive Producer BD Read	Resvd	11	RO	0	–
	PH Value	10:9	RW	10	–
	TH Enable	8	RW	0	–
Send BD Read	Resvd	7	RO	0	–
	PH Value	6:5	RW	10	–
	TH Enable	4	RW	0	–
Packet Data Read	Resvd	3	RO	0	–
	PH Value	1:0	RW	10	–
	TH Enable	0	RW	0	–

EAV REF COUNT LSB Register (offset: 0x6900)

Name	*Bits*	*Access*	*Default Value*	*Description*
EAV Reference Count [lower half]	31:3	RW	UUUU	LSB of the EAV Reference Count – Reading this LSB freezes the count and is only unfrozen when the corresponding MSB is read. Writing to this LSB latches the value and subsequent write to the MSB transfers the 64-bit value to EAV Ref Counter and counting resumes from there.
Reserved	2:0	RO	000	[2:0] is always 000.

EAV REF COUNT MSB Register (offset: 0x6904)

Name	Bits	Access	Default Value	Description
EAV Reference Count [Upper half]	31:0	RW	UUUU	MSB of the EAV Reference Count – Reading this MSB unfreezes the count which was earlier frozen by the corresponding LSB read. Writing to this MSB transfers the 64-bit value, this plus previously latched LSB, to EAV Ref Counter and counting resumes from there. Writing to this without writing to LSB has no effect.

EAV REF CLOCK CONTROL Register (offset: 0x6908)

This register controls the EAV Reference Counter and the TimeSync-related GPIO pins. Each MAC's 1588 HW owns a dedicated TimeSync_GPIO pin which may be connected to any of its four Snap-shot/WatchDog HW logic. If a MAC needs to use more pins beyond its TimeSync_GPIO pin, it may use any or all of the four APE_GPIO[3:0] pins – note that these pins are shared among APE HW and four MAC-1588 HW. Thus a platform must design-in these pins and have individual BootCode or FW configure this register and APE-GPIO register accordingly.

Note: HW behavior shall be indeterminate in case of conflicting or duplicate assignment of GPIO pins to the same resource. A platform must allocate its dedicated TimeSync_GPIO pin first before using any pin from APE_GPIO shared pool.

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	00	–
APE_GPIO[3] Mapping	29:27	RW	000	Same as below
APE_GPIO[2] Mapping	26:24	RW	000	Same as below
APE_GPIO[1] Mapping	23:21	RW	000	Same as below
APE_GPIO[0] Mapping	20:18	RW	000	An APE_GPIO[n] pin is mapped to 1588 input/output via this field: 000 => Do not use APE_GPIO[n] pin 001 => Reserved 010 => Reserved 011 => Reserved 100 => Use as Snap-Shot[0] Input Trigger 101 => Use as Snap-Shot[1] Input Trigger 110 => Use as Time Watchdog[0] Output 111 => Use as Time Watchdog[1] Output

Name	*Bits*	*Access*	*Default Value*	*Description*
TimeSync_GPIO Mapping	17:16	RW	00	The MAC/Port dedicated TimeSync_GPIO pin is mapped via this field: 00 => Use as Snap-Shot[0] Input Trigger 01 => Use as Snap-Shot[1] Input Trigger 10 => Use as Time Watchdog[0] Output 11 => Use as Time Watchdog[1] Output
Reserved	15:12	RO	0x00	–
Reset on Network Link Down -> Up	11	RW	0	–
Reset on Network Link Up -> Down	10	RW	0	–
Reset on GRC Reset and PCIe FLR	9	RW	0	Reset on GRC Reset pulse
Reset on PCIe reset	8	RW	0	Reset on deasserting edge of PCIe Reset
Reserved	7:3	RO	0x0	–
Resume EAV Ref Count	2	W1C	0	–
Stop EAV Ref Count	1	RW	0	–
Reset EAV Ref Count	0	W1C	0	–

EAV REF-COUNT SNAP-SHOT LSB Register (offset: 0x6910)

Name	*Bits*	*Access*	*Default Value*	*Description*
EAV Reference Count Snap-shot [lower half]	31:0	RO	U	LSB of the EAV Reference Count as snap-shotted by GPIO[0] [2:0] shall always be 000

EAV REF-COUNT SNAP-SHOT MSB Register (offset: 0x6914)

Name	*Bits*	*Access*	*Default Value*	*Description*
EAV Reference Count Snap-shot [Upper half]	31:0	RO	U	MSB of the EAV Reference Count as snap-shotted by GPIO[0]

TX TIME WATCHDOG LSB[0] REG (offset: 0x6918)

Name	*Bits*	*Access*	*Default Value*	*Description*
Watchdog LSB Value	31:0	RW	0x0000	See "TX TIME WATCHDOG MSB[0] REG (offset: 0x691C)" below.

TX TIME WATCHDOG MSB[0] REG (offset: 0x691C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Lock Timer	31	RW	0	Write a 1 to enable Time Watchdog[0]
Watchdog MSB Value	30:0	RW	0x00	This, concatenated with the Watchdog[0] LSB value, constitutes a 63-bit value. Precision of the value is 1ns – which equates to the precision of the EAV Reference Count. If bit[31] ==1, the desired Timesync/APE GPIO shall toggle as soon as EAV Reference Count[62:0] ***increments*** to match this 63-bit value. ***Note:*** Setting this time value back in time will produce no toggle.

TX TIME WATCHDOG LSB[1] REG (offset: 0x6920)

Name	*Bits*	*Access*	*Default Value*	*Description*
Watchdog LSB Value	31:0	RW	0x0000	See "TX TIME WATCHDOG MSB[0] REG (offset: 0x691C)" above.

TX TIME WATCHDOG MSB[1] REG (offset: 0x6924)

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable Lock Timer	31	RW	0	Write a 1 to enable Time Watchdog[0]
Watchdog MSB Value	30:0	RW	0x00	This, concatenated with the Watchdog[0] LSB value, constitutes a 63-bit value. Precision of the value is 1ns – which equates to the precision of the EAV Reference Count. If bit[31] ==1, the desired Timesync/APE GPIO shall toggle as soon as EAV Reference Count[62:0] ***increments*** to match this 63-bit value. ***Note:*** Setting this time value back in time will produce no toggle.

EAV REF CORRECTOR Register (offset: 0x6928)

Name	*Bits*	*Access*	*Default Value*	*Description*
Correction Enable	31	RW	0	Write a 1 to enable the correction feature.
Correction Sense	30	RW	0	0: The correction is an addition. 1: The correction is a subtraction.
Reserved	29:24	RO	0x00	Reserved
Correction Value	23:0	RW	0x000001	This value is accumulated in an accumulator every EAV REF CLK tick until it overflows. At that time, the EAV REF COUNT is corrected by a 8 ns unit and the acc is reloaded. A 0x0 value is not permissible.

EAV REF-COUNT SNAP-SHOT LSB[1] REG (offset: 0x6930)

This LSB & MSB pair captures the EAV Reference Count when externally triggered by the desired TimeSync/ APE_GPIO pin – a toggle serves a trigger. The only legal sequence of accessing this pair is Read-LSB followed by Read-MSB.

Name	*Bits*	*Access*	*Default Value*	*Description*
EAV Reference Count Snap-shot [lower half]	31:0	RO	U	LSB of the EAV Reference Count as snap-shot by TimeSync/APE_GPIO. [2:0] shall always be 000.

EAV REF-COUNT SNAP-SHOT MSB[1] REG (offset 0X6934)

Name	*Bits*	*Access*	*Default Value*	*Description*
EAV Reference Count Snap-shot [Upper half]	31:0	RO	U	MSB of the EAV Reference Count as snapshot by TimeSync/APE_GPIO

PCI Host Direct Register Access to PCIE Configuration Registers via BAR Access

The host can access the PCIE Configuration Register via direct memory cycle using BAR + {0-3ff}. In this case, the PCIE Client will translate these cycles to the internal GRC Bus with the address of 0x6C00-0x6FFF. This is needed in order to avoid the contention when the RXCPU is also accessing the various PCIE Configuration Registers in the different PCIE Functions.

GRC Register 0x6C00-0x6fff are mapped to PCIE Configuration Registers 0x0-0x3FF in the PCIE Function 0.

Nonvolatile Memory (NVM) Interface Registers

NVM Command Register (0x7000)

Name	*Bits*	*Access*	*Default Value*	*Description*
Policy Error	31:28	RO	0	Reports Address Lockout Policy Error violations
Atmel page size setting	27	RO	0	–
Reserved	26:23	RO	0	–
Reserved	22	–	–	–
Reserved	21	–	–	–
Reserved	20	–	–	–
Reserved – WRSR	19	RO	0	The write status register command bit. Set 1 will make the flash interface state machine generate wrsr_comd (0x1) to the flash device to set the status register of the flash device to be written with sr data. For SST25VF512 only.
Reserved EWSR	18	RO	0	The enable write status register command bit. Set 1 will make the flash interface state machine generate ewsr_cmd (0x50) to the flash device to set the status register for the flash device to be write-enabled. For SST25VF512 only.
Reserved - Write Disable Command	17	RO	0	The write disable command bit. Set 1 will make the flash interface state machine generate a write disable command cycle to the flash device to clear the write enable bit in the device status register. This command is used for devices with a write protection function.
Reserved - Write Enable Command	16	RO	0	The write enable command bit. Set 1 will make the flash interface state machine generate a write enable command cycle to the flash device to set the write enable bit in the device status register. This command is used for devices with a write protection function.
Reserved	15:11	RO	0	–
Atmel power of 2 page size config	10	RW	0	Program the page size of Atmel D device to be power of 2. Please note, a power cycle must be issued for this configuration to take effect.
Atmel page size read	9	RW	0	Read Atmel page size setting, the result is posted in bit 27 of the command register. Please note, issuing this command will effect the content of the read register (0x7010)
Last	8	RW	0	When this bit is set, the next command sequence is interpreted as the last one of a burst and any cleanup work is done. This means that the buffer is written to flash memory if needed on a write
First	7	RW	0	This bit is passed to the SEE_FSM or SPI_FSM if the pass_mode bit is set

Name	*Bits*	*Access*	*Default Value*	*Description*
Erase	6	RW	0	The erase command bit. Set high to execute an erase. This bit is ignored if the wr is clear
Wr	5	RW	0	The write/not read command bit Set to execute write or erase
Doit	4	RW	0	Command from software to start the defined command. The done bit must be clear before setting this bit. This bit is self clearing and will remain set while the command is active
Done	3	WTC	0	Sequence completion bit that is asserted when the command requested by assertion of the doit bit has completed. The done bit will be cleared while the command is in progress. The done bit will stay asserted until doit is reasserted or the done bit is cleared by writing a 1 to the done bit. The done bit is the FLSH_ATTN signal
Reserved	2:1	RO	0	–
Reset	0	RW		When set, the entire NVM state machine is reset. This bit is self clearing. Please note that this bit should NOT be set along with bit 4 (doit). Setting both bits will result a policy error in [31:28] and the reset command will be ignored.

NVM Status Register (offset: 0x7004h)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31	RO	0	–

Name	***Bits***	***Access***	***Default Value***	***Description***
SPI_AT_READ State	30:26	RO	0	SPI Atmel Read State ‘AT_RD_IDLE ‘AT_RD_AST_CS ‘AT_RD_WCS_SET ‘AT_RD_CMD ‘AT_RD_ADDR1 ‘AT_RD_ADDR2 ‘AT_RD_ADDR3 ‘AT_RD_DUMMY1 ‘AT_RD_DUMMY2 ‘AT_RD_DUMMY3 ‘AT_RD_DUMMY4 ‘AT_RD_RDATA1 ’AT_RD_RDATA2 ’AT_RD_RDATA3: ‘AT_RD_RDATA4 ‘AT_RD_SET_DONE ‘AT_RD_WNEXT: ‘AT_RD_WCS_HLD ‘AT_RD_WCS_MINH:

Name	*Bits*	*Access*	*Default Value*	*Description*
SPI_AT_Write State	25:20	RO	0	SPI Atmel Write State ‘AT_WR_IDLE ‘AT_TF_AST_CS ‘AT_TF_WCS_SET ‘AT_TF_CMD ‘AT_TF_ADDR1 ‘AT_TF_ADDR2 ‘AT_TF_ADDR3 ‘AT_TF_WCS_HLD ‘AT_TF_DST_CS ‘AT_TF_WCS_MINH ‘AT_TF_STS_AST_CS ‘AT_TF_STS_WCS_SET ‘AT_TF_STS_RCMD ‘AT_TF_STS_RDATA ‘AT_TF_STS_CHK ‘AT_TF_STS_WCLK ‘AT_TF_STS_WCS_HLD ‘AT_TF_STS_DST_CS: ‘AT_TF_STS_WCS_MINH ‘AT_WR_AST_CS ‘AT_WR_WCS_SET ‘AT_WR_CMD ‘AT_WR_ADDR1 ‘AT_WR_ADDR2 ‘AT_WR_ADDR3 ‘AT_WR_SDATA1 ‘AT_WR_SDATA2 ‘AT_WR_SDATA3 ‘AT_WR_SDATA4 ‘AT_WR_WNEXT ‘AT_WR_UPD_CMD ‘AT_WR_WCLK ‘AT_WR_WCS_HLD ‘AT_WR_DST_CS: ‘AT_WR_WCS_MINH: ‘AT_WR_STS_AST_CS: ‘AT_WR_STS_WCS_ ‘AT_WR_STS_RCMD: ‘AT_WR_STS_RDATA ‘AT_WR_STS_CHK

Name	Bits	Access	Default Value	Description
SPI_ST_Read State	19:16	RO	0	‘AT_WR_STS_WCLK ‘AT_WR_STS_WCS_HLD ‘AT_WR_STS_DST_CS ‘AT_WR_STS_WCS_MINH: SPI ST Read State ‘ST_RD_IDLE ‘ST_RD_AST_CS ‘ST_RD_CMD: ‘ST_RD_ADDR1 ‘ST_RD_ADDR2 ‘ST_RD_ADDR3: ‘ST_RD_RDATA1 ‘ST_RD_RDATA2 ‘ST_RD_RDATA3 ‘ST_RD_RDATA4 ‘ST_RD_SET_DONE ‘ST_RD_WNEXT ‘ST_RD_DST_CS: ‘ST_RD_WCS_MINH:

Name	*Bits*	*Access*	*Default Value*	*Description*
SPI_ST_Write State	15:10	RO	0	SPI ST Write State ‘ST_WR_IDLE ‘ST_WR_EN_AST_CS ‘ST_WR_ENABLE ‘ST_WR_EN_WCLK: ‘ST_WR_EN_DST_CS ‘ST_WR_EN_WCS_MINH: ‘ST_WR_AST_CS ‘ST_WR_CMD ‘ST_WR_ADDR1: ‘ST_WR_ADDR2: ‘ST_WR_ADDR3 ‘ST_WR_SDATA1: ‘ST_WR_SDATA2 ‘ST_WR_SDATA3: ‘ST_WR_SDATA4 ‘ST_WR_WNEXT: ‘ST_WR_UPD_CMD: ‘ST_WR_WCLK: ‘ST_WR_DST_CS: ‘ST_WR_WCS_MINH: ‘ST_WR_STS_AST_CS: ‘ST_WR_STS_CS_WCLK: ‘ST_WR_STS_RCMD: ‘ST_WR_STS_RDATA: ‘ST_WR_STS_CHK ‘ST_WR_STS_WCLK: ‘ST_WR_STS_DST_CS: ‘ST_WR_STS_WCS_MINH: ‘ST_WR_DIS_AST_CS: ‘ST_WR_DISABLE: ‘ST_WR_DIS_WCLK: ‘ST_WR_DIS_WCS_MINH:
SEQ_FSM State	9:6	RO	0	SEQ_IDLE SEQ_REQ SEQ_CMD SEQ_WNEXT SEQ_BITBANG:

Name	*Bits*	*Access*	*Default Value*	*Description*
SEE_FSM State	5:0	RO	0	SEE Read/Write State Read: 'EP_RD_IDLE: 0x0 'EP_RD_WR_START: 0x1 'EP_RD_WR_CTRL: 0x2 'EP_RD_WR_CWACK: 0x3 'EP_RD_WR_SDATAH: 0x4 'EP_RD_WR_ADDR : 0x5 'EP_RD_WR_AWACK: 0x6 'EP_RD_WR_ADDR2: 0x7 'EP_RD_WR_AWACK2: 0x8 'EP_RD_SDATAH: 0x9 'EP_RD_START: 0xA 'EP_RD_CTRL: 0xB 'EP_RD_CWACK: 0xC 'EP_RD_RDATA: 0xD 'EP_RD_SACK; 0xE 'EP_RD_WNEXT: 0x13 'EP_RD_SET_DONE: 0x0F 'EP_RD_SDATAL; 0x10 'EP_RD_STOP: 0x11 'EP_RD_TIMEOUT: 0x12 Write State: 'EP_WR_IDLE: 'EP_WR_START 'EP_WR_CTRL 'EP_WR_CWACK 'EP_WR_ADDR: 'EP_WR_AWACK: 'EP_WR_ADDR2: 'EP_WR_AWACK2 'EP_WR_SDATA: 'EP_WR_DWACK 'EP_WR_SET_DONE: 'EP_WR_SDATAL 'EP_WR_STOP: 'EP_WR_TIMEOUT:

NVM Write Register (offset: 0x7008h)

Name	*Bits*	*Access*	*Default Value*	*Description*
Write Data	31:0	RW	0	32 bits of write data are used when write commands are executed. When bitbang_mode is set, bits 0 to 3 control the drive value of the SCK, CS_L, SO, and SI pins respectively

NVM Address Register (offset: 0x700Ch)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
Write Address	23:0	RW	0	24bit address value When bitbang_mode is set, bits 0 to 3 control the OE value of the SCK, CS_L, SO, and SI pins respectively ***Note:*** SCL, SDA, SI are active high, and SCK, CS_L, and SO are active low

NVM Read Register (offset: 0x7010)

Name	*Bits*	*Access*	*Default Value*	*Description*
Read Data	31:0	RO	0	32 bits of read data are used when read commands are executed. When bitbang_mode is set, bits 0 to 3 reflect the current input value of SCK, CS_L, SO, and SI pins, respectively.

NVM Config 1 Register (offset: 0x7014h)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
SEE_CLK_DIV Enable	31	R/W	0	This bit enables 0x7014[21:11] as the SEE_CLK_DIV count for EPROM clock generation. 1'b0: 0x7014[21:11] is NOT used as SEE_CLK_DIV for EPROM clock generation. SEE_CLK_DIV is locked to 11'h10. 1'b1: 0x7014[21:11] is used as

Name	*Bits*	*Access*	*Default Value*	*Description*
Page Size	30:28	R/W	Depends on flash strapping	These bits indicate the page size of the attached flash device. The are set automatically depending on the chosen flash as indicated by the strapping option pins Page sizes are as follows: 000b: 256 bytes 001b: 512 bytes 010b: 1024 bytes 011b: 2048 bytes 100b: 4096 bytes 101b: 264 bytes 110b: 528 bytes 111b: reserved SW needs to read the page size to determine whether or not the external Atmel Device is 256Bytes page size or 264Bytes and then program the device accordingly.
Reserved	27	RO	Depends on Addr Lockout state	–
Reserved - Safe Erase	26	RO	0	–
Flash Size – strap bit 3	25	RO	Pin	Set this bit for a 1MB device or 0 for 512KB device Hard Reset, GRC Reset, and setting command register bit 0 will reset this bit to pin strap
Protect Mode – strap bit 2	24	RO	pin	Set this bit for flash devices that implement a write protect function Hard Reset, GRC Reset, and setting command register bit 0 will reset this bit to pin strap
Strap bit 5	23	RO	0	Always return 0
Strap bit 4	22	RO	0	0 for the following devices: EEPROM-64 kHz, EEPROM – 376 kHz, Microchip 24LC0X – 64 kHz, Microchip 24LC0x-376 kHz, AtmelAT45DB011B, Atmel AT25F512, SST25VF512, ST1Mbit, ST MP2505-A 1 for the following devices when Auto-Configure is enabled STM45PE10, STM45PE20, STM45PE40, AT45DB011B, AT45DB021B, and AT45DB041B

Name	*Bits*	*Access*	*Default Value*	*Description*
SEE_CLK_DIV	21:11	RW	0x10	This field is a divisor used to create all 1x times for all SEEPROM interface I/O pin timing definitions. A value of 0means that an SCL transitions at a minimum of each XTAL (25 MHz) rising edge. The equation to calculate the clock freq. for SCL is: XTAL_CLK / ((SEE_CLK_DIV + 1) x 4) ***Note:*** SCL is 4 times slower than 1x time. The default value corresponds to 368 kHz. In order for SCL clock generation logic to function, the NVRAM controller clock must be running faster than XTAL_CLK/((SEE_CLK_DIV + 1) x 2). A minimum delay of 10 NVRAM controller clocks must be placed between each SEE_CLK_DIV update and an EPROM access. This is to make sure SEE_CLK_DIV stabilizes and is safe to use in the CK_25 clock domain.
SPI_CLK_DIV	10:7	RW	4	This field is a divisor used to create all 1x times for all Flash interface I/O pin timing definitions. A divisor of 0 means that an SCK transitions at a minimum of each CORE_CLK rising edge. The equation to calculate the clock freq. for SCK is: CORE_CLK / ((SPI_CLK_DIV + 1) * 2) ***Note:*** SCLK is 4 times slower than 1x time. The default value corresponds to 6.25 MHz
Status	6:4	RW	0 if flash_mode 7 if buffer_mode X otherwise	This field represents the bit offset in the status command response to interpret as the ready flag
Reserved - Bitbang Mode	3	RO	0	–
Reserved - Pass Mode	2	RO	0	–
Buffer Mode	1	RW	Pin	Enable SSRAM Buffered Interface mode
Flash Mode	0	RW	Pin	Enable Flash Interface mode

NVM Config 2 Register (offset: 0x7018h)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
Status Command	23:16	RW	0xD7 if pin strap = Atmel 0x05 if pin strap = ST	This is the Flash status register read command.
Reserved	15:8	RO	0	–
Erase Command	7:0	RW	0x81 if pin strap = Atmel 0xDB if pin strap = STMxx	This is the Flash page erase command. Note: ST25xx does not support page erase, therefore the corresponding command is for sector erase.

NVM Config 3 Register (offset: 0x701C)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
Read Command	31:24	RW	0x03 if AT26DFXX 0xE8 if AT45DBXX non D 0x03 if AT45DBXX D 0x03 if STM25PEXX 0x03 if STM45PEXX	This is the Flash/SEEPROM read command Following this command, any number of bytes may be read up to the end of the flash memory For SEEPROM (flash mode = 0), this is SEEPROM read command. Bits(26:25) are address bits A1 and A0 of SEEPROM User should modify those two bits based on the value of A1 and A0 assigned to this SEEPROM device
Reserved - Buffer Write Command	23:16	RO	0	–
Write Command	15:8	RW	0x82 if AT26DFXX 0x82 if AT45DBXX 0x0A if STM25PEXX 0x0A if STM45PEXX	Command to write a series of bytes into a selected page in the Flash device. Note: this write command wraps around to the beginning of the page after the internal address counter in the Flash device reaches the end of the page. For SEEPROM (flash_mode = 0), this is SEEPROM write command. Bits[10:9] are address bits A1 and A0 of SEEPROM. User should modify those two bits based on the value of A1 and A0 assigned to this SEEPROM device
Reserved - Buffer Read Command	7:0	RO	0	–

Software Arbitration Register (Offset 0x7020)

With this Flash Controller, SW entity must win arbitration before accessing the NVRAM

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
REQ3	15	RO	0	Software request bit3 1 in this bit indicates that the request5 is active
REQ2	14	RO	0	Software request bit2 1 in this bit indicates that the request5 is active
REQ1	13	RO	0	Software request bit1 1 in this bit indicates that the request5 is active
REQ0	12	RO	0	Software request bit 0 When Req_set0 bit is set, this bit will be set
ARB_WON3	11	RO	0	Arbitration won bit 3 (see Bit 8, ARB_WON0)
ARB_WON2	10	RO	0	Arbitration won bit 2 (see Bit 8, ARB_WON0)
ARB_WON1	9	RO	0	Arbitration won bit 1 (see Bit 8, ARB_WON0)
ARB_WON0	8	RO	0	When req0 arbitration is won, this bit will be read as 1. When an operation is complete, then Req_clr0 must be written to clear bit. At that point, the next high priority arb bit will be set if requested. At any time, only one of the ARB_WON[5:0] bits will be read as 1. ARB 0 has the highest priority, and ARB5 has the lowest priority
REQ_CLR3	7	WO	X	Write 1 to this bit to clear REQ3 bit
REQ_CLR2	6	WO	X	Write 1 to this bit to clear REQ2 bit
REQ_CLR1	5	WO	X	Write 1 to this bit to clear REQ1 bit
REQ_CLR0	4	WO	X	Write 1 to this bit to clear REQ0 bit
REQ_SET3	3	WO	X	Write 1 to this bit to set REQ3 bit
REQ_SET2	2	WO	X	Write 1 to this bit to set REQ2 bit
REQ_SET1	1	WO	X	Write 1 to this bit to set REQ1 bit
REQ_SET0	0	WO	X	Set software arbitration request bit 0. This bit is set by writing 1.

NVM Access Register (offset: 0x7024h)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0	–
ST Lockup Fix Enable	5	RW	TBD	ST lockup fix enable bit. 1'b0: Fix disabled 1'b1: Fix enabled
Disable Auto EEPROM reset	4	RW	Inversion of OTP Bit 142	This bit controls the nvm state machine to issue EEPROM reset. 1'b0: Issue EEROM reset sequence before any EPROM R/W access right after a HW or SW reset. 1'b1: EEPROM reset sequence is not issued automatically.
EPROM SDA_OE mode	3	RW	Inversion of OTP Bit 143	This bit controls EPROM SDA_OE generation 1'b0: SDA_OE is only disabled when the NVRAM controller is not outputting data. 1'b1: SDA_OE is disabled either when the NVRAM controller is not outputting data or when the output of SDA is HIGH. By disabling SDA_OE, the NVRAM controller relies on the pull-up to pull SDA HIGH.
Ate_mode	2	RW	0	When 1, the EEPROM Data in and data out are on 2 different pins so that we don't need to worry about the turnaround issues
NVM Access Write Enable	1	RW	0	When 1, allows the NVRAM write command to be issued even if the NVRAM write enable bit of Mode Control register is 0
NVM Access Enable	0	RW	0	When 0, prevents write access to all other NVRAM registers, except for the Software arbitration register

NVM Write1 Register (offset: 0x7028)

This register is reset by POR only.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-16	RO	0x0	–
Write Disable Command	15-8	RW	0x4h	Flash write disable command when device with protection function is used. This command will be issued by the flash interface state machine through SPI interface To flash device, and make the flash device write-disabled.
Write Enable Command	7-0	RW	0x6h	Flash write enable command when device with protection function is used. This command will be issued by the flash interface state machine through SPI interface To flash device, and make the flash device write-enabled.

Arbitration Watchdog Timer Register (offset: 0x702C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31-28	RW	1	–
Reserved	27-24	Ro	0	–
Reserved	23:8	RW	0x1000000	–
Reserved	7	–	0	–
Reserved	6	RW	0	–
Reserved	5	RW	0	–
Reserved	4-0	RO	0	–

Address Lockout Boundary Register (offset: 0x7030)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RO	X	–

Address Lockout Address Counter Debug Register (offset: 0x7034)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	X	–
Reserved	23:0	RO	X	–

NVM Auto-Sense Status Register (offset: 0x7038)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	RO	0	–
Auto Config Device ID	20:16	RO	0	DEV_AT45DB011B 8'b0000_0000 // 1Mb B part DEV_AT45DB021B 8'b0000_0001 // 2Mb B part DEV_AT45DB041B 8'b0000_0010 // 4Mb B part DEV_AT45DB081B 8'b0000_0011 // 8Mb B part DEV_AT45DB011D 8'b0001_0000 // 1Mb D part DEV_AT45DB021D 8'b0001_0001 // 2Mb D part DEV_AT45DB041D 8'b0001_0010 // 4Mb D part DEV_AT45DB081D 8'b0001_0011 // 8Mb D part DEV_AT45DB161 8'b0001_0100 // 16Mb DEV_AT45DB321 8'b0001_0101 // 32Mb DEV_STM25PE10 8'b0010_0000 // 1Mb DEV_STM25PE20 8'b0010_0001 // 2Mb DEV_STM25PE40 8'b0010_0010 // 4Mb DEV_STM25PE80 8'b0010_0011 // 8Mb DEV_STM25PE16 8'b0010_0100 // 16Mb DEV_STM45PE10 8'b0011_0000 // 1Mb DEV_STM45PE20 8'b0011_0001 // 2Mb DEV_STM45PE40 8'b0011_0010 // 4Mb DEV_STM45PE80 8'b0011_0011 // 8Mb DEV_STM45PE16 8'b0011_0100 // 16Mb DEV_MX25L100 8'b0100_0000 // 1Mb DEV_MX25L200 8'b0100_0001 // 2Mb DEV_MX25L400 8'b0100_0010 // 4Mb DEV_MX25L800 8'b0100_0011 // 8Mb DEV_MX25L160 8'b0100_0100 // 16Mb DEV_MX25L320 8'b0100_0101 // 32Mb DEV_AT45_USPT 8'b0000_1111 // Unknown DEV_AT26_USPT 8'b0001_1111 // Unknown DEV_ST25_USPT 8'b0010_1111 // Unknown DEV_ST45_USPT 8'b0011_1111 // Unknown DEV_MX25_USPT 8'b0100_1111 // Unknown Unspecified combinations are reserved.
Reserved	15:13	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Auto Config State	12:8			‘AC_IDLE: 0x0 ‘AC_AT_INT_WCSH: 0x14 ‘AC_AT_AST_CS: 0x1 ‘AC_AT_WCS_SET : 0x2 ‘AC_AT_STS_RCMD: 0x3 ‘AC_AT_RDATA: 0x4 ‘AC_AT_WCLK: 0x5 ‘AC_AT_WCS_HLD: 0x6 ‘AC_AT_DST_CS: 0x7 ‘AC_AT_WCS_MINH: 0x8 ‘AC_AT_CHK: 0x9 ‘AC_AT_AST_CS2: 0x15 ‘AC_AT_WCS_SET2: 0x16 ‘AC_AT_SIG_RCMD: 0x17 ‘AC_AT_RDATA1: 0x18 ‘AC_AT_RDATA2: 0x19 ‘AC_AT_RDATA3: 0x1A ‘AC_AT_WCLK2: 0x1B ‘AC_AT_WCS_HLD2: 0x1C ‘AC_AT_DST_CS2: 0x1D ‘AC_AT_WCS_MINH2: 0x1E ‘AC_AT_CHK2: 0x1F ‘AC_ST_AST_CS: 0xA ‘AC_ST_SIG_RCMD: 0xB ‘AC_ST_RDATA1: 0xC ‘AC_ST_RDATA2: 0xD ‘AC_ST_RDATA3: 0xE ‘AC_ST_WCLK: 0xF ‘AC_ST_DST_CS: 0x10 ‘AC_ST_WCS_MINH: 0x11 ‘AC_ST_CHK: 0x12 ‘AC_UPD_RTRY: 0x13
Reserved	7:6	RO	0	–
Auto Config Successful	5	RO	0	Auto Config logic successfully detected a Flash device.
Auto Config Enable	4	RO	Depend on strap values	Auto config feature is enabled through pin strap.
Reserved	3:1	RO	0	–
Auto_config_busy	0	RO	0	1: Auto-config FSM is busy 0: Auto-config is complete

NVRAM State Machine Status 2 Register (offset: 0x703C)

Name	*Bits*	*Access*	*Default Value*	*Description*
NVRAM Controller Idle	31	RO	1	NVRAM controller idle status 1: Idle 0: not idle
Reserved	30:20	RO	0x0	Reserved
NVRAM Access Arbitration FSM State	9:6	RO	0	NVRAM access arbitration state machine state 4'b0000: NVM_ARB_IDLE 4'b0001: NVM_ARB0 4'b0010: NVM_ARB1 4'b0011: NVM_ARB2 4'b0100: NVM_ARB3 4'b0101: NVM_ARB4
Micronix Write FSM State	4:0	RO	0	Micronix write state machine state: 3'b000: MX_WR_IDLE 3'b001: MX_WR_RD_SCTR 3'b010: MX_WR_ERS_SCTR 3'b011: MX_WR_BUFFER 3'b100: MX_WR_WNEXT 3'b101: MX_WR_PGM_SCTR

BIST Registers

SRAM Test Mode Register (offset: 0x7408)

Name	*Bits*	*Access*	*Default Value*	*Description*
APE_ARB	31:30	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_5	29:25	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_4	24:20	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_3	19:15	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_2	14:10	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_1	9:4	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_0	4:0	RW	0	SRAM Test Mode Control

SRAM Test Mode Register (offset: 0x740c)

Name	*Bits*	*Access*	*Default Value*	*Description*
APE_ATB	31:30	RW	0	SRAM Test Mode Control
APE_Shared_Mem	29:20	RW	0	SRAM Test Mode Control
APE_OLS	19:15	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_7	9:4	RW	0	SRAM Test Mode Control
APE_Scratch_Pad_6	4:0	RW	0	SRAM Test Mode Control

SRAM Test Mode Register (offset: 0x7410)

Name	*Bits*	*Access*	*Default Value*	*Description*
EMAC_DECR_FIFO_0	31:30	RW	0	SRAM Test Mode Control
MA_RX_MBUF	29:20	RW	0	SRAM Test Mode Control
PCIE_Replay	19:15	RW	0	SRAM Test Mode Control
CPU_Scratch_Pad	14:10	RW	0	SRAM Test Mode Control
Boot_ROM	9:0	RW	0	SRAM Test Mode Control

SRAM Test Mode Register (offset: 0x7414)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RW	0	–
EMAC_TX_FIFO	25:24	RW	0	SRAM Test Mode Control

Name	*Bits*	*Access*	*Default Value*	*Description*
EMAC_RX_FIFO	23:22	RW	0	SRAM Test Mode Control
EMAC_DECR_FIFO_1	21:20	RW	0	SRAM Test Mode Control
APE_ROM	19:10	RW	0	SRAM Test Mode Control
MA_TX_MBUF	9:0	RW	0	SRAM Test Mode Control

SRAM Test Mode Register (offset: 0x7418)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:0	RW	0	SRAM Test Mode Control

OTP Registers

OTP Mode Register (offset: 0x7500)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	–
Mode	0	RW	0	Set to 1 to control OTP access via GRC access. Default is control through the JTAG interface.

OTP Control Register (offset: 0x7504)

Name	*Bits*	*Access*	*Default Value*	*Description*
bypass_otp_clk	31	RW	0	This bit, if set, replaces i_jtag_otp_clk by i_jtag_ser_clk throughout the JTAG_OTP IP. This control bit affects only CPU transactions. USE_TCK_IN_CPU_MODE define should be used for this bit to be effective.
Reserved	30:29	RW	0	Reserved
cpu_debug_sel	28:25	RW	0	Select signal used to define if data or debug signals are sent to the o_otp_cpu_data output.
Burst_stat_sel	24	RW	0	Used during word program command. 0-data from the OTP memory is sent out, 1=FAIL returned for programmed bits is sent out (used to debug the failed bits during burst program)
Access_mode	23:22	RW	0	00 = Raw. This mode is used for all commands except for prog_bit and prog_word command 01 = manufacturing area 10 = configuration bits area 11 = ram repair bits area (not supported) The modes 01, 10, and 11 are used during prog_bit and prog_word commands
Otp_prog_en	21	RW	0	Only when otp_prog_en is 1, otp programming from cpu side is enabled. Set this pin to 0 will disable otp programming from the cpu side. This bit is usually driven by chip pin to prevent otp programming from cpu side when chip ship to customer
Otp_debug_mode	20	RW	0	Causes bits [18:5] to be used for direct OTP control – used by the library group for debugging. The bits [18:5] are control bits of the OTP memory that can be programmed using software (CPU interface)

Name	*Bits*	*Access*	*Default Value*	*Description*
Wrp_continue_on_fail	19	RW	0	0 = OTP will hang jtag_wrapper module on fail – stay in PROG_FAIL state forever OR until reset is applied (or this bit is made 1). 1 = OTP will set the FAIL status and continue to accept commands. Please note that even if this bit is 0, the jtag_otp module as such will not hang and the status indicates that there was programming failure. It is only the wrapper module that hangs. Command_done signal will be asserted even in this condition.
Wrp_time_margin	18:16	RW	0	Timing margin – debug only (bit [20] of this register must be set in order to see the effect of these bits)
Wrp_sadbyp	15	RW	0	Sadbpy: senseAmp delay bypass – debug only (bit [20] of this register must be set in order to see the effect of these bits)
Unused	14	RW	0	Unused
wrp_pbyp	13	RW	0	OTP cell clock control – debug only (bit [20] of this register must be set in order to see the effect of these bits)
Wrp_pcount	12:10	RW	0	Program pulse count – debug only (bit [20] of this register must be set in order to see the effect of these bits)
Wrp_vsel	9:6	RW	0	Verify level select – debug only (bit [20] of this register must be set in order to see the effect of these bits)
Wrp_prog_sel	5	RW	0	Program select – debug only, selects which of the 2 analog bit cells to burn (bit [20] of this register must be set in order to see the effect of these bits)
Command	4:1	RW	0	0: Read 1: Program Bit internal clk 2: Program Word 3: Verify 4: Init 5: Set 6: Reset 7: OCST 8: Lock 9: Prescreen test 10: Program bit External clock mode – field programming mode 11: Program Word external clock
Start	0	RW	0	Rising edge of the signal will execute OTP command

OTP Status Register (offset: 0x7508)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:12	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
control_err	11	RO	–	The control module state machine went into an error state. Else returns 0.
wrp_error	10	RO	–	The wrapper module state machine went into an error state. Else returns 0.
invalid_command	9	RO	–	An invalid command was issued.
otp_stby_reg	8	RO	–	The OTP cell is in standby state. Applicable only when the OTP_STANDBY define is used. Else returns 0.
Init_wait_done	7	RO	–	Can be used to indicate when the OTP is out of RESET. Also if this bit is not a 1, there is no clock to the block OR the block is still in RESET.
Prog_blocked	6	RO	–	The row is locked and the programming attempt was not allowed
Invalid_prog_req	5	RO	–	A program request was sent with the incorrect access mode and 12 bit address settings
Wrp_fail	4	RO	–	A FAIL was returned from the wrapper logic. The part is bad
Wrp_busy	3	RO	–	A wrp_busy was seen – your request was started. Check command_done (bit 0) to see when it is complete
Wrp_dout	2	RO	–	This is the single bit data pointed to by the 12 bit address
Wrp_data_read	1	RO	–	The wrp_dout is now valid to read
Command_done	0	RO	–	This bit is set when the state machine has returned to IDLE. This will not be set until the last bit of a WORD program is complete

OTP Address Register (offset: 0x750C)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	RO	0	–
Address	15:0	RW	0	This 16 bit CPU OTP address register provides up to 8K address space for OTP program, read access from CPU interface. In prog_word command, it determines the address of the entire memory row to be programmed or the data to be read.

OTP Write Data Register (offset: 0x7510)

Name	*Bits*	*Access*	*Default Value*	*Description*
Write Data	31:0	RW	0	The 32 bit CPU OTP write data register is used to provide write data with burst write command from CPU side. Different than the burst write from JTAG, cpu interface only supports 32 bit burst write data. This is to reduce the number of bits needed to wire to cpu interface at top level.

OTP Read Data Register (offset: 0x7514)

Name	*Bits*	*Access*	*Default Value*	*Description*
Read Data	31:0	RO	–	The 32 bit CPU OTP read data register outputs the 32 bits read data from read command of CPU interface

OTP Soft Reset Register (offset: 0x7518)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:1	RO	0	–
Reset	0	RW	0	A value of 1 will assert the OTP soft reset

Memory Address Map

Table 105: Memory Address Map

Region	*Size*	*NIC CPU View*	*Host Flat View*	*Host Standard View*	*Host UNDI View*
Unmapped	256B	0x00000000- 0x000000FF	0x01000000- 0x010000FF	0x00000000-* 0x000000FF	0x00000000-* 0x000000FF
Send Ring 0 RCB	12B	0x00000100- 0x0000010B	0x01000100- 0x0100010B	0x00000100-* 0x0000010B	0x00000100-* 0x0000010B
Unmapped	4B	0x0000010C- 0x0000010F	0x0100010C- 0x0100010F	0x0000010C- 0x0000010F	0x0000010C- 0x0000010F
Send Ring 1 RCB	12B	0x00000110- 0x0000011B	0x01000110- 0x0100011B	0x00000110-* 0x0000011B	0x00000110-* 0x0000011B
Unmapped	4B	0x0000011C- 0x0000011F	0x0100011C- 0x0100011F	0x0000011C- 0x0000011F	0x0000011C- 0x0000011F
Unmapped	224B	0x00000110- 0x000001FF	0x01000110- 0x010001FF	0x00000110- 0x000001FF	0x00000110- 0x000001FF
Receive Return Ring 0 RCB	8B	0x00000200- 0x00000207	0x01000200- 0x01000207	0x00000200-* 0x00000207	0x00000200-* 0x00000207
Unmapped	8B	0x00000208- 0x0000020F	0x01000208- 0x0100020F	0x00000208-* 0x0000020F	0x00000208-* 0x0000020F
Receive Return Ring 1 RCB	8B	0x00000210- 0x00000217	0x01000210- 0x01000217	0x00000210-* 0x00000217	0x00000210-* 0x00000217
Unmapped	8B	0x00000218- 0x0000021F	0x01000218- 0x0100021F	0x00000218-* 0x0000021F	0x00000218-* 0x0000021F
Receive Return Ring 2 RCB	8B	0x00000220- 0x00000227	0x01000220- 0x01000227	0x00000220-* 0x00000227	0x00000220-* 0x00000227
Unmapped	8B	0x00000228- 0x0000022F	0x01000228- 0x0100022F	0x00000228-* 0x0000022F	0x00000228-* 0x0000022F
Receive Return Ring 3 RCB	8B	0x00000230- 0x00000237	0x01000230- 0x01000237	0x00000230-* 0x00000237	0x00000230-* 0x00000237
Unmapped	8B	0x00000238- 0x0000023F	0x01000238- 0x0100023F	0x00000238-* 0x0000023F	0x00000238-* 0x0000023F
Unmapped	2.3KB	0x00000240- 0x00000B4F	0x01000240- 0x01000B4F	0x00000240- 0x00000B4F	0x00000240- 0x00000B4F
Software GenComware Gencomm	1KB	0x00000B50- 0x00000F4F	0x01000B50- 0x01000F4F	0x00000B50-* 0x00000F4F	0x00000B50-* 0x00000F4F
Unmapped	4KB	0x00000F50- 0x00001FFF	0x01000F50- 0x01001FFF	0x00000F50- 0x00001FFF	0x00000F50- 0x00001FFF

Table 105: Memory Address Map (Cont.)

Region	Size	NIC CPU View	Host Flat View	Host Standard View	Host UNDI View
Unmapped	8KB	0x00002000- 0x00003FFF	0x01002000- 0x01003FFF	0x00002000- 0x00003FFF	0x00002000- 0x00003FFF
Send Ring 0	1KB	0x00004000- 0x000043FF	0x01004000- 0x010043FF	0x00004000-* 0x000043FF	0x00004000-* 0x000043FF
Send Ring 1	1KB	0x00004400- 0x000047FF	0x01004400- 0x010047FF	0x00004400-* 0x000047FF	0x00004400-* 0x000047FF
Unmapped	6KB	0x00004800- 0x00005FFF	0x01004800- 0x01005FFF	0x00004800- 0x00005FFF	0x00004800- 0x00005FFF
Standard Receive Ring	4KB	0x00006000- 0x00006FFF	0x01006000- 0x01006FFF	0x00006000-* 0x00006FFF	0x00006000-* 0x00006FFF
Jumbo Receive Ring	4KB	0x00007000- 0x00007FFF	0x01007000- 0x01007FFF	0x00007000-* 0x00007FFF	0x00007000-* 0x00007FFF
TXMBUF 0	22KB	0x00008000- 0x0000D7FF	0x01008000- 0x0100D7FF	0x00008000-* 0x0000D7FF	0x00008000-* 0x0000D7FF
TXMBUF 1	8KB	0x0000D800- 0x0000F7FF	0x0100D800- 0x0100F7FF	0x0000D800- 0x0000F7FF	0x0000D800- 0x0000F7FF
Unmapped	2KB	0x0000F800- 0x0000FFFF	0x0100F800- 0x0100FFFF	0x0000F800- 0x0000FFFF	0x0000F800- 0x0000FFFF
RX MBUFRXMBUF	40KB	0x00010000- 0x00019FFF	0x01010000- 0x01019FFF	0x00010000-* 0x00019FFF	0x00010000-* 0x00019FFF
Unmapped	15M+920KB	0x0001A000- 0x00FFFFFF	0x0101A000- 0x01FFFFFF	0x0001A000- 0x00FFFFFF	0x0001A000- 0x00FFFFFF
RXCPU Scratch Pad	64KB	0x08000000- 0x0800FFFF	–	–	–
RXCPU ROM	1.328KB	0x40000000- 0x4000054F	–	–	–
RXCPU SB ROM	18.672KB	0x40000550- 0x40004FFF	–	–	–
PCI Configuration	256B	0xC0000000- 0xC00000FF	0x00000000- 0x000000FF	0x00000000- 0x000000FF	0x00000000- 0x000000FF
High Priority Mailbox	512B	–	0x00000200- 0x000003FF	0x00000200- 0x000003FF	0x00005800-** 0x000059FF
Functional Registers	31KB	0xC0000400- 0xC0007FFF	0x00000400- 0x00007FFF	0x00000400- 0x00007FFF	0x00000100-** 0x00007FFF
Mailboxes	1MB	–	0x00100000- 0x001FFFFF	–	–
RXCPU ROM+SB ROM Slave Access	20KB	0xC0020000- 0xC0024FFF	0xC0020000-** 0xC0024FFF	0xC0020000-** 0xC0024FFF	0xC0020000-** 0xC0024FFF

Table 105: Memory Address Map (Cont.)

Region	Size	NIC CPU View	Host Flat View	Host Standard View	Host UNDI View
Unmapped	44KB	0xC0025000- 0xC002FFFF	0xC0025000-** 0xC002FFFF	0xC0025000-** 0xC002FFFF	0xC0025000-** 0xC002FFFF
RXCPU Scratch Pad Slave Access	64KB	0xC0030000- 0xC003FFFF	0xC0030000-** 0xC003FFFF	0xC0030000-** 0xC003FFFF	0xC0030000-** 0xC003FFFF
Unmapped	64KB	0xC0040000- 0xC004FFFF	0xC0040000-** 0xC004FFFF	0xC0040000-** 0xC004FFFF	0xC0040000-** 0xC004FFFF

Section 13: Transceiver Registers

Purpose

This section describes the MII registers of the integrated 10/100/1000T PHY transceiver. The access to the transceiver registers is provided indirectly through the MII Communication register (see "MII Communication Register (offset: 0x44C)" on page 327) of the MAC. The transceiver registers are accessed with the PHY_Addr bit of the MII Communication register set to 0x1. The integrated transceiver contains the set of registers shown in the tables below.

Register Field Access Type

R/W = read/write	H = forced high
RO = read only	L = forced low
LH = latches high value (until read)	SC = self-clearing
LL = latches low value (until read)	CR = clear on read

Transceiver Register Map

Address	*Name*
00h	MII_Control_Register
01h	MII_Status_Register
02h	PHY_Identifier_MSB_Register
03h	PHY_Identifier_LSB_Register
04h	Auto_Negot_Advertisement_Register
05h	Auto_Negot_Link_Partner_Ability_Base_Pg_Register
06h	Auto_Negot_Expansion_Register
07h	Auto_Negot_Next_Page_Transmit_Register
08h	Auto_Negot_Link_Partner_Ability_Nxt_Pg_Register
09h	1000Base_T_Control_Register
0Ah	1000Base_T_Status_Register
0Bh–0Eh	Reserved_by_IEEE

Address	*Name*
0Fh	IEEE_Extended_Status_Register
10h	PHY_Extended_Control_Register
11h	PHY_Extended_Status_Register
12h	Receive_Error_Counter_Register
13h	False_Carrier_Sense_Counter_Register
14h	Local_Remote_Rcvr_NOT_OK_Counters_Register
15h	DSP_Coefficient_Read_Write_Port_Register
16h	DSP_Control_Register
17h	DSP_Coefficient_Address_Register
18h	Auxiliary_Control_Register Shadow Registers: 001 => 10 BASE-T 010 => Power Control 011 => IP Phone 100 => Misc Test 101 => Misc Test 2 110 => Manual IP Phone seed 111 => Misc Control
19h	Auxiliary_Status_Register
1Ah	Interrupt_Status_Register
1Bh	Interrupt_Mask_Register

Address	***Name***
1Ch	Miscellaneous_Shadow_Registers: 00000 => Cabletron LED modes 00001 => DLL Control 00010 => Spare Control 1 00011 => Clock Aligner 00100 => Spare Control 2 00101 => Spare Control 3 00110 => TDR Control 1 00111 => TDR Control 2 01000 => Led Status 01001 => Led Control 01010 => Auto-Power Down 01011 => External Control 1 01100 => External Control 2 01101 => LED Selector 1 01110 => LED Selector 2 01111 => LED GPIO Control/Status 10000 => Reserved 10001 => SerDes 100-FX Status 10010 => SerDes 100-FX Test 10011 => SerDes 100-FX Control 10100 => External SerDes Control 10101 => Sgmii Slave Control 10110 => Misc 1000X Control 2 10111 => Misc 1000X Control 11000 => Auto-Detect SGMII/GBIC 11001 => Test 1000X 11010 => Autoneg 1000X Debug 11011 => Auxiliary 1000X Control 11100 => Auxiliary 1000X Status 11101 => Misc 1000X Status 11110 => Auto-Detect Medium 11111 => Mode Control
1Dh	Master_Slave_Seed_Register Shadow Register: 1 => HCD Status
1Eh	Test1_Register
1Fh	Test2_Register

00h–0Fh 10/100/1000T Register Map Detailed Description

To access the following registers, make sure that the Enable 1000BASE-X Register control (Register 1Ch, Shadow 11111, bit 0) = 0.

00h: MII_Control_Register

Bit	Name	R/W	Default	Description
15	RESET	R/W SC	0	1 = PHY Reset 0 = normal operation
14	LOOPBACK	R/W	intlpbk_def	1 = loopback mode 0 = normal operation
13	SPEED_SELECT_LSB (#)	R/W	"force_speed0_def & ~force_speed1_def"	0.6, 0.13: 11 = Reserved 10 = 1000 Mbit/s 01 = 100 Mbit/s 00 = 10 Mbit/s
12	AUTONEGOTIATION_ENABLE (#)	R/W	"(an_def & ~extlpbk_def) | (mode_sel_def == 2'b11)"	1 = auto-negotiation enabled 0 = auto-negotiation disabled
11	POWER_DOWN	R/W	"(mode_sel_def == 2'b01) & ~autodet_med_pindef"	1 = low power mode 0 = normal operation
10	ISOLATE	R/W	isolate_def	1 = isolate PHY from MII 0 = normal operation
9	RESTART_AUTONEGOTIATION (#)	R/W SC	"anen & restart_anen_pin"	1 = restart auto-negotiation process 0 = normal operation
8	DUPLEX_MODE (#)	R/W	dplx_def	1 = full duplex 0 = half duplex
7	COLLISION_TEST	R/W	0	1 = collision test mode enabled 0 = collision test mode disabled
6	SPEED_SELECT_MSB (#)	R/W	force_speed1_def	0.6, 0.13: 11 = Reserved 10 = 1000 Mbit/s 01 = 100 Mbit/s 00 = 10 Mbit/s
5	UNIDIRECTIONAL_ENABLE	R/W	0	When 0.12=0 AND 0.8=1: 1 = able to transmit packets when no link 0 = requires link in order to transmit packets

Bit	Name	R/W	Default	Description
4:0	RESERVED	R/W	000000	write as 0, ignore on read
* For BCM5461 defaults to 1 when PHYA[4:0] = 00000 and mode_sel[1] = 0 # registers updated with pin defaults on rising edge of restart autoneg pin (tied to gnd at top level)				

01h: MII_Status_Register

Bit	Name	R/W	Default	Description
15	100BASE_T4_CAPABLE	RO L	0	1 = 100BASE-T4 capable 0 = not 100BASE-T4 capable
14	100BASE_X_FULL_DUPLEX_ CAPABLE	RO H	1	1 = 100BASE-X full duplex capable 0 = not 100BASE-X full duplex capable
13	100BASE_X_HALF_DUPLEX_ CAPABLE	RO H	1	1 = 100BASE-X half duplex capable 0 = not 100BASE-X half duplex capable
12	10BASE_T_FULL_DPLEX_CAPABLE	RO H	1	1 = 10BASE-T full duplex capable 0 = not 10BASE-T full duplex capable
11	10BASE_T_HALF_DPLEX_CAPABLE	RO H	1	1 = 10BASE-T half duplex capable 0 = not 10BASE-T half duplex capable
10	100BASE_T2_FULL_DUPLEX_ CAPABLE	RO L	0	1 = 100BASE-T2 full duplex capable 0 = not 100BASE-T2 full duplex capable
9	100BASE_T2_HALF_DUPLEX_ CAPABLE	RO L	0	1 = 100BASE-T2 half duplex capable 0 = not 100BASE-T2 half duplex capable
8	EXTENDED_STATUS	RO H	1	1 = extended status information in register 0Fh 0 = no extended status info in register 0Fh
7	UNIDIRECTIONAL_CAPABLE	RO	1	1 = capable of Unidirectional Transmit
6	MF_PREAMBLE_SUPPRESSION	RO H	1	1 = PHY will accept management frames with preamble suppressed 0 = PHY will not accept management frames with preamble suppressed
5	AUTO_NEGOTIATION_COMPLETE	RO	0	1 = auto-negotiation complete 0 = auto-negotiation in progress
4	REMOTE_FAULT	RO LH	0	1 = remote fault detected 0 = no remote fault detected
3	AUTO_NEGOTIATION_ABILITY	RO H	1	1 = auto-negotiation capable 0 = not auto-negotiation capable
2	LINK_STATUS	RO LL	0	1 = link pass 0 = link fail

Bit	Name	R/W	Default	Description
1	JABBER_DETECT	RO LH	0	1 = jabber condition detected 0 = no jabber condition detected
0	EXTENDED_CAPABILITY	RO H	1	1 = extended register capabilities supported 0 = basic register set capabilities only

02h: PHY_Identifier_MSB_Register

Bit	Name	R/W	Default	Description
15:0	OUI_MSB	RO	600 dh	Bits 3:18 of organizationally unique identifier

03h: PHY_Identifier_LSB_Register

Bit	Name	R/W	Default	Description
15:10	OUI_LSB	RO	100001	Bits 19:24 of organizationally unique identifier
9:4	MODEL	RO	111000	Device model number (metal programmable)
3:0	REVISION	RO	0000	Device revision number (metal programmable)

04h: Auto_Negot_Advertisement_Register

Bit	Name	R/W	Default	Description
15	NXT_PAGE	R/W	0	1 = next page ability supported 0 = next page ability not supported
14	RESERVED	R/W	0	write as 0, ignore on read
13	REMOTE FAULT	R/W	0	1 = advertise remote fault detected 0 = advertise no remote fault detected
12	RESERVED	R/W	0	write as 0, ignore on read
11	ASYMMETRIC_PAUSE	R/W	asym_pause_def	1 = Advertise asymmetric pause 0 = Advertise no asymmetric pause
10	PAUSABLE	R/W	pause_def	1 = capable of full duplex Pause operation 0 = not capable of Pause operation
9	100BASET4_CAPABLE	R/W	0	1 = 100BASE-T4 capable 0 = not 100BASE-T4 capable

Bit	*Name*	*R/W*	*Default*	*Description*
8	100BASETX_FULL_ DUPLEX_CAPABLE (#)	R/W	"dplx_def & (f1000 | spd0) & ~((mode_sel_def == 2'b11) | (f1000 & spd0) | (en10b & ~mode_sel_def[1]))"	1 = 100BASE-TX full duplex capable 0 = not 100BASE-TX full duplex capable
7	100BASETX_HALF_ DUPLEX_CAPABLE (#)	R/W	"adv_hdx_def & (f1000 | spd0) & ~((mode_sel_def == 2'b11) | (f1000 & spd0) | (en10b & ~mode_sel[1]))"	1 = 100BASE-TX capable 0 = not 100BASE-TX capable
6	10BASET_FULL_DUPLEX_ CAPABLE (#)	R/W	"dplx_def & ~((mode_sel_def == 2'b11) | (f1000 & spd0) | (en10b & ~mode_sel[1]))"	1 = 10BASE-T full duplex capable 0 = not 10BASE-T full duplex capable
5	10BASET_HALF_DUPLEX_ CAPABLE (#)	R/W	"adv_hdx_def & ~((mode_sel_def == 2'b11) | (f1000 & spd0) | (en10b & ~mode_sel[1]))"	1 = 10BASE-T half duplex capable 0 = not 10BASE-T half duplex capable
4:0	PROTOCOL_SELECT	R/W	00001	00001 = IEEE 802.3 CSMA/CD

# registers updated with pin defaults on rising edge of restart autoneg pin (tied to gnd at top level)

05h: Auto_Negot_Link_Partner_Abfility_Base_Pg_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15	NEXT_PAGE	RO	0	1 = link partner is next page able 0 = link partner is not next page able
14	ACKNOWLEDGE1	RO	0	1 = link partner has received link code word 0 = link partner has not received link code word
13	REMOTE_FAULT	RO*	0	1 = link partner has detected remote fault 0 = link partner has not detected remote fault
12	RESERVED	RO*	0	write as 0, ignore on read
11	LINK_PARTNER_ASYMMETRIC_PAUSE	RO*	0	link partner's asymmetric pause bit
10	PAUSE_CAPABLE	RO*	0	1 = link partner is capable of Pause operation 0 = link partner not capable of Pause operation
9	100BASE_T4_CAPABLE	RO*	0	1 = link partner is 100BASE-T4 capable 0 = link partner is not 100BASE-T4 capable
8	100BASE_TX_FULL_DUPLEX_CAPABLE	RO*	0	1 = link partner is 100BASE-TX full duplex capable 0 = link partner is not 100BASE-TX full duplex capable
7	100BASE_TX_HALF_DUPLEX_CAPABLE	RO*	0	1 = link partner is 100BASE-TX half duplex capable 0 = link partner is not 100BASE-TX half duplex capable
6	10BASE_T_FULL_DUPLEX_CAPABLE	RO*	0	1 = link partner is 10BASE-T full duplex capable 0 = link partner is not 10BASE-T full duplex capable
5	10BASE_T_HALF_DUPLEX_CAPABLE	RO*	0	1 = link partner is 10BASE-T half duplex capable 0 = link partner is not 10BASE-T half duplex capable
4:0	PROTOCOL_SELECTOR	RO*	00000	link partner's protocol selector (see IEEE spec for encodings)
* R/W when "writeable link partner ability test mode" (reg 1Fh bit 10) is set				

06h: Auto_Negot_Expansion_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15:7	RESERVED	RO	000h	ignore on read
6	NEXT_PAGE_RECEIVE_LOCATION_ ABLE	RO H	1	1 = register 6.5 determines next page receive location 0 = register 6.5 does not determine next page receive location
5	NEXT_PAGE_RECEIVE_LOCATION	RO H	1	1 = next pages stored in register 8 0 = next pages stored in register 5
4	PARALLEL_DETECTION_FAULT	RO LH	0	1 = parallel detection fault 0 = no parallel detection fault
3	LINK_PARTNER_NEXT_PAGE_ABILITY	RO	0	1 = link partner is next page able 0 = link partner is not next page able
2	NEXT_PAGE_ABILITY	RO H	1	1 = local device is next page able 0 = local device is not next page able
1	PAGE_RECEIVED	RO LH	0	1 = new link code word has been received 0 = new link code word has not been received
0	LINK_PARTNER_AUTONEG_ABILITY	RO	0	1 = link partner is auto-negotiation able 0 = link partner is not auto-negotiation able

07h: Auto_Negot_Next_Page_Transmit_Register (Software Controlled Next Pages)

Bit	*Name*	*R/W*	*Default*	*Description*
15	NXT_PG	R/W	0	1 = additional next pages will follow 0 = sending last page
14	RESERVED	RO	0	write as 0, ignore on read
13	MESSAGE_PAGE	R/W	1	1 = message page 0 = unformatted page
12	ACKNOWLEDGE 2	R/W	0	1 = will comply with message (not used during 1000BASE-T next pages) 0 = cannot comply with message
11	TOGGLE1	RO	0	toggled by arbitration state machine during next page exchange - write as 0, ignore on read
10:0	CODE_FIELD	R/W	0000000001	message code field or unformatted code field

08h: Auto_Negot_Link_Partner_Ability_Nxt_Pg_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15	NEXT_PG	RO	0	1 = additional next pages will follow 0 = sending last page
14	ACKNOWLEDGE3	RO	0	1 = acknowledge 0 = no acknowledge
13	MESSAGE_PG	RO	0	1 = message page 0 = unformatted page
12	ACKNOWLEDGE_2	RO	0	1 = will comply with message (not used during 1000BASE-T next pages) 0 = cannot comply with message
11	TOGGLE2	RO	0	1 = sent 0 during previous Link Code Word 0 = sent 1 during previous Link Code Word
10:0	CODEFIELD	RO*	0	Message code field or unformatted code field
* R/W when "writeable link partner ability test mode" (reg 1Fh bit 10) is set When reg 1Fh bit 8 is set, the link partner random seed can be written in bits 10:0 (write only)				

09h: 1000Base_T_Control_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15:13	TEST_MODE	R/W	000	1xx = Test Mode 4 011 = Test Mode 3 010 = Test Mode 2 001 = Test Mode 1 000 = Normal Operation
12	MASTER_SLAVE_CONFIG_ ENABLE (#)	R/W	manms_def	1 = enable Master/Slave manual config value 0 = disable Master/Slave manual config value
11	MASTER_SLAVE_CONFIG_ VALUE (#)	R/W	hub_def	1 = configure PHY as Master when 9.12 is set 0 = configure PHY as Slave when 9.12 is set
10	REPEATER_DTE (#)	R/W	hub_def	1 = Repeater/switch device port 0 = DTE device port
9	ADV_1000BASE_T_FULL_ DUPLEX (#)	R/W	"dplx_def & (f1000 | (mode_sel == 2'b11))"	1 = Advertise 1000BASE-T full duplex capable 0 = Advertise not 1000BASE-T full duplex capable

Bit	*Name*	*R/W*	*Default*	*Description*
8	ADV_1000BASE_T_HALF_DUPLEX (#)	R/W	"r09_adv_hdx_def & (f1000 | (mode_sel == 2'b11))"	1 = Advertise 1000BASE-T half duplex capable 0 = Advertise not 1000BASE-T half duplex capable
7:0	RESERVED	R/W	00000000	write as 0, ignore on read
# registers updated with pin defaults on rising edge of restart autoneg pin (tied to gnd at top level)				

0Ah: 1000Base_T_Status_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15	MASTER_SLAVE_CONFIG_FAULT	RO LH (#)	0	1 = Master/Slave configuration fault detected 0 = no Master/Slave configuration fault detected (cleared by restart_an, an_complete or reg read)
14	MASTER_SLAVE_CONFIG_RESOLUTION	RO	0	1 = local PHY configured as Master 0 = local PHY configured as Slave
13	LOCAL_RECEIVER_STATUS	RO	0	1 = local receiver status OK 0 = local receiver status not OK
12	REMOTE_RECEIVER_STATUS	RO	0	1 = remote receiver status OK 0 = remote receiver status not OK
11	LINK_PARTNER_1000BASE-T_FULL_DUPLEX_CAPABLE	RO*	0	1 = link partner is 1000BASE-T full duplex capable 0 = link partner is not 1000BASE-T full duplex capable
10	LINK_PARTNER_1000BASE_T_HALF_DUPLEX_CAPABLE	RO*	0	1 = link partner is 1000BASE-T half duplex capable 0 = link partner is not 1000BASE-T half duplex capable
9:8	RESERVED	RO	00h	ignore on read
7:0	IDLE_ERROR_COUNT	RO CR	00h	Number of idle errors since last read
(#)=not LH & *=R/W when "writeable link partner ability test mode" (reg 1Fh bit 10) is set				

0Eh: BroadReach LRE Access Register

Bit	*Name*	*R/W*	*Default*	*Description*
15:3	RESERVED	RO	000h	Ignore on Read
2	ENABLE LRE REGISTER ACCESS OVERRIDE	R/W	0	1 = Allow access to BroadReach LRE Registers 0 = BroadReach LRE Registers enabled only by normal operation
1	LRE REGISTER OVERRIDE VALUE	R/W	0	1 = Force Access to IEEE Registers (only available when 0E.2 = 1) 0 = Force Access to BroadReach LRE Registers (only available when 0E.2 = 1)
0	LRE REGISTER ACCESS STATUS	RO	0	1 = BroadReach LRE Registers are currently accessible 0 = IEEE Registers are currently accessible

0Fh: IEEE_Extended_Status_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15	1000BASE_X_FULL_DUPLEX_ CAPABLE	RO L	0	1 = 1000BASE-X full duplex capable 0 = not 1000BASE-X full duplex capable
14	1000BASE_X_HALF_DUPLEX_ CAPABLE	RO L	0	1 = 1000BASE-X half duplex capable 0 = not 1000BASE-X half duplex capable
13	1000BASE_T_FULL_DUPLEX_ CAPABLE	RO H	1	1 = 1000BASE-T full duplex capable 0 = not 1000BASE-T full duplex capable
12	1000BASE_T_HALF_DUPLEX_ CAPABLE	RO H	1	1 = 1000BASE-T half duplex capable 0 = not 1000BASE-T half duplex capable
11:0	RESERVED	RO	000h	ignore on read

10h–15h Register Map Detailed Description

10h: PHY_Extended_Control_Register

Bit	*Name*	*R/W*	*Default*	*Description*
15	MAC_PHY_INTERFACE_MODE	R/W	"en10b_def & ~mode_sel_def[1] & ~autodet_med_pindef"	1 = 10B interface mode 0 = GMII mode
14	DISABLE_AUTOMATIC_MDI_CROSSOVER	R/W	mdix_disable_def	1 = automatic MDI crossover disabled 0 = automatic MDI crossover enabled
13	TRANSMIT_DISABLE	R/W	0	1 = force transmit output to high impedance 0 = normal operation
12	INTERRUPT_DISABLE	R/W	0	1 = interrupts disabled 0 = interrupts enabled
11	FORCE_INTERRUPT	R/W	0	1 = force interrupt status to "active" 0 = normal interrupt operation
10	BYPASS_ENCODER	R/W	0	1 = bypass 4B5B encoder and decoder 0 = normal operation
9	BYPASS_SCRAMBLER	R/W	0	1 = bypass scrambler and descrambler 0 = normal operation
8	BYPASS_NRZI_MLT3	R/W	0	1 = bypass NRZI/MLT3 encoder and decoder 0 = normal operation
7	BYPASS_ALIGNMENT	R/W	0	1 = bypass receive symbol alignment 0 = normal operation
6	RESET_SCRAMBLER	R/W SC	0	1 = reset scrambler to all 1's state 0 = normal scrambler operation
5	ENABLE_LED_TRAFFIC_MODE	R/W	0	1 = LED traffic mode enabled 0 = LED traffic mode disabled
4	FORCE_LEDS_ON	R/W	0	1 = force all LED's into "ON" state 0 = normal LED operation

Bit	*Name*	*R/W*	*Default*	*Description*
3	FORCE_LEDS_OFF	R/W	0	1 = force all LED's into "OFF" state 0 = normal LED operation
2	BLOCK_TXEN_MODE	R/W	0	1 = extend transmit IPG's to at least 4 nibbles in 100BASE-TX mode 0 = do not extend short transmit IPG's
1	UNIDIRECTIONAL_ENABLE	R/W	0	1 = able to transmit packets when no link 0 = requires link in order to transmit packets (similar to IEEE register 0, bit 5)
0	GMII_RGMII_FIFO_ELASTICITY[0] # in copper mode or rgmii 100fx on serdes #pads #modes: #(RGMII 10/100/1000 copper mode or GMII #1000 copper mode: #from gmii/rgmii pads to #copper link partner) #(RGMII 100fx: #from rgmii pads to serdes link partner)	R/W	0	GMII/RGMII FIFO ELASTICITY [1:0] in copper mode or rgmii 100fx on SerDes pads 11 = Support 20k byte packets (18k 1G+) 10 = support 15k byte packets 01 = support 10k byte packets 00 = support 5k byte packets (w/ 200ppm offset) MSB is located at expansion reg 46[14]

11h: PHY_Extended_Status_Register (copper side only)

Bit	*Name*	*R/W*	*Default*	*Description*
15	AUTONEG_BASE_PG_SELECTOR_FIELD_ MISMATCH	RO LH	0	1 = link partner base page selector field mismatched advertised selector field since last read 0 = no mismatch detected since last read
14	WIRESPEED_DOWNGRADE	RO	0	1 = autoneg advertising downgraded 0 = autoneg advertised as shown in regs 04h & 09h
13	MDI_CROSSOVER_STATE	RO	0	1 = MDIX 0 = MDI

Bit	*Name*	*R/W*	*Default*	*Description*
12	INTERRUPT_STATUS	RO	0	1 = unmasked interrupt currently active 0 = interrupts clear
11	REMOTE_RECEIVER_STA TUS	RO LL	0	1 = remote receiver status OK 0 = remote receiver status not OK
10	LOCAL_RECEIVER_STATUS	RO LL	0	1 = local receiver status OK 0 = local receiver status not OK
9	LOCKED	RO	0	1 = descrambler locked 0 = descrambler unlocked
8	LINK_STATUS	RO	0	1 = link pass 0 = link fail
7	CRC_ERROR_DETECTED	RO LH	0	1 = CRC error detected since last read 0 = no CRC error detected since last read
6	CARRIER_EXTENSION_ERROR_DETECTED	RO LH	0	1 = carrier ext. error detected since last read 0 = no carrier ext. error detected since last read
5	BAD_SSD_DETECTED (FALSE CARRIER)	RO LH	0	1 = bad SSD error detected since last read 0 = no bad SSD error detected since last read
4	BAD_ESD_DETECTED (PREMATURE END)	RO LH	0	1 = bad ESD error detected since last read 0 = no bad ESD error detected since last read
3	RECEIVE_ERROR_DETECTED	RO LH	0	1 = receive coding error detected since last read 0 = no receive error detected since last read
2	TRANSMIT_ERROR_DETECTED	RO LH	0	1 = transmit error code detected since last read 0 = no transmit error detected since last read
1	LOCK_ERROR_DETECTED	RO LH	0	1 = lock error detected since last read 0 = no lock error detected since last read
0	MLT3_CODE_ERROR_DETECTED	RO LH	0	1 = MLT3 code error detected since last read 0 = no MLT3 error detected since last read

12h: Receive_Error_Counter_Register

Bit	Name	R/W	Default	Description
15:0	RECEIVE_ERROR_COUNTER	R/W CR	0000h	Number of non-collision packets with receive errors since last read. Freezes at FFFFh. (Counts SerDes errors when register 1ch shadow "11011" bit 9 = 1 otherwise copper errors)

13h: False_Carrier_Sense_Counter_Register

Bit	Name	R/W	Default	Description
15:8	SERDES_BER_COUNTER	RO	00h	Number of invalid code groups received while sync_status = 1 since last cleared. Cleared by writing expansion register 4D bit 15 = 1
7:0	FALSE_CARRIER_SENSE_COUNTER # (TX ERROR COUNTER)	R/W CR	00h	Number of false carrier sense events since last read. Counts packets received with transmit error codes when TXERVIS bit in test register is set. Freezes at FFh. (Counts SerDes errors when register 1ch shadow "11011" bit 9 = 1 otherwise copper errors)

14h: Local_Remote_Receiver_NOT_OK_Counters_Register

Bit	Name	R/W	Default	Description
15:8	LOCAL_RECEIVER_NOT_OK_COUNTER	R/W CR	00h	number of times local receiver status was not OK since last read. Freezes at FFh.
7:0	REMOTE_RECEIVER_NOT_OK_COUNTER	R/W CR	00h	number of times remote receiver status was not OK since last read. Freezes at FFh.
15:0	CRC_ERROR_COUNTER	R/W CR	0000h	when CRC error count visibility test mode is set, this reg becomes a 16 bit CRC error counter. Freezes at FFFFh. (Counts SerDes errors when register 1ch shadow "11011" bit 9 = 1 otherwise copper errors)

Make 100BASE-TX local receiver status signal and front end reset that works more like 1000BASE-T (no drop in link unless maxwait_timer expires). Use this counter to replace the watchdog timeout count used in 5203 PHY status register.

15h: DSP Coefficient Read/Write Port (when filter select is not “0000”)

The DSP coefficient selected in the DSP control register will be mapped to this address. The register may be directly read or written via management accesses to this port.

The SM_DSP clock must be enabled (18-0.11) before writing to these registers.

When reading or writing the echo, next, or dfe filters some overhead is required:

- There is an internal RAM located between the filter and the serial management interface. The serial management interface does not interface to the filter directly and can only be accessed with the macro operations located in register 16h [12, 9:4].
- Read/writes to register 15h access the RAM only. The upper word select is toggled every other access. The tap address is the ram address currently accessed. It increments every other access automatically.
- The busy bit should be polled to guarantee the operation is complete before continuing.

DFE, ECHO, or NEXT FILTER COEFFICIENT READ/WRITE: (all filter select and all channel select ignored) UPPER WORD SELECT = “0”

Bit	*Name*	*R/W*	*Default*	*Description*
15	COEFFICIENT INVALID	RO	–	0 = coefficient read valid 1 = coefficient prefetch not complete
14:0	DSP COEFFICIENT LSB’s	R/W	XXXXh	COEFFICIENT[14:0]

UPPER WORD SELECT = “1”

Bit	*Name*	*R/W*	*Default*	*Description*
15	COEFFICIENT INVALID	RO	–	0 = coefficient read valid 1 = coefficient prefetch not complete
14:4	UNUSED	R/W	00000000000	write as 0, ignore on read
3:0	DSP COEFFICIENT LSB’s	R/W	Xh	COEFFICIENT[18:15] For ECHO & NEXT : only bits [1:0] are valid with sign extension.

DCOFFSET FILTER COEFFICIENT READ/WRITE: (no prefetch necessary; all channel select used on writes only) UPPER WORD SELECT = "X"

Bit	*Name*	*R/W*	*Default*	*Description*
15:0	DSP COEFFICIENT MSB's	R/W	XXXXh	COEFFICIENT[18:3]; COEFFICIENT[2:0] always written as zero Reads upper 16 bits of DCOFFSET coefficient.

FFE FILTER COEFFICIENT READ/WRITE: (no prefetch necessary; all channel select used on writes only) UPPER WORD SELECT = "X"

Bit	*Name*	*R/W*	*Default*	*Description*
15:0	DSP COEFFICIENT	R/W	XXXXh	COEFFICIENT[15:0]; bits [15:14] are always "00".

16h–1Fh Register Map Detailed Description

16h: DSP_Control Register (x4x9 for bits 1 and 2)

Bit	*Name*	*R/W*	*Default*	*Description*
15:13	RESERVED	R/W	000	write as 0, ignore on read
12	BUSY	RO	0	BIT is set if a read/write to register 15h for DFE, ECHO, or NEXT FILTER IS ACTIVE, or a MACRO OPERATION listed with a $ below is active. (Busy will block all writes to registers 15-17h and reads to register 15h.)
11	TAP READ PREFETCH FOR WRITES, ARBITRATION ERROR FOR READS $	R/W	0	On Write 1 = read prefetch (required for 1st access of a block of consecutive reads to echo, next, or dfe filters) 0 = normal On Read 1 = error (write to 15-17h occurred while busy active) 0 = no error
10	UPPER WORD SELECT	R/W	0	1 = read/write 16 MSBs of coefficient 0 = read/write 16 LSBs of coefficient This value toggles each time register 15h is accessed with echo, next, or dfe filter selected.
9	WRITE FILTER COEFFICIENTS FROM RAM	R/W SC	0	1 = Filter and channel specified in register 17h will be written from RAM. (Only echo, dfe, or next can be selected; all filter select and all channel select ignored) 0 = normal operation
8	WRITE ALL NEXT FILTER COEFFICIENTS FROM RAM$	R/W SC	0	1 = All next filters will be written from RAM. (filter and channel select ignored) 0 = normal operation
7	WRITE RAM FROM FILTER	R/W SC	0	1 = Read Filter and channel specified in register 17h and write to RAM. (Only echo, dfe, or next can be selected; all filter select and all channel select ignored) 0 = normal operation
6	INITIALIZE RAM	R/W SC	0	1 = Initialize RAM to all zeros (filter select and channel select ignored) 0 = normal operation
5	INITIALIZE ECHO/NEXT $	R/W SC	0	1 = Initialize all echo/next filters on selected channel specified in register 17h to zeros (all channel select is used; all filter select ignored) 0 = normal operation

Bit	*Name*	*R/W*	*Default*	*Description*
4	INITIALIZE DFE $	R/W SC	0	1 = Initialize all dfe on selected channel specified in register 17h to zeros (all channel select is used; all filter select ignored) 0 = normal operation
3	RESERVED	R/W	0	write as 0, ignore on read
2	DISABLE FILTER (AND filter output with 0)	R/W	0	1 = disable specified filter (all filter select and all channel select used) 0 = enable specified filter
1	FREEZE FILTER	R/W	0	1 = freeze specified filter (all filter select and all channel select used) 0 = unfreeze specified filter
0	RESERVED	R/W	0	write as 0, ignore on read

$ Only 1 macro operation can be active at a time ([5:4] can both be active at the same time).
All macro operations will generate a busy bit until the action is complete.

17h: DSP Coefficient Address Register

Bit	*Name*	*R/W*	*Description*	*Default*
15	CONTROL ALL CHANNELS	R/W	when this bit is set, writes to per-channel control bits affect all channels, regardless of bits 14:13	0
14:13	CHANNEL SELECT	R/W	channel select for DSP coefficient read/writes and per-channel control/status register bits (marked by *): 11 = channel 3 10 = channel 2 01 = channel 1 00 = channel 0	00
12	CONTROL ALL FILTERS (echo, next, dfe)	R/W	when this bit is set, writes to per-filter control bits affect all filters in the specified channel, regardless of bits 11:8 (when bit 15 is also set, writes to DSP control bits affect all echo, next, and dfe filters in the chip)	0

Bit	*Name*	*R/W*	*Description*	*Default*
11:8	FILTER SELECT	R/W	select DSP filter for coefficient read/write: 1111 = EXPANSION REGISTERS 1110 = EXTERNAL SerDes REGISTERS 1101 = reserved 1100 = DCOFFSET 1011 = reserved 1010 = reserved 1001 = reserved 1000 = reserved 0111 = NEXT[3] 0110 = NEXT[2] 0101 = NEXT[1] 0100 = NEXT[0] 0011 = ECHO 0010 = DFE 0001 = FFE 0000 = misc. receiver registers (see bits 7:0) note: NEXT[n] does not exist for channel n. If NEXT[n] is selected for channel n, all NEXT cancellers for that channel are selected when writing control bits. BIT 12 (CONTROL ALL FILTERS) MUST BE ZERO IN ORDER TO SELECT MISC, DCOFFSET, or FFE.	0000
7:0	TAP NUMBER Tap Number gets incremented whenever: [read 15h and filter = (echo,next,or dfe) and upperword='1')] OR [write 15h and filter = (echo,next,or dfe) and upperword = '1')]	R/W	selects which tap is to be read/written within the selected filter (taps are numbered from 0 to n in chronological order (earliest to latest)) when filter select = 000 (misc. receiver regs): 0 = AGC A Register 1 = AGC B & IPRF Register 2 = MSE/Pair Status Register 3 = Soft Decision Register 4 = Phase Register 5 = WireMap/Skew & ECHO/NEXT & TX & ADC Register 6 -8 = reserved 9 = Frequency Register 10 = PLL Bandwidth & Path Metric Register 11 = PLL Phase Offset Register ...to 31, 61:63	00h

18h: Auxiliary Control Register (Shadow Register Selector = "000")

Bit	*Name*	*R/W*	*Description*	*Default*
15	EXTERNAL LOOPBACK	R/W	1 = external loopback enabled 0 = normal operation	extlpbk and mode_sel / = "11" (gbic) 54880: extlpbk tied to gnd at top level
14	EXTENDED PACKET LENGTH	R/W	1 = allow reception of extended length packets (GBIC) 0 = allow normal length Ethernet packets only	mode_sel[0] and mode_sel[1]
13:12	EDGERATE CONTROL (1000T) (LSB or'ed with ER pin)	R/W	00 = 4.0ns (1000T) 01 = 5.0ns (1000T) 10 = 3.0ns (1000T) 11 = 0.0ns (1000T)	00
11	ENABLE SM_DSP CLOCK	R/W	1 = Clock is enabled. 0 = Clock is gated off.	0
10	TRANSMIT 6dB CODING	R/W	1 = transmit using 6dB coding 0 = transmit using 3dB coding	1
9:8	RECEIVE SLICING	R/W	00 = normal Viterbi/DFE MLSE 01 = 4D symbol by symbol slicing for 3 dB option 10 = 3 level 1D symbol by symbol slicing 11 = 5 level 1D symbol by symbol slicing during SEND IDLE/DATA, 3 level else	00
7	DISABLE PARTIAL RESPONSE FILTER	R/W	1 = transmitter partial response filter disabled 0 = transmitter partial response filter enabled	0
6	DISABLE INVERSE PRF	R/W	1 = receiver inv. partial response filter disabled (overrides Phy Control and other MII register settings if disabled) 0 = receiver inv. partial response filter enabled	0
5:4	EDGERATE CONTROL (100TX) (LSB or'ed with ER pin)	R/W	00 = 4.0 ns (100TX) 01 = 5.0 ns (100TX) 10 = 3.0 ns (100TX) 11 = 0.0 ns (100TX)	00
3	DIAGNOSTIC MODE	R/W	1 = When convergence fails, hold in failed state until cleared 0 = Normal operation, retrain on failure	0

Bit	Name	R/W	Description	Default
2 :0	SHADOW REGISTER SELECTOR (These bits are written on all writes to 18h regardless of the value)	R/W	000 = Normal operation 001 = 10 BASE-T register 010 = Power Control register 011 = IP Phone register 100 = Misc Test register 1 101 = Misc Test register 2 110 = Manual IP Phone Seed register 111 = Misc Control register Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	000

18h: 10BASE-T Register (Shadow Register Selector = "001")

Bit	Name	R/W	Description	Default
15	MANCHESTER CODE ERROR	RO LH	1 = manchester code error (10BASE-T) 0 = no manchester code error	0
14	EOF ERROR	RO LH	1 = EOF detection error (10BASE-T) 0 = no EOF detection error	0
13	POLARITY ERROR	RO	1 = channel polarity inverted 0 = channel polarity correct	0
12	BLOCK RXDV EXTENSION (IPG)	R/W	1 = block rxdv for 4 additional rxc cycles for ipg 0 = normal operation	0
11	10BT TXC INVERT MODE	R/W	1 = invert TXC output 0 = normal operation	0
10	CLASS A/B LINE DRIVER SELECT (CLASSB_BT)	R/W	1 = select class A line driver 0 = select class B line driver	0
9:	JABBER DISABLE	R/W	1 = Jabber function disabled 0 = Jabber function enabled (half-duplex only)	0
8	10BT SIGNAL DETECT AUTOSWITCH	R/W	1 = 10BT sigdet threshold auto drops during receive packets, improves CAT3 cable reach 0 = 10BT sigdet threshold controlled by bit 7	1
7	10 BASE-T SIGNAL DETECT THRESHOLD	R/W	0 = high signal detect threshold 1 = low signal detect threshold	0
6	10BT ECHO MODE	R/W	1 = echo transmit data to receive data 0 = normal operation	0

Bit	Name	R/W	Description	Default
5	SQE ENABLE MODE	R/W	1 = enable SQE 0 = disable SQE	0
4	10BASE-T NO DRIBBLE	R/W	1 = correct 10BT dribble nibble 0 = normal operation	0
3	10BASE-T POLARITY ERROR COUNT MAX	R/W	1 = 1 polarity error needed for 10BT polarity swap 0 = 7 polarity errors needed for 10BT polarity swap	0
2:0	SHADOW REGISTER SELECTOR	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	001

18h: Power/MII Control Register (Shadow Register Selector = "010")

Bit	Name	R/W	Description	Default
15:6	Reserved	–	–	–
5	SUPER ISOLATE	R/W	1 = isolate mode with no linkpulses transmitted 0 = normal operation	Super_isolate_def
4:3	Reserved	–	–	–
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	010

18h: IP Phone Register (Shadow Register Selector = "011")

Bit	Name	R/W	Description	Default
15	NON-STOP IP PHONE DETECT MODE	R/W	1 = IP status will always update 0 = IP status will halt after detecting an IP PHONE	0
14:10	EXTENDED LINK PULSE WIDTH COUNTER	R/W	00000 = normal link pulse width, otherwise additional link pulse width in 50 ns increments	00000

Bit	Name	R/W	Description	Default
9	ALTERNATE RANDOM SEED GENERATION	R/W	1 = generate random seed on next received link pulse if flp_receive_idle is true 0 = normal operation Writing this bit requires a RESTART of auto-negotiation.	0
8	RESTART AUTO-NEGOTI ATION	R/W SC	1 = restart auto-negotiation 0 = normal operation	0
7	IP PHONE WINDOW MODE	R/W	1 = enable IP Phone window detection mode 0 = normal operation	0
6	EXTENDED LINK PULSE WIDTH ENABLE	R/W	1 = enable extended link pulse width transmission 0 = normal operation	0
5	ENABLE IP PHONE DETECTION	R/W	1 = IP Phone detection enabled 0 = IP Phone detection disabled	0
4	DISABLE BLOCK LINK10 WINDOW MODE ON WRITES, IP PHONE MISMATCH ON READS	R/W	Write: 1 = disable block link10 window mode 0 = enable block link10 window mode (filters out received linkpulses for 2.5-5us after detecting a linkpulse; ip phone detection must be enabled) Read: 1 = link partner is not an IP PHONE 0 = link partner is an IP PHONE if bit 3 set other wise not determined	0
3	DISABLE SWITCH ON/OFF BLOCK LINK10 WINDOW MODE ON WRITES, IP PHONE DETECTED ON READS	R/W	Write: 1 = block link10 window mode always on/off (controlled by bit 4) 0 = shut off block link10 window mode when IP MISMATCH detected. Restart nway_flp_rx block (ip phone detection must be enabled) Read: 1 = link partner is an IP PHONE 0 = link partner is not an IP PHONE if bit 4 set on read otherwise not determined	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	011

18h: Misc Test Register 1 (Shadow Register Selector = "100")

Bit	*Name*	*R/W*	*Description*	*Default*
15	REMOTE LOOPBACK ENABLE	R/W	1 = enable loopback from MDI (cable end) receive packet, through pcs and back to MDI trans mit packet. (only use in copper or fiber mode) 0 = disable loopback	0
14	TDK FIX ENABLE	R/W	1 = TDK fix (extend EOP on transmit 10BT packets)	1
13	ENABLE DEDICATED 10BT DLL BYPASS CLOCK	R/W	1 = 10BT dll bypass clock generated from tpin10 in dll bypass mode 0 = 10BT dll bypass clock generated from inverted xtali input in dll bypass mode (BASET, ADC10BT, PC10BT, CRS10BT, DAC10_100 test modes use tpin10; register value ignored)	0
12	BLOCK 10BT RESTART AUTO-NEGOTIATION	R/W	1 = prevent 10BT from restarting auto-negotiation in order to break the link 0 = normal operation	0
11	REMOTE LOOPBACK TRISTATE	R/W	1 = tristate the receive MII pins (CRS, RXDV, RXD, etc.) when Remote Loopback is enabled (only use in copper or fiber mode) 0 = Remote Loopback packets appear on MII	0
10	10BT WAKEUP	R/W	1 = enable 10BT dac	0
9	10BT POLARITY BYPASS	R/W	1 = enable polarity bypass 0 = normal operation	0
8	10BT IDLE BYPASS	R/W	1 = enable idle bypass 0 = normal operation	0
7	10BT CLOCK RESET ENABLE	R/W	1 = clock reset controlled from tpin11 0 = normal operation	0
6	10BT BYPASS ADC	R/W	1 = bypass 10BT adc 0 = normal operation	0
5	10BT BYPASS CRS	R/W	1 = bypass 10BT crs 0 = normal operation	0
4	SWAP RXMDIX	R/W	1 = rx and tx operate on same pair 0 = normal operation	0
3	HALFOUT	R/W	1 = transmit half amplitude, all speeds 0 = normal operation also see exp reg f9.1	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	100

18h: Misc Test Register 2 (Shadow Register Selector = "101")

Bit	*Name*	*R/W*	*Description*	*Default*
15	COPPER ENERGY DETECT OVERRIDE	R/W	1 = override copper energy detect 0 = normal operation	0
14	ADCFIFO_TX_FIX	R/W	0 = enable adcfifo fix for TX mode	0
13	Class A/B DVT enable (5478-A4/A5 and later)	R/W	1 = enable, 0 = disable	0
12	CLASS A/B ENABLE (100tx speed)	R/W	1 = class AB txdac mode for 100tx speed 0 = class A txdac mode See reg 18-2.15:13	0
11:10	ENC error scale	R/W	00 = no scaling 01 = scaled by 0.5 10 = scaled by 0.25 others : no scaling	00
9	SPARE	R/W	write as 0, ignore on read	0
8	Disable Auto Encoding Correction	R/W	0 = Autoencoding correction enabled (overrides bits 6 & 7) 1 = Autoencoding correction disabled	0
7	Old PCS Encoding RX	R/W	0 = Select IEEE compliant PCS encoding (for PCS receive) 1 = Select old PCS encoding (for PCS receive)	0
6	Old PCS Encoding TX	R/W	0 = Select IEEE compliant PCS encoding (for PCS receive) 1 = Select old PCS encoding (for PCS receive)	0
5	Enable EC as NEXT	R/W	1 = enable, 0 = disable	0
4	Enable force_mdix	R/W	1 = enable, 0 = disable	0
3	En_PWRDNTDAC	R/W	1 = enable, 0 = disable	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	101

18h: Manual IP Phone Seed Register (Shadow Register Selector = "110")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITABLE IP PHONE SEED	R/W	1 = use manual IP Phone seed 0 = normal operation	0
14	SPARE	R/W	write as 0, ignore on read	0
13:3	LOCAL IP PHONE SEED	R/W	returns the automatically generated ip phone random seed when bit 15 is cleared. Writable value used when bit 15 is set. The ip phone seed is a 14 bit value [13:0]. [2:0] cannot be read. [2:0] will always use "001" when bit 15 is set.	00
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	110

18h: Miscellaneous Control Register (Shadow Register Selector = "111")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE (BITS [8:3])	R/W SC	1 = write bits [8:3] 0 = only write bits [14:12]	0
14:12	SHADOW REGISTER READ SELECTOR	R/W	000 = shadow register 0 read select 001 = shadow register 1 read select ... 111 = shadow register 7 read select	000
11	PACKET COUNTER MODE	R/W	1 = count packets received 0 = count packets transmitted	0
10	BYPASS WIRESPEED TIMER	R/W	1 = Link fail counter will clear as soon as link is up 0 = Link must be up for at least 2.5 seconds otherwise link fail counter will increment. Note: Can be set only if gphy port werespd_timer_disable = 0	ghpy:0
9	FORCE AUTO MDIX MODE	R/W	1 = Auto-mdix will operate when autoneg is disabled via reg 0.12 0 = Auto-mdix is disabled when autoneg is disabled via reg 0.12	forced_automdix_def 54880: tied to gnd at top level

Bit	Name	R/W	Description	Default
8	RGMII TIMING MODE (output delay only)	R/W	1 = clock delayed 90 degrees 0 = clock and data aligned	0
7	RGMII MODE	R/W	1 = use reduced gmii mode 0 = normal gmii/mii operation	0
6	RGMII RXER MODE	R/W	1 = mux rx_er with rx_dv for rgmii mode 0 = mux crs with rx_dv for rgmii mode	1
5	RGMII OUT-OF-BAND STATUS DISABLE	R/W	1 = send regular rx data during IPG 0 = send out-of-band status info in RGMII mode	1
4	WIRESPEED ENABLE	R/W	1 = enable wirespeed mode 0 = normal operation Note: Can be set only if gphy port wirespd_enable = 1.	gphy:wirespd_def (tipin(3))
3	MDIO ALL PHY SELECT	R/W	1 = all phy selected during mdio writes when the phy address = "00000" 0 = normal operation	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	R/W	Writes to the selected shadow register are done on a single cycle (no setup required). Reads are selected by first writing to register 18h, shadow 7, bits 14:12.	111
Bit 7 affects mode of operation. Please consult 0.5. for further information.				

19h: Auxiliary Status Summary (copper side only)

Bit	Name	R/W	Description	Default
15	AUTO-NEGOTIATION COMPLETE	RO	1 = auto-negotiation complete 0 = auto-negotiation in progress	0
14	AUTO-NEGOTIATION COMPLETE ACK	RO LH	1 = entered auto-neg "link good check" state 0 = state not entered since last read	0
13	AUTO-NEGOTIATION ACK DETECT	RO LH	1 = entered auto-neg "acknowledge detect" state 0 = state not entered since last read	0
12	AUTO-NEGOTIATION ABILITY DETECT	RO LH	1 = entered auto-neg "ability detect" state 0 = state not entered since last read	0
11	AUTO-NEGOTIATION NEXT PAGE WAIT	RO LH	1 = entered auto-neg "next page wait" state 0 = state not entered since last read	0

Bit	Name	R/W	Description	Default
10:8	AUTO-NEGOTIATION HCD (CURRENT OPERAT ING SPEED AND DUPLEX MODE)	RO	111 = 1000BASE-T full duplex* 110 = 1000BASE-T half duplex* 101 = 100BASE-TX full duplex* 100 = 100BASE-T4 011 = 100BASE-TX half duplex* 010 = 10BASE-T full duplex* 001 = 10BASE-T half duplex* 000 = no highest common denominator (when auto-neg complete = 1) or auto-negotiation not complete (when auto-neg complete = 0)	000
7	PARALLEL DETECTION FAULT	RO LH	1 = parallel detection fault 0 = no parallel detection fault	0
6	REMOTE FAULT	RO	1 = link partner has detected remote fault 0 = link partner has not detected remote fault	0
5	PAGE RECEIVED	RO LH	1 = new link code word has been received 0 = new link code word has not been received	0
4	LINK PARTNER AUTO- NEG. ABILITY	RO	1 = link partner is auto-negotiation able 0 = link partner is not auto-negotiation able	0
3	LINK PARTNER NEXT PAGE ABILITY	RO	1 = link partner is next page able 0 = link partner is not next page able	0
2	LINK STATUS	RO	1 = link pass 0 = link fail	0
1	PAUSE RESOLUTION - RECEIVE DIR	RO	1 = enable pause receive 0 = disable pause receive	0
0	PAUSE RESOLUTION - TRANSMIT DIR	RO	1 = enable pause transmit 0 = disable pause transmit	0

* Indicates the negotiated HCD when auto-negotiation complete = 1. Indicates the manually selected speed and duplex mode when auto-negotiation enable = 0.

1Ah: Interrupt Status Register (copper side only)

Bit	Name	R/W	Description	Default
15	IP STATUS CHANGE (register 1c shadow 5 bit 5 = 0)	RO LH	1 = IP status changed since last read 0 = interrupt cleared	0

Bit	*Name*	*R/W*	*Description*	*Default*
	SD/ENERGY DETECT CHANGE (register 1c shadow 5 bit 5 = 1)	RO LH	1 = filtered fiber signal detect (via en10b pin) change or energy detect change since last read 0 = interrupt cleared	0
14	ILLEGAL PAIR SWAP	RO LH	1 = Illegal pair swap detected 0 = interrupt cleared	0
13	MDIX STATUS CHANGE	RO LH	1 = MDIX status changed since last read (means that linkpulse or carrier was detected on a different pair than previously detected) 0 = interrupt cleared	0
12	EXCEEDED HIGH COUNTER THRESH.	RO	1 = value in one or more counters is above 32k 0 = all counters below 32k	0
11	EXCEEDED LOW COUNTER THRESH.	RO	1 = value in one or more counters is above 128 0 = all counters below 128	0
10	AUTO-NEG. PAGE RX	RO LH	1 = page received since last read 0 = interrupt cleared	0
9	HCD NO LINK	RO LH	1 = negotiated HCD did not establish link 0 = interrupt cleared	0
8	NO HCD	RO LH	1 = auto-negotiation returned HCD=none 0 = interrupt cleared	0
7	NEGOTIATED UNSUPPORTED HCD	RO LH	1 = auto-negotiation HCD is not supported by local PHY 0 = interrupt cleared	0
6	SCRAMBLER SYNC. ERROR	RO LH	1 = scrambler synchronization error occurred since last read 0 = interrupt cleared	0
5	REMOTE RECEIVER STATUS CHANGE	RO LH	1 = remote receiver status changed since last read 0 = interrupt cleared	0
4	LOCAL RECEIVER STATUS CHANGE	RO LH	1 = local receiver status changed since last read 0 = interrupt cleared	0
3	DUPLEX MODE CHANGE	RO LH	1 = duplex mode changed since last read 0 = interrupt cleared	0
2	LINK SPEED CHANGE	RO LH	1 = link speed changed since last read 0 = interrupt cleared	0
1	LINK STATUS CHANGE	RO LH	1 = link status changed since last read 0 = interrupt cleared	0
0	CRC ERROR	RO LH	1 = CRC error occurred since last read 0 = interrupt cleared	0

1Bh: Interrupt Mask Register

Bit	*Name*	*R/W*	*Description*	*Default*
15:0	INTERRUPT MASK VECTOR	R/W	1 = interrupt masked (status bits still operate normally but do not generate interrupt output) 0 = interrupt enabled	FFFFh

1Ch: Cabletron LED Register (Shadow Register Selector = "00h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector 00000 = shadow register 0 read/write select 00001 = shadow register 1 read/write select ... 11111 = shadow register 31 read/write select	00000
9:8	RESERVED	R/W	write as 0, ignore on read	-
7:0	CABLETRON LED REGISTER	R/W	[7] cabletron led select 1 = select cabletron 0= normal mode [6] online 1 = online 0 = offline [5] tx enable 1 = transmit enable 0 = transmit disable [4] rx enable 1 = receive enable 0 = receive disable [3] link enable 1 = link enable 0 = link disable [2] led mode 1 = traffic mode 0 = normal mode [1] bypass internal yellow and blink clocks 1 = enable external yellow and blink clocks, via tpin0 and tpin2, respectively. 0 = use internally generated clocks [0] spare	00000000

For the BCM5725/BCM5762/BCM57767 GPHY, 1Ch, shadow register from 01 to 1F will not be reset by reg0[15].

1Ch: DLL Selection Register (Shadow Register Selector = "01h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00001
9:6	RESERVED	R/W	write as 0, ignore on read	000

5:0	HIGH QUALITY CLOCK TEST MODE	R/W	xx000x = no clock
			nn0010 = clk125
			nn0011 = rxclk
			nn01yy = rxiclk(yy), where yy = 0-3
			nn1yyy = txiclk(yyy), where yyy = 0-7
			
			1. nn = 0-3 for slice 1-4
			2. test mode selection is from slice1.
				0000

1Ch: Spare Control 1 Register (Shadow Register Selector = "02h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00010
9	SD status	RO	1= sd input active	0
8	FORCE SD ON	R/W	1= force sd on. disable sd pin input	0
7	INVERT SD PIN	R/W	1 = invert sd pin	0
6	CFC_INITFILTER ENABLE	R/W	1 = enable cfc_initFilter signal to control clock gating of 1000t clocks. Do not gate off 1000t clocks wherever cfiltercntl is initializing the filter.	0
5	USE FILTERED SD	R/W	1= enable filter on sd input pin	0
4	100FX MODE COPPER PATH	R/W	1 = enable 100BASE-FX on TRD+/- pins 0 = normal copper operation on mdi pairs	0
3	RESERVED	R/W	write as 0, ignore on read	0

Bit	*Name*	*R/W*	*Description*	*Default*
2	BICOLOR LINK SPEED LED MODE	R/W	1 = enable Bicolor Link Speed led mode LINKSPD[1:0] = speed 10 = 1000 base-t 01 = 100 base-t 11 = auto-negotiation, 10 base-t	0
1	LOST TOKEN FIX DISABLE	R/W	When 0, enables lost token fix reset circuits	1
0	LINK LED MODE	R/W	1 = enable Link LED mode: LINKSPD[1:0] = speed 00 = 1000 base-t 01 = 100 base-t 10 = 10 base-t 11 = auto-negotiation SLAVE = active low link 0 = normal link/slave mode	0

1Ch: Clock Alignment Control Register (Shadow Register Selector = "03h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00011
9	GTXCLK delay bypass disable (input delay only)	R/W	1 = do not bypass gtxclk delay 0 = bypass gtxclk delay	0
8	GMII CLOCK ALIGNMENT STROBE	R/W	Delay value is latched into selected GMII clock delay line on rising edge of this bit	0
7	RXCLK ALIGNMENT STROBE	R/W	Delay value is latched into selected RX clock delay line on rising edge of this bit	0

Bit	*Name*	*R/W*	*Description*	*Default*
6:4	DELAY VALUE	R/W	RXCLK delay: reset = default delay 000 = +1 unit delay ... 111 = +8 units delay GMII clock delay: 110 = -1.0ns 111 = -0.5ns 000 = 0ns 001 = 0.5ns 010 = 1.0ns	000
3:0	DELAY LINE SELECTOR	R/W	RXCLK strobe: xx00 = std cell rxclk xx01 = dfse rxclk xx10 = dfe rxclk xx11 = enc rxclk GMII clock strobe: 0111 = TBI gtx_clk 1000 = GMII gtx_clk 1001 = RGMII gtx_clk 1010 = GMII rx_clk 1011 = RGMII rx_clk 1100 = TBI RBC0 1101 = TBI RBC1	0000

1Ch: Spare Control 2 Register (Shadow Register Selector = "04h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00100
9	SPARE	R/W	Flop exists, but does not change any function.	0
8	wirespeed retry disable (fixed in 54980)	R/W	1 = downgrade after 1 failed link attempt 0 = use wirespeed retry limit (1c-04.4:2)	0
7	select tpout_rxd	R/W	1 = tpout_rxd port muxed to rxd 0 = normal operation for rxd	0

Bit	*Name*	*R/W*	*Description*	*Default*
6	DISABLE PHYA2	R/W	1 = internally disable phya2 input (consult Testability document for suggested usage)	0
5	enable rbc0/1 & txc/rxc tri sate	R/W	1 = enable tristating of rbc0/1 or txc/rxc 0 = rbc0/1 & txc/rxc not tristated	0
4:2	WIRESPEED RETRY LIMIT	R/W	000: downgrade after 2 failed link attempts 001: downgrade after 3 failed link attempts ... 111: downgrade after 9 failed link attempts	011
1	ENERGY DETECT ON INTR PIN	R/W	1 = routes Energy Detect to interrupt signal. Use LED selectors (reg 1c shadow 01101 and 01110) to direct interrupt signal to an LED output.	0
0	testonbyte7_0	R/W	Controls value of gphy output port testonbyte7_0. 5478 1 = mux tpout[7:0] to led2[4:1] , led1[4:1]	0

1Ch: Spare Control 3 Register (Shadow Register Selector = "05h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00101
9	DLL lock enable during auto-power down	R/W	1 = allow time for dll to lock before enabling clocks & analog components. Only applicable when dll is powered down during auto-power down (r1c.5 bit 1 is LOW).	0
8	txc/rxc disable during auto-power down	R/W	1 = disable txc/rxc during auto-power down when there's no energy on the cable	0
7	10BT CARRIER REJECT FILTER ENABLE	R/W	1 = enable 10BT 15MHz Carrier Rejection Filter	0
6	TXC OFF ENABLE	R/W	1 = gates off TXC output in 1000T/1000-X/sgmii slave 1000 mode	0
5	SD/energy detect change mux select	R/W	1 = interrupt based on energy detection (copper energy detect change or filtered fiber signal detect change from input pin) 0 = normal ipphone interrupt selected	0
4	LOW POWER ENC DISABLE	R/W	1 = disable low power ENC mode	1
3	DISABLE LOW POWER 10BASE-T LINK MODE	R/W	1 = disable low power 10BASET link mode	1
2	SIGDET DEASSERT TIMER LENGTHEN	R/W	1 = 100TX Sigdet Deassert Timer = 40 us 0 = Sigdet Deassert Timer =1 us	1

Bit	Name	R/W	Description	Default
1	AUTO-POWER DOWN DLL OFF DISABLE	R/W	1 = disable powering down of the dll during auto-power down 0 = enable powering down of dll during auto-power down	1
0	CLK125 OUTPUT ENABLE	R/W	1 = enable CLK125 output 0 = disable CLK125 output	1

1Ch: Tdr Control 1 Register (Shadow Register Selector = "06h"

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00110
9	SPARE	R/W	write as 0, ignore on read	0
8	TDR LINK TIME OUT	R	tdr linkpulse time out status	0
7:5	TEST PULSE SIZE	R/W	size of test pulse in increments of 8 ns. minimum value = 1.	001
4:3	TX CHANNEL SEL	R/W	channel to transmit test pulse	00
2:1	RX CHANNEL SEL	R/W	channel to receive test data	00
0	TDR START/DONE	R/W SC	write: 1: tdr start. self clearing read: 1: tdr done status	0

1Ch: Tdr Control 2 Register (Shadow Register Selector = "07h")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	00111
9:8	SPARE	R/W	write as 0, ignore on read	00
7	PHASE STATUS	R	phase status	0
6	PHASE STATUS CLEAR	R/W	1: clear phase status on bit 7	0
5	FASTTIMERS	R/W	1: enable fasttimers	0

Bit	Name	R/W	Description	Default
4	FEXT	R/W	1: enable fext mode 0: disable fext mode	0
3	MASTER	R/W	1: master mode 0: slave mode	0
2	EXTERNAL PHY NO AUTO-NEG	R/W	1: tdr test with external phy without auto-negotiation 0: tdr test with external phy with auto-negotiation	0
1	EXTERNAL PHY	R/W	1: tdr test with external phy 0: tdr test as stand-alone phy	0
0	TDR MODE ENABLE	R/W	1: enable tdr mode 0: disable tdr mode	0

1Ch: Led Status Register (Shadow Register Selector = “08h”)

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01000
9	RESERVED	RO	ignore on read	0
8	SLAVE_N	RO	(active low)	–
7	FDXLED_N	RO	(active low)	–
6	INTR_N	RO	(active low)	–
5	SPARE	RO	ignore on read	–
4:3	LINKSPD_N	RO	11: no link 10: 10bt link 01: 100tx link 00: 1000t link	–
2	TRANSMIT LED	RO	(active low)	–
1	RECEIVE LED	RO	(active low)	–
0	QUALITY LED	RO	(active low)	–

1Ch: Led Control Register (Shadow Register Selector = "09h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01001
9	collision blink led mode	R/W	1 = blink fdxled when collision occurs 0 = fdxled indicates duplex status only	0
8	activity link led MSB	R/W	See description for bit 4 below.	0
7	SPARE	R/W	write as 0, ignore on read	0
6	external_serdes_inuse led mode	R/W	1= drive transmit led low when external SerDes is selected, inactive when not selected 0= normal operation	0
5	Override GBIC LED mode	R/W	1 = LEDs not remapped in GBIC mode. 0 = In GBIC mode LEDs mapped as follows. LED1: RX_LOSS (copper link) LED2: RX (copper receive activity) LED3: TX (copper transmit activity) LED4: LINK (both SerDes and copper are linked)	0
4	activity link led LSB	R/W	Combined with bit 8 above. 11 = activity,linkspd[1:0] indicate 1000,100,10 link respectively, and blinks with activity. 10 = linkspd[1:0] indicate encode link, and blinks when activity 01 = active indicates link, and blinks when activity 00 = normal operation These bits overrides bit 3 below.	0
3	ACTIVITY LED ENABLE	R/W	1 = normal operation (activity led indicates transmit or receive activity). 0 = activity led indicates receive activity only	1
2	REMOTE FAULT LED ENABLE	R/W	1 = drive remote fault on quality led 0 = normal operation	0
1:0	LINK UTILIZATION LED SELECTOR (NORTEL LED)	R/W	00 = normal operation 01 = transmit data on receive led 10 = receive data on receive led 11 = activity data on receive led (This mode has higher priority than the activity led enable in bit 3.)	00

1Ch: Auto-Power Down Register (Shadow Register Selector = "0ah")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01010
9:8	RESERVED	R/W	write as 0, ignore on read	00
7	LOWPWR136 ENC ENABLE	R/W	1=disable last 56 taps of EC	0
6	AUTO POWERDOWN IGNORE AUTONEG	R/W	1 = ignore autoneg enable during auto-power down mode	0
5	AUTO POWERDOWN MODE ENABLE	R/W	1 = enable auto power-down	autopwrdn_reg_def 5478: tied to gnd at top level
4	SLEEP TIMER SEL	R/W	1 = Sleep timer is 5.4 seconds 0 = Sleep timer is 2.7 seconds	0
3:0	WAKE UP TIMER SEL	R/W	Counter for wakeup timer in units of 84 ms	0001

1Ch: Spare Control 4 Register (Shadow Register Selector = "0bh")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01011
9:8	RESERVED	R/W	write as 0, ignore on read	00
Slice 1				
7	Sled shift bit selection	R/W	Select which bit to be shifted out to next chip in the daisy-chain. This is sled_sel2_pprt(1). 00 = from slice 8 01 = from slice 4 10 = from slice 2 11 = from slice 1	0
6	Sled output disable	R/W	1 = all serial LED outputs are high impedance. 0 = all serial LEDs are driven qualified by sled_enable below.	0
5	Reserved	–	–	–
4	Serdes tpout enable	R/W	1 = mux SerDes sigdet and freq lock to tpout	0
3	Sled_reset	R/W	1 = reset serial led registers	0

Bit	Name	R/W	Description	Default
2	Sled_enable	R/W	1 = enable serial led function	tdr_mode(0)
1	testonbyte23_16	R/W	1 = enable tpout(23:16) during test mode	0
0	Reserved	R/W	open	0
Slice 2/ slice 6 (not used by BCM5725/BCM5762/BCM57767)				
7	GMII SNOOP Mode Enable (slice2)	R/W	1 = enables GMII SNOOP mode for GMII interface observation. 0 = disables GMII SNOOP mode. Note. See register 1C:0C[7:4] in slice 2 for GMII bus selection.	0
6:5	Change_ref_10BT	R/W	Controls change_ref_10BT value	0
4:0	Bias_control	R/W	Controls bias_control value	10000
Slice 3/ slice 7(not used by BCM5725/BCM5762/BCM57767)				
7:5	Rc_offset	R/W	Controls rc_offset value	0
4:0	Rc_preset_val	R/W	Controls rc_preset_val value.	0
Slice 4 / slice 8(not used by BCM5725/BCM5762/BCM57767)				
7:5	R_offset	R/W	Controls r_offset value. If [7:5] are "000" and reg1C shdw12(slice3) bit 1 is '0' then bits 6 & 5 are inverted and assigned to r_offset, else [7:5] directly controls value of r_offset	0
4:0	R_preset_val	R/W	Controls r_preset_val value.	0
Slice 5 (not used by BCM5725/BCM5762/BCM57767)				
7	Top mii reg write enable	R/W	Write Command. After data has been written to the Top Level MII Write Register, this bit must be set '0'. 0 = Read 1 = Write	0
6:0	Top mii reg address bits	R/W	Top Level MII Register offset address	0000000

1Ch: Spare Control 5 Register (Shadow Register Selector = "0ch")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01100
9:8	RESERVED	R/W	write as 0, ignore on read	00
Slice 1				
7	Reserved	–	–	–
6	Sled shift bit selection	R/W	Select which bit to be shifted out to next chip in the daisy-chain. This is sled_sel2_pprt(0). 00 = from slice 8 01 = from slice 4 10 = from slice 2 11 = from slice 1	0
5:3	External Control 2	R/W	Open	–
2:1	Corner Oscillator ring selection	R/W	00 = Ring number 1, Typ freq is 64 01 = Ring number 2, Typ freq is 56.1 10 = Ring number 3, Typ freq is 52.3 11 = Ring number 4, Typ freq is 56.1	00
0	Corner Oscillator test output enable	R/W	1 = enable. Oscillator output from tpout{1:0]	0
Slice 2				
7:4	GMII Bus Selection in Snoop Mode	R/W	Selects a GMII bus to observe it on tpout bus as follows 0000 - gmii_rx1 1000 - gmii_tx1 0001 - gmii_rx2 1001 - gmii_tx2 0010 - gmii_rx3 1010 - gmii_tx3 0011 - gmii_rx4 1011 - gmii_tx4 0100 - gmii_rx5 1100 - gmii_tx5 0101 - gmii_rx6 1101 - gmii_tx6 0110 - gmii_rx7 1110 - gmii_tx7 0111 - gmii_rx8 1111 - gmii_tx8 Note. GMII SNOOP mode is activated by bit 1C:0B[7] in slice 2.	0000
3	GMII Test Mode Enable	R/W	1 = enables GMII TEST mode for GMII interface external control via tipin bus and for observation of selected GMII bus on tpout bus. 0 = disables GMII TEST mode. Note. See register 1C:0C[2:0] in slice 2 for GMII bus selection in GMII TEST mode.	0

Bit	*Name*	*R/W*	*Description*	*Default*
2:0	GMII Bus Selection in Test Mode	R/W	Select a GMII bus t observe it on tpout bus as follows 000 - gmii_rx1 001 - gmii_rx2 010 - gmii_rx3 011 - gmii_rx4 100 - gmii_rx5 101 - gmii_rx6 110 - gmii_rx7 111 - gmii_rx8 Note. GMII TEST mode is activated by bit 1C:0C[3] in slice 2. All gmii_tx buses are driven simultaneously from tipin bus.	000
Slice 3(not used by BCM5725/BCM5762/BCM57767)				
7	Serdes Select Enable	WO	0 to 1 transition enables bit [6:5] write to SerDes select register	0
6:5	Serdes Select	WO	00 = select slice 7 and 8 01 = select slice 5 and 6 10 = select slice 3 and 4 11 = select slice 1 and 2	00
4:2	External Control 2	R/W	Open	–
1	Set default value of r_offset to pll_bias to improve 10BT return loss	R/W	See description of reg1C shdw11(slice4) bit [7:5]	0
0	Set default 10BT amplitude value	R/W	0 = if slice4 [7:4]=0000, the value sent to bias will be changed to 0001 to increase 10BT amplitude. 1 = slice4 [7:4] value directly sent to bias.	0
Slice 4 (not used by BCM5725/BCM5762/BCM57767)				
7:4	cntrl_vdd_int	R/W	controls cntrl_vdd_int value. If bits [7:4] are "000" and bit 0 of reg 1C shdw 12 (slice3) is '0' then bit 4 is inverted and assigned to cntrl_vdd_int, else bits[7:4] directly controls cntrl_vdd_int value	0000
3	rc_reset	R/W	Controls rc_reset value	0
2	r_reset	R/W	Controls r_reset value	0
1	rc_preset	R/W	Controls rc_preset value	0
0	r_preset	R/W	Controls r_preset value	0
Slice 5(not used by BCM5725/BCM5762/BCM57767)				
7	Run LED Fault Detect BIST	R/W	1 = start LED fault detect bist 0 = reset LED fault detect bist	0

Bit	*Name*	*R/W*	*Description*	*Default*
6:5	LED Fault Detect Debug Select	R/W	Select which LED fault data to debug. See reg1c[0f].	00
4:1	LED Fault Detect Mask	R/W	1 = mask corresponding LED outputs across all slices	0000
0	External Control 2	R/W	Open	–
Slice 6				
7:0	Top MII register data	R/W	Data to be read if bit[7] = '0' of reg1Ch,shdw 0Bh,Prt 5. Data to be written if bit[7] = '1' of reg1Ch,shdw 0Bh,Prt 5.	0

1Ch: LED Selector 1 Register (Shadow Register Selector = "0dh")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01101
9:8	RESERVED	R/W	write as 0, ignore on read	00
7:4	LED2 SELECT	R/W	0000: linkspd(0) 0001: linkspd(1) 0010: xmtled 0011: activity 0100: fdxled 0101: slave 0110: interrupt 0111: quality 1000: rcvled 1001: wirespeed downgrade 1010: Bicolor LED1 1011: Cable Diagnostic Open/Short found 1100: energy_link 1101: sgmii receiving crs (from copper link partner) (do not use if snoop mode is enabled) 1110: off 1111: on	0001

Bit	Name	R/W	Description	Default
3:0	LED1 SELECT	R/W	0000: linkspd(0) 0001: linkspd(1) 0010: xmtled 0011: activity 0100: fdxled 0101: slave 0110: interrupt 0111: quality 1000: rcvled 1001: wirespeed downgrade 1010: Bicolor LED0 1011:Cable Diagnostic Open/Short found 1100: energy_link 1101: sgmii receiving crs (from copper link partner) (do not use if snoop mode is enabled) 1110: off 1111: on	0000

1Ch: LED Selector 2 Register (Shadow Register Selector = "0eh")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01110
9:8	RESERVED	R/W	write as 0, ignore on read	0000
7:4	LED4 SELECT	R/W	0000: linkspd(0) 0001: linkspd(1) 0010: xmtled 0011: activity 0100: fdxled 0101: slave 0110: interrupt 0111: quality 1000: rcvled 1001: RX_SOP 1010: Bicolor LED1 1011:Cable Diagnostic Open/Short found 1100: energy_link 1101: sgmii receiving crs (from copper link partner) (do not use if snoop mode enabled) 1110: off 1111: on	0110

Bit	*Name*	*R/W*	*Description*	*Default*
3:0	LED3 SELECT	R/W	0000: linkspd(0) 0001: linkspd(1) 0010: xmtled 0011: activity 0100: fdxled 0101: slave 0110: interrupt 0111: quality 1000: rcvled 1001: TX_SOP 1010: Bicolor LED0 1011:Cable Diagnostic Open/Short found 1100: energy_link 1101: sgmii receiving crs (from copper link partner) (do not use if snoop mode enabled) 1110: off 1111: on	0011

1Ch: LED GPIO Control/Status Register (Shadow Register Selector = "0fh")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	01111
9:8	RESERVED	R/W	write as 0, ignore on read	0000
7:4	LED DATA IN	RO	bit 7: LED4 in bit 6: LED3 in bit 5: LED2 in bit 4: LED1 in	led4_i, led3_i, led2_i, led1_i
3:0	PROGRAMMABLE CURRENT MODE LED ENABLE	R/W	0 = Enable corresponding LED current mode 1 = Disable corresponding LED current mode bit 3: LED4 bit 2: LED3 bit 1: LED2 bit 0: LED1	0000

1Ch: SerDes 100-FX Status Register (Shadow Register Selector = "11h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10001
9	100FX LINK STATUS CHANGE (SerDes)	RO LH	1 = link status change since last read 0 = link status has not changed since last read	0
8	BAD ESD DETECTED (PREMATURE END)	RO LH	1 = bad ESD error detected since last read 0 = no error detected since last read	0
7	FALSE CARRIER DETECTED	RO LH	1 = false carrier detected since last read 0 = no error detected since last read	0
6	TRANSMIT ERROR DETECTED	RO LH	1 = received packet with txer code since last read 0 = no error detected since last read	0
5	RECEIVE ERROR DETECTED	RO LH	1 = receive coding error detected since last read 0 = no error detected since last read	0
4	LOCK TIMER EXPIRED	RO LH	1 = unable to lock within 730us since last read 0 = condition not detected occurred since last read	0
3	LOST LOCK	RO LH	1 = lost lock since last read 0 = lock has not been lost since last read	0
2	FAULTING	RO LH	1 = far end fault detected since last read 0 = no fault detected since last read	0
1	LOCKED	RO	1 = enough idles are properly detected to lock 0 = not locked	0
0	100FX LINK (SerDes)	RO	1 = link is up 0 = link is down	0

1Ch: SerDes 100-FX Test Register (Shadow Register Selector = "12h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10010
9:7	RESERVED	RO	write as 0, ignore on read	000

Bit	Name	R/W	Description	Default
6	BYPASS NRZ	R/W	1 = bypass NRZ encoder 0 = normal operation	0
5	BYPASS ENCODER	R/W	1 = bypass 4B5B encoder 0 = normal operation	0
4	BYPASS ALIGNMENT	R/W	1 = bypass 5B code group alignment 0 = normal operation	0
3	FORCE LINK	R/W	1 = force link 0 = normal operation	0
2	FORCE LOCK	R/W	1 = force lock 0 = normal operation	0
1	FAST UNLOCK TIMER	R/W	1 = speed up unlock timer 0 = normal operation	0
0	FAST TIMERS	R/W	1 = speed up timers to acquire lock and link (test vectors and simulation) 0 = normal operation	0

1Ch: SerDes 100-FX Control Register (Shadow Register Selector = "13h")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10011
9:7	RESERVED	R/W	write as 0, ignore on read	000
6	100-FX AUTO-DETECT TIMER SELECT	R/W	1 = 125-166 ms (Do not use if fiber auto-power down enabled; register 1ch shadow "1eh" [4]) 0 = 31- 42ms	0
5	DISABLE RX QUALIFY	R/W	1 = always use sample bit without filtering 0 = normal operation	0
4	FORCE RX QUALIFY	R/W	1 = always compare 2 surrounding bits with sample bit to filter noise 0 = normal operation	0
3	FAR END FAULT ENABLE	R/W	1 = enable far-end fault 0 = disable far-end fault	1
2	AUTO-DETECT 100-FX SerDes	R/W	1 = auto-detect between mii/rgmii -> 100fx and mii/rgmii/gmii -> 1000-X 0 = disable auto-detection	autodet_fx100_def 5478: tied to gnd at quad level

Bit	*Name*	*R/W*	*Description*	*Default*
1	100-FX SERDES FULL-DUPLEX	R/W	1 = 100-fx SerDes full-duplex 0 = 100-fx SerDes half-duplex	fx100_fulldplx_def 5478: tied to vdd at quad level
0	100-FX SERDES ENABLE (register 1c shadow 1fh bit [2] must be 0)	R/W	1 = select mii/rgmii -> 100fx (SerDes) 0 = select gmii/rgmii -> 1000X	fx100_serdes_def 5478: tied to gnd at quad level

1Ch: External SerDes Control Register (Shadow Register Selector = "14h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10100
9	EXTERNAL SERDES LINK STATUS CHANGE	RO LH	1 = link status change detected since last read 0 = link status change not detected since last read	0
8	EXTERNAL SERDES LINK	RO	1 = link is up 0 = link is down	0
7	EXTERNAL SERDES DUPLEX	RO	1 = SerDes full-duplex 0 = SerDes half-duplex or auto-negotiating in progress	external_serdes_duple x 5478: tied to gnd at quad level
6:5	SGMII ShadowPHY selection (Only in Slice 0)	R/W	This is only applied to BCM54640 revB0. When tpin[9] = 1, copper phy address is phya+0 ~ phya+3 sgmii phy address is phya+4 ~ phya+7 When tpin[9] = 0, copper and sgmii phy address share the same phyaddress which are phya+0 ~ phya+3. The way to switch between copper and SGMII is a. To sgmii interface Write port 0, 1C_shdw14h[6:5] = 2'b10 Write port 0, 1C_shdw14h[6:5] = 2'b01 b. To copper interface Write port 0, 1C_shdw14h[6:5] = 2'b10 Write port 0, 1C_shdw14h[6:5] = 2'b00	–

Bit	*Name*	*R/W*	*Description*	*Default*
6	EXTERNAL SERDES AUTO-DETECT 100FX MODE	R/W	1 = auto-detect between sgmii->100fx or sgmii->1000x 0 = disable auto-detection	dual_autodet_fx100 and mode_sel_def(1) 5478: dual_autodet_fx100 tied to gnd at quad level
5	EXTERNAL SERDES 100FX FULL-DUPLEX	R/W	1 = external SerDes 100fx full-duplex 0 = external SerDes 100fx half-duplex	dual_fx100_fulldplx and mode_sel_def(1) 5478: dual_fx100_fulldplx tied to gnd at quad level
4	EXTERNAL SERDES 100-FX MODE	R/W	1 = operate external SerDes in 100-fx mode 0 = operate external SerDes in 1000-X mode	dual_fx100_serdes and mode_sel_def(1) 5478: dual_fx100_serdes tied to gnd at quad level
3	EXTERNAL SERDES LED MODE	R/W	1 = use external SerDes transmit, receive, and link for leds whenever external SerDes is selected via bit 0 of this register. 0 = disable external SerDes led mode	(external_serdes_def or dual_autodet_mediu m) and mode_sel_def(1)
2	SELECT SYNC STATUS	R/W	1 = use sync status from external SerDes for dual SerDes auto-medium detection. In 100-FX mode use external SerDes link status. (see register 1ch shadow 1eh [9]) 0 = use sync status from internal SerDes for auto- medium detection. In 100-FX mode use internal SerDes link status. (see register 1ch shadow 1eh [0])	(external_serdes_def or dual_autodet_medium) and mode_sel_def(1)
1	SELECT SD	R/W	1 = use signal detect from pin for external SerDes fiber auto-power-down and pcs synchronization (see register 1ch shadow 1eh [4]) 0 = use signal detect from pin for internal SerDes fiber auto-power-down and pcs synchronization (see register 1ch shadow 1eh[4])	(external_serdes_def or dual_autodet_medium) and mode_sel_def(1)

Bit	*Name*	*R/W*	*Description*	*Default*
0	EXTERNAL SERDES SELECT (register 1c shadow 1fh bit [2] must be 1)	R/W	1 = sgmii-> fiber 0 = sgmii/gbic -> copper	(external_serdes_def or dual_autodet_mediu m) and mode_sel_def(1)

1Ch: SGMII Slave Register (Shadow Register Selector = "15h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10101
9	SERDES LINK	RO	1 = link up in fiber, sgmii,or gbic modes (when set in sgmii or gbic mode, then both copper and SerDes link must be valid) 0 = link down	0
8	SERDES DUPLEX	RO	1 = SerDes full-duplex 0 = SerDes half-duplex or auto-negotiating in progress	1
7:6	SERDES SPEED	RO	10 = SerDes speed 1000 (sgmii 1000 or 1000X) 01 = SerDes speed 100 (sgmii 100 or 100fx) 00 = SerDes speed 10 (sgmii 10)	10
5	SERDES LINK STATUS CHANGE	RO LH	1 = link status change detected since last read 0 = link status change not detected since last read	0
4:3	MODE SELECT	RO	00 = copper 01 = fiber 10 = sgmii 11 = gbic	mode_sel[1:0]

Bit	*Name*	*R/W*	*Description*	*Default*
2	RGMII/MII -> SGMII SLAVE 10/100 TX FIFO FREQUENCY LOCK MODE (forced high internally when => sgmii_slave_mode and not rgmii_mode and not remote_lpbk)	R/W	1 = sgmii transmit FIFO will assume that the SerDes pll clock and the MAC transmit clock are frequency locked. This will essentially bypass the FIFO with the lowest possible latency in 10/100 speeds. (useful for applications where the MAC is in rgmii mode and the PHY is in rgmii-> sgmii slave mode, and both the MAC and PHY are using the same crystal. When the MAC is in mii mode and the PHY is in mii->sgmii slave mode, then this bit should always be set.) 0 = normal operation	sgmii_slave_mode and not rgmii_mode_def 5478 : sgmii_slave_mode to tied to gnd at top level
1	SGMII SLAVE MODE (register 1c shadow 1fh bit [2:1] must be "01")	R/W	1 = enable mii/gmii/rgmii -> sgmii mode (only slave mode supported; useful for MAC rgmii to sgmii conversions , which are attached to an external phy) 0 = disable sgmii slave mode	sgmii_slave_mode 5478: tied to gnd at quad level
0	SGMII SLAVE AUTO-DETECTION (register 1c shadow 1fh bit [2] must be 0)	R/W	1 = enable sgmii slave auto-detection. Switch between 1000-X and sgmii slave modes based on SerDes received auto-negotiation code word. 0 = normal operation	autodet_sgmii_slave 5478: tied to gnd at quad level

1Ch: Misc 1000-X Control 2 Register (Shadow Register Selector = "16h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	WO	1 = write bits [9:7] if [6] = 1 or write bits[5:0] if [6] = 0 0 = read bits [9:0] Read will always return 0	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector NOTE: (If bit 8 of this register is set prior to writing this register, then the selector value written will not be stored; To properly read the shadow registers, one must write bit 8 as a zero, then on a subsequent write set the desired shadow register to read)	10110
9	REMOTE COPPER MODE (register 1c shadow 1fh bits 2:1 must be "11")	R/W	1 = enable remote copper mode (used in backplane applications to attach a remote phy to a mac). 0 = normal operation	remote_copper_mo de _def * mode_sel_def[1] * mode_sel_def[0]
8	ENABLE REMOTE ACCESS	R/W	1 = Enable remote mac link partner to access local mdio registers via SerDes autoneg next-page mdio message pages. Bit [9] of this register must also be set. Local mdio access is blocked. All local mdio read accesses will return the results of this register regardless of register address. The only local mdio write access allowed is [9:7] of this register. 0 = normal operation	remote_copper_mo de _def * mode_sel_def[1] * mode_sel_def[0] * not disable_remote_acc es s_def
7	RESTART SERDES AUTONEG	R/W SC	On Writes: 1 = restart SerDes auto-negotiation. Bit [9] of this register must also be set. 0 = normal operation On Reads: 1 = "gbic mode" (reg 1ch shadow 1f [2:1] = "11") and "remote copper mode" ([9] of this register = 1)	0

Bit	*Name*	*R/W*	*Description*	*Default*
6	REGISTER WRITE SELECTOR	R/W	On Writes: 1 = write bits [9:7] when bit [15] is also set during a write cycle 0 = write bits [5:0] when bit [15] is also set during a write cycle On Reads: 1 = “SerDes link” (reg 1ch shadow 15h[9] = 1) and “remote_copper_mode” ([9] of this register = 1)	0
5	ENABLE AMPLITUDE SIGNAL DETECT	R/W	1 = SerDes synchronization will fail if signal amplitude is not above a certain threshold. (useful in applications that do not use a fiber module and the receiver may be floating) 0 = normal operation	enable_sd_amp_def 5478: tied to vdd at quad level
4	SNOOP MODE (register 1c shadow 1fh bits[2:1] must be “00”)	R/W	1 = allow SerDes to act as snoop port. All receive traffic on copper pins will be forwarded to sgmii trans mit pins. 0 = disable snoop port	snoop_mode_def 5478: tied to gnd at quad level
3	FILTER FORCED LINK	R/W	1 = synchronization status must be valid for 10 ms before link will come up when auto-negotiation is disabled. 0 = normal operation	filter_forced_link_def (tied to vdd at gphy_g6 level)
2	DISABLE FALSE LINK	R/W	1 = do not allow xmit=data when auto-negotiation is disabled unless rudi=idle detected. Force xmit=idle if rudi=config. 0 = normal operation	1
1	SERDES AUTO-NEGOTIATION PARALLEL DETECT ENABLE	R/W	1 = turn auto-negotiation on/off in order to link up with link partner. Algorithm based on received code words 0 = disable parallel detection	fiber_parallel_det_def 5478: tied to vdd at quad level
0	FIFO ELASTICITY [1] for dig1000x_tx_fifo and dig1000x_rx_fifo Modes: (rgmii/gmii pads towards serdes link partner) (rgmii pads to sgmii-slave 10/ 100/1000 towards serdes link partner) (10/100 SGMII transmit and receive) (1000 SGMII transmit from copper link partner towards serdes link partner)	R/W	dig1000x_tx_fifo and dig1000x_rx_fifo FIFO ELASTICITY [1:0]: 11 = support 20k byte packets 10 = support 15k byte packets 01 = support 10k byte packets 00 = support 5k byte packets (w/200 ppm offset) LSB located at register 1ch shadow 1bh [1]	jumbo_packet_def[1] 5478: tied to gnd at quad level

1Ch: Misc 1000-X Control Register (Shadow Register Selector = "17h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	10111
9	QUALIFY DISABLE CARRIER EXTEND AND DISABLE TRRR WITH FULL-DUPLEX	R/W	1 = In full-duplex mode : disable carrier extension when reg 1ch shadow 1bh [6] = 1 and disable trrr generation when reg 1ch shadow 1bh [5] = 1 In half-duplex: always allow carrier extension and trrr generation regardless of reg 1ch shadow 1bh [6:5]. 0 = Always disable carrier extension when reg 1ch shadow 1bh [6] = 1 and disable trrr generation when reg 1ch shadow 1bh [5] = 1	0
8	FIBER AUTO POWERDOWN WAKE UP TIME	R/W	1 = wake up for 250 ms before powering down NOTE: (set to 1 when enabling fiber auto-power down or 100-fx auto-detection; register 1ch shadow "1eh" [4] and register 1ch shadow "13h" [2]) 0 = wake up for 42 ms before powering down	0
7	FIBER AUTO POWERDOWN SLEEP TIME	R/W	1 = power down for 3 seconds before waking up 0 = power down for 5 seconds before waking up	0
6	SERDES TRANSMIT DISABLE	R/W	1 = force all SerDes transmit data to 0 0 = normal operation	0
5	SIGNAL DETECT ENABLE	R/W	1 = force synchronization to fail if signal detect is not active. Disabled if register 1ch shadow 14h [1] = 1. 0 = ignore signal detect pin	enable_sd_pin_def
4	DISABLE GBIC UPDATES	R/W	1 = use register 4 and 9 for copper auto-neg, register 4 for SerDes autoneg. Do not allow gbic updates. 0 = allow registers 4 and 9 to update in gbic mode	0
3	FORCE XMIT=DATA	R/W	1 = force xmit=data regardless of state of receive channel 0 = normal operation	force_xmit_data_def 5478: tied to gnd at top level

Bit	*Name*	*R/W*	*Description*	*Default*
2	DISABLE SERDES AMPLITUDE SWITCHING	R/W	1 = SerDes amplitude will be fixed to operate with fiber modules (1000x) 0 = normal operation (sgmii or 1000x amplitudes based on current mode of operation)	0
1	FIBER SUPER-ISOLATE	R/W	1 = Fiber super-isolate	0
0	DISABLE 1000-X POWERDOWN	R/W	1 = disable1000-X power-down from auto-medium detection and fiber auto-power down. (register 0 power-down not affected) 0 = normal operation	0

1Ch: Auto-Detect SGMII/GBIC Register (Shadow Register Selector = "18h")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11000
9	SERDES RESOLUTION FAULT	RO	1 = selector field mismatch (link partner base page code word bit 0 mismatches local base page) 0 = no mismatch has occurred or auto-detect SGMII/GBIC mode is disabled	0
8	XMIT=DATA (SERDES only)	RO	1 = SerDes side is in the xmit=data state 0 = SerDes side is in the xmit=config or idle state	0
7	FIBER SD (directly from pin, no inversion or filtering)	RO	1 = fiber signal detect from pin is high 0 = fiber signal detect from pin is low	0
6	SD AMPLITUDE STATUS	RO	1 = signal detect amplitude is above the minimum threshold 0 = signal detect amplitude is below the minimum threshold	0
5	SD AMPLITUDE STATUS CHANGED	RO LH	1 = signal detect amplitude status has changed since last read 0 = signal detect amplitude status has not changed since last read	0
4	ENABLE CRC FRAGMENT ERRORS	R/W	1 = enable SerDes crc checker to count fragments as crc errors 0 = ignore fragments as crc errors	0

Bit	Name	R/W	Description	Default
3	MEDIA CONVERTER MODE (register 1c shadow 1fh bits [2:1] must be "11")	R/W	1 = do not allow copper side to link up until fiber SD is set and SerDes synchronization is valid. (fiber SD is not used when "select SD" in register 1ch shadow 14 [1] = 1) 0 = normal GBIC operation	media_converter_def 5478: tied to gnd at quad level
2	GMII FIFO ELASTICITY [0] in sgmii/gbic mode (1000T PCS transmit FIFO in SGMII/GBIC mode : from serdes to copper link partner)	R/W	GMII FIFO ELASTICITY [1:0] in sgmii/gbic mode 11 = Support 18k byte packets 10 = support 15k byte packets 01 = support 10k byte packets 00 = support 5k byte packets (w/200 ppm offset) MSB is located at expansion reg 46[15]	jumbo_packet_def[0]
1	SGMII 10/100 RX FIFO FREQUENCY LOCK MODE (forced high internally when => sgmii_slave_mode and not remote_lpbk)	R/W	1 = SGMII RX FIFO will assume that the SerDes recovered clock and the local clock are frequency locked. This will essentially bypass the FIFO with the lowest possible latency in 10/100 speeds (useful for applications where the MAC/switch and PHY are using the same crystal) 0 = normal operation	0
0	AUTO-DETECT SGMII/ GBIC MODE	R/W	1 = enable auto-detection between SGMII and GBIC modes 0 = normal operation	autodet_sgmii_gbic

1Ch: Test 1000-X Register (Shadow Register Selector = "19h")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11001
9	USE COPPER SPEED	R/W	1 = force dig1000x speed to copper speed (used to test FIFOs in 10/100 mode from gmii pins) 0 = normal operation	0
8	DLL BYPASS CLOCK ENABLE	R/W	1 = enable dll bypass clock (tpin 11) 0 = disable dll bypass clock	0

Bit	*Name*	*R/W*	*Description*	*Default*
7	TX FIFO SGMII 10/100 EARLY PREAMBLE MODE => towards SerDes link partner	R/W	1 = enable transmitting early preamble in sgmii 10/ 100 mode in order to reduce latency in half-duplex (tx FIFO) 0 = normal operation	0
6	RX FIFO SGMII 10/100 EARLY PREAMBLE MODE => towards copper link partner	R/W	1 = enable transmitting early preamble in sgmii 10/ 100 mode in order to reduce latency in half-duplex (rx FIFO) 0 = normal operation	0
5	BLOCK TXEN MODE FOR dig1000x_tx_fifo => towards SerDes link partner	R/W	1 = block txen for 7 bytes after end of packet in 1000-X mode, 6.5 bytes in sgmii 10/100 0 = normal operation	0
4	RESERVED	R/W	write as 0, ignore on read	0
3	FORCE TXFIFO ON (dig1000x_tx_fifo) towards SerDes link partner	R/W	1 = force TXFIFO to be always active in 1000X 0 = normal operation	0
2	BYPASS PCS RECEIVE	R/W	1 = bypass pcs receive 0 = normal operation	0
1	BYPASS PCS TRANSMIT	R/W	1 = bypass pcs transmit 0 = normal operation	0
0	ZERO COMMA DETECT PHASE	R/W	1 = force comma detect phase to zero 0 = normal operation	0

1Ch: Autoneg 1000-X Debug Register (Shadow Register Selector = "1ah")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11010
9	CONSISTENCY MISMATCH	RO LH	1 = consistency mismatch occurred since last read 0 = no consistency mismatch occurred since last read	0
8	RUDI INVALID	RO LH	1 = rudi invalid detected since last read 0 = no rudi invalid detected since last read	0

Bit	Name	R/W	Description	Default
7	SYNC_STATUS DETECTED	RO LH	1 = sync_status detected since last read 0 = no sync_status detected since last read	0
6	LINK WENT DOWN FROM LOSS OF SYNC	RO LH	1 = a valid link went down due to a loss of synchronization for over 10 ms. 0 = failure condition has not been detected since last read	0
5	IDLE DETECT STATE	RO LH	1 = idle detect state entered since last read 0 = idle detect state has not been entered since last read	0
4	COMPLETE ACKNOWLEDGE STATE	RO LH	1 = complete acknowledge state entered since last read 0 = complete acknowledge state has not been entered since last read	0
3	ACKNOWLEDGE DETECT STATE	RO LH	1 = acknowledge detect state entered since last read 0 = acknowledge detect state has not been entered since last read	0
2	ABILITY DETECT STATE	RO LH	1 = ability detect state entered since last read 0 = ability detect state has not been entered since last read	0
1	ERROR STATE (reg 1ch shadow 27 [3] = 1)	RO LH	1 = error state entered since last read 0 = error state has not been entered since last read	0
	SYNC_STATUS FAILED (reg 1ch shadow 27 [3] = 0)	RO LH	1 = sync_status failed since last read 0 = sync_status has not failed since last read	0
0	AN_ENABLE STATE	RO LH	1 = an_enable state entered since last read 0 = an_enable state has not been entered since last read	0

1Ch: Auxiliary 1000-X Control Register (Shadow Register Selector = "1bh")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11011
9	USE SERDES MODE COUNTERS	R/W	1 = use registers 12-14h for SerDes data 0 = normal operation	0

Bit	Name	R/W	Description	Default
8	AUTONEG FAST TIMERS	R/W	1 = speed up link_timer for test vectors (1.6 us SGMII; 9.84 us IEEE) 0 = normal operation	0
7	JAM FALSE CARRIER MODE	R/W	1 = Transmit packet on SerDes pins with txen, txer, txd=55h for duration of false carrier received on copper pins in sgmii/gbic half-duplex mode 0 = ignore false carriers in sgmii/gbic mode	1
6	DISABLE CARRIER EXTEND	R/W	1 = force rxer, rxd to zeros in TRR+extend state (pcs receive state) 0 = normal operation	0
5	DISABLE TRRR	R/W	1 = bypass extend_by_1 state (pcs transmit state) 0 = normal operation	0
4	DISABLE REMOTE FAULT SENSING	R/W	1 = disable automatic remote fault sensing of autoneg resolution errors and offline errors. (offline errors occur in gbic mode when the copper link is down) 0 = normal operation	0
3	AUTONEG ERROR TIMER ENABLE	R/W	1 = enable autoneg error timer (error state entered when error timer expires in ability_detect, acknowledge_detect, or idle_detect state) 0 = normal operation	0
2	COMMA DETECT ENABLE	R/W	1 = enable comma detection 0 = disable comma detection	1

Bit	*Name*	*R/W*	*Description*	*Default*
1	FIFO ELASTICITY [0] for dig1000x_tx_fifo and dig1000x_rx_fifo Modes: (rgmii/gmii 1000x towards SerDes link partner) (rgmii to sgmii-slave 10/ 100/ 1000 towards SerDes link partner) (10/100 SGMII transmit and receive) (1000 SGMII transmit from copper link partner towards SerDes link partner) FIFO ELASTICITY [0] for dig1000x_tx_fifo and dig1000x_rx_fifo (1000-X PCS transmit, 10/100 SGMII transmit and receive, 1000 SGMII transmit from copper link partner towards SerDes MAC)	R/W	dig1000x_tx_fifo and dig1000x_rx_fifo FIFO ELASTICITY [1:0]: 11 = support 20k byte packets 10 = support 15k byte packets 01 = support 10k byte packets 00 = support 5k byte packets (w/200 ppm offset) MSB located at a register 1ch shadow 16h [0]	jumbo_packet_def [0]
0	DISABLE CRC CHECKER	R/W	1 = disable crc checker (clock gated-off to save power) 0 = enable crc checker	1

1Ch: Auxiliary 1000-X Status Register (Shadow Register Selector = "1ch")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11100
9	SERDES LINK STATUS CHANGE	RO LH	1 = link status change has occurred since last read 0 = link status change has not occurred since last read	0

Bit	*Name*	*R/W*	*Description*	*Default*
8	SGMII SELECTOR MISMATCH	RO	1 = SGMII selector mismatch in SGMII mode 0 = Fiber, copper, GBIC mode, or SGMII selector does not mismatch, or autoneg disabled	0
7	AUTONEG RESOLUTION ERROR	RO	1 = autoneg hcd is none (no common half or full duplex abilities) 0 = SGMII mode, or autoneg disabled, or no resolution error	0
6:5	LINK PARTNER REMOTE FAULT	RO	1000-X register 5 [13:12]	00
4	AUTONEG PAGE RECEIVED	RO LH	1 = page has been received since last read 0 = page has not been received since last read	0
3	CURRENT OPERATING DUPLEX MODE	RO	1 = phy is operating in full-duplex mode 0 = phy is operating in half-duplex mode (or auto- negotiation has not completed)	0
2	SERDES LINK	RO	1 = link is up for SerDes applications (sgmii or gbic mode => both SerDes and copper must have link for this bit to be set) 0 = link is down for SerDes applications	0
1	PAUSE RESOLUTION - RECEIVE SIDE	RO	1 = enable pause receive 0 = disable pause receive	0
0	PAUSE RESOLUTION- TRANSMIT SIDE	RO	1 = enable pause transmit 0 = disable pause transmit	0

1Ch: Misc 1000-X Status Register (Shadow Register Selector = "1dh")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11101
9	TX FIFO ERROR	RO LH	1 = transmit FIFO error since last read 0 = no transmit FIFO error since last read	0
8	RX FIFO ERROR	RO LH	1 = receive FIFO error since last read 0 = no receive FIFO error since last read	0
7	BAD FIFO POINTER	RO LH	1 = FIFO pointer all zeros since last read 0 = bad FIFO pointer has not occurred since last read	0
6	FALSE CARRIER JAMMED (TX SIDE)	RO LH	1 = false carrier detected on copper receiver and jammed in SGMII/GBIC mode to SerDes transmitter since last read 0 = no false carrier jammed or mode is disabled via register 1ch shadow 1bh [7]	0
5	FALSE CARRIER DETECTED (RX SIDE)	RO LH	1 = false carrier detected on SerDes receiver since last read 0 = no false carriers detected since last read	0
4	CRC ERROR DETECTED	RO LH	1 = crc error detected since last read 0 = no crc error detected since last read or mode is disabled via register 1ch shadow 1bh [0]	0
3	TRANSMIT ERROR DETECTED	RO LH	1 = transmit error code detected since last read (rx_data_error state in pcs receive) 0 = no transmit error code detected since last read	0
2	RECEIVE ERROR DETECTED	RO LH	1 = receive error since last read (early_end state in pcs receive) 0 = no receive error since last read	0
1	CARRIER EXTEND ERROR DETECTED	RO LH	1 = carrier extend error since last read (extend_err state in pcs receive) 0 = no carrier extend error since last read	0
0	EARLY END EXTENSION DETECTED	RO LH	1 = early end extension since last read (early_end_ext state in pcs receive) 0 = no early end extension since last read	0

1Ch: Auto-Detect Medium Register (Shadow Register Selector = "1eh")

Bit	Name	R/W	Description	Default
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11110
9	2ND SERDES AUTO-DETECTION	R/W	1 = enable dual SerDes auto-detect medium (switch between sgmii->fiber and sgmii->copper) 0 = disable dual SerDes auto-detect medium	autodet_med_dual 5478: tied to gnd at top level
8	INVERT FIBER SD FROM PIN	R/W	1 = invert fiber sd from pin (active low pin) 0 = normal operation (active high pin)	invert_sd_pin_def 5478: tied to gnd at top level
7	FIBER IN-USE LED MODE	R/W	1 = drive transmit led active low when fiber is selected, inactive when copper selected 0 = normal transmit led operation	autodet_med_pindef and not mode_sel_def[1] 5478:autodet_med_pindef tied to gnd at top level
6	FIBER LED MODE	R/W	1 = use fiber transmit, receive, and link for leds whenever fiber mode is selected via register 1ch shadow 1fh bits [2:1] 0 = always use copper transmit, receive, and link for leds regardless of the mode selected	(autodet_med_pindef or mod_sel_def[0]) and not mode_sel_def[1] 5478:autodet_med_pindef tied to gnd at top level
5	QUALIFY FIBER SD WITH SYNC_STATUS/100FX LINK	R/W	1 = fiber signal detect from pin is ANDed with sync_status in 1000-x and link in 100-fx mode. (Used only for auto-detect medium and dual SerDes auto-detection.) 0 = normal operation	1
4	FIBER AUTO-POWER DOWN MODE (register 1c shadow 1fh bit 2 must be 0 unless a fiber module is used in media converter mode)	R/W	1 = power-down fiber when signal detect is inactive (wake-up for 42ms every 5 seconds to transmit code words; see register 1ch shadow 17h [8:7] for different time options). 0 = normal operation	fiber_autopwrdn_def 5478:tied to gnd at top level
3	SIGNAL DETECT ENABLE OVERRIDE	R/W	1 = force signal detect enable to 1 in auto-detect medium block. Used for test purposes only! 0 = normal operation	0

Bit	*Name*	*R/W*	*Description*	*Default*
2	AUTO-DETECT MEDIUM DEFAULT	R/W	1 = fiber selected when no medium active 0 = copper selected when no medium active	0
1	AUTO-DETECT MEDIUM PRIORITY	R/W	1 = fiber selected when both medium active 0 = copper selected when both medium active	1
0	AUTO-DETECT MEDIUM ENABLE	R/W	1 = enable auto-detect medium (switch between gmii/rgmii->copper or gmii/rgmii->fiber) 0 = disable auto-detect medium	autodet_med_pindef and not mode_sel_def[1] 5478:autodet_med_pindef tied to gnd at top level

1Ch: Mode Control Register (Shadow Register Selector = "1fh")

Bit	*Name*	*R/W*	*Description*	*Default*
15	WRITE ENABLE	R/W	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	R/W	Shadow Register Selector	11111
9	DUAL SERDES CAPABLE	RO	1 = chip supports sgmii->fiber mode 0 = chip does not support sgmii->fiber mode	dual_serdes_capable 5478: tied to gnd at top level
8	MODE SELECT CHANGE	RO LH	1 = mode select change since last read 0 = no mode select change since last read	0
7	COPPER LINK	RO	1 = link up on copper side (copper, sgmii, or gbic mode) 0 = link down	0
6	SERDES LINK	RO	1 = link up in fiber, sgmii,or gbic modes (when set in sgmii or gbic mode, then both copper and SerDes link must be valid) 0 = link down	0
5	COPPER ENERGY DETECT	RO	1 = copper energy detected 0 = no copper energy detected	0
4	FIBER SIGNAL DETECT and SYNC_STATUS (filtered) (reg 1ch shadow 1eh [5] = 1)	RO	1 = fiber signal detect from pin and code group synchronization (filtered). (In 100-FX mode use link status instead of synchronization) 0 = no fiber signal detect from pin	0

Bit	Name	R/W	Description	Default
	FIBER SIGNAL DETECT (filtered) (reg 1ch shadow 1eh [5] = 0)	RO	1 = fiber signal detect from pin (filtered) 0 = no fiber signal detect from pin	0
3	SERDES CAPABLE	RO	1 = SerDes capable device 0 = not SerDes capable device	GPHY input: SerDes_capable
2:1	MODE SELECT	R/W	00 = copper-qsgmii 01 = fiber-qsgmii 10 = sgmii-copper 11 = gbic	mode_sel_def[1:0]
0	ENABLE 1000-X REGISTERS	R/W	1 = select 1000-X regs 0-0fh 0 = select copper regs 0-0fh	mode_sel_def[1:0] = 01

1Dh: Master/Slave Seed Register (Bit 15 = 0)

Bit	Name	R/W	Description	Default
15	ENABLE SHADOW REGISTER Writes to the selected register are done on a single cycle (no setup required).	R/W	1 = select shadow register 0 = normal operation	0
14	MASTER/SLAVE SEED MATCH	RO LH	1 = Seeds match 0 = Seeds don't match (Value not latched when bit9 in reg 1Fh is set)	0
13	LINK PARTNER REPEATER/ DTE BIT	RO*	1 = link partner is a repeater/switch device port 0 = link partner is a DTE device port	0
12	LINK PARTNER MANUAL M/S CONFIG VALUE	RO*	1 = link partner is configured as master 0 = link partner is configured as slave	0
11	LINK PARTNER MANUAL M/S CONFIG ENABLE	RO*	1 = link partner manual master/slave configuration enabled 0 = link partner manual master/slave configuration disabled	0
10:0	LOCAL MASTER/ SLAVE SEED VALUE	R/W	returns the automatically generated master/ slave random seed when bits 9 and 11 in reg 1Fh are cleared. Writeable value is used for master/slave seed when bit 11 in reg 1Fh is set and the link partner base page is received; writable value is used immediately when bit 9 in reg 1Fh is set.	000h
* R/W when "writeable link partner ability test mode" (reg 1Fh bit 10) is set				

1Dh: HCD Status Register (Bit 15 = 1)

Bit	*Name*	*R/W*	*Description*	*Default*
15	ENABLE SHADOW REGISTER Writes to the selected register are done on a single cycle (no setup required).	R/W	1 = select shadow register 0 = normal operation	0
14	WIRESPEED DISABLE GIGABIT ADVERTISING	RO	1 = disable advertising gigabit 0 = advertise gigabit based on register 9	0
13	WIRESPEED DISABLE 100TX ADVERTISING	RO	1 = disable advertising 100tx 0 = advertise 100tx based on register 4	0
12	WIRESPEED DOWN GRADE	RO LH	1 = wirespeed downgrade occurred since last read 0 = wirespeed downgrade cleared	0
11	* HCD 1000T FDX	RO LH	1 = gigabit full-duplex hcd occurred since last read 0 = hcd cleared	0
10	* HCD 1000T	RO LH	1 = gigabit half-duplex hcd occurred since last read 0 = hcd cleared	0
9	* HCD 100T FDX	RO LH	1 = 100tx full-duplex hcd occurred since last read 0 = hcd cleared	0
8	* HCD 100T	RO LH	1 = 100tx half-duplex hcd occurred since last read 0 = hcd cleared	0
7	* HCD 10T FDX	RO LH	1 = 10base-t full-duplex hcd occurred since last read 0 = hcd cleared	0
6	* HCD 10T	RO LH	1 = 10base-t half-duplex hcd occurred since last read 0 = hcd cleared	0
5	* HCD 1000T FDX (Link never came up)	RO LH	1 = gigabit full-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0
4	* HCD 1000T (Link never came up)	RO LH	1 = gigabit half-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0
3	* HCD 100T FDX (Link never came up)	RO LH	1 = 100tx full-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0
2	* HCD 100T (Link never came up)	RO LH	1 = 100tx half-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0
1	* HCD 10T FDX (Link never came up)	RO LH	1 = 10base-t full-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0

Bit	Name	R/W	Description	Default
0	* HCD 10T (Link never came up)	RO LH	1 = 10base-t half-duplex hcd and link never came up occurred since last read 0 = hcd cleared	0
* Bits [12:0] also cleared on restarting or disabling auto-negotiation.				

1Eh: Test1_Register

Bit	Name	R/W	Description	Default
15	CRC ERROR COUNT VISIBILITY	R/W	1 = receiver NOT_OK counters merged into one 16 bit counter to count CRC errors instead (crc errors will only be counted after this bit is set!) 0 = normal operation	0
14	TRANSMIT ERROR CODE VISIBILITY	R/W	1 = false carrier sense counter counts packets received with transmit error codes instead (errors will only be counted after this bit is set!) 0 = normal operation	0
13	COUNTER TEST MODE	R/W	1 = forces counters into test mode 0 = normal operation	0
12	FORCE LINK	R/W	1 = force link state machine into pass state 0 = normal operation	0
11	FORCE LOCK	R/W	1 = force descrambler in to locked state 0 = normal operation	0
10	SCRAMBLER TEST	R/W	1 = speed up descrambler unlock detect timer 0 = normal operation	0
9	EXTERNAL LINK	R/W	1 = use tpin11 input as link status 0 = normal operation	0
8	FAST TIMERS	R/W	1 = timers are sped up for LSI test 0 = normal operation	0
7	MANUAL SWAP MDI STATE	R/W	1 = Swap 0 = off	0
6	RECEIVE WATCHDOG TIMER DISABLE	R/W	1 = watchdog timer disabled 0 = reset receive PMD when descrambler can't lock within 730us of link or lock loss.	0
5	DISABLE POLARITY ENCODE	R/W	1 = disable 1000BASE-T polarity encoding 0 = normal operation	0

Bit	*Name*	*R/W*	*Description*	*Default*
4	ENABLE SOFTWARE TRIM SETTING (MAIN DAC)	R/W	1 = use software trim setting 0 = use hardware trim setting Notes: 1) register settings from slice1, slice8 are used to control main dac 2) register settings from slice2, slice7 are used to control hybrid dac	0
3:0	TRIM[3:0] (MAIN DAC)	R/W	Software trim setting Notes: 1) only slice1, slice8 register setting are used. 2) Main dac value going to BIAS block (when enable software trim setting = 0) is fuse_in(main dac)+011(default)+software trim setting (main dac)	0

1Fh: Test2_Register

Bit	*Name*	*R/W*	*Description*	*Default*
15:13	TEST SELECT AUTONEG FSM	R/W	000 = ARB 001 = RX1000 010 = RX 011 = TX1000 100 = TX 101 = BASET LINK	000
12	TEST AUTO-NEG TIMER	R/W	1 = auto-negotiation timer test mode 0 = normal operation	0
11	TEST MASTER/SLAVE SEED	R/W	1 = use MDIO programmable master/slave seed 0 = normal operation	0
10	WRITEABLE LINK PARTNER ABILITY	R/W	1 = link partner advertised ability may be overwritten by MII management 0 = normal operation	0
9	FORCE HCD	R/W	1 = force auto-negotiation HCD resolution (hcd can only be checked in register 19h bits [10:8]; hcd status register will not be updated) 0 = normal operation	0
8	WRITEABLE LINK PARTNER M/S SEED	R/W	1 = link partner master/slave seed may be overwritten by MII management 0 = normal operation	0
7	TRANSMIT 10B MODE	R/W	1 = force GMII transmit into 10B mode 0 = use normal mode bit	0

Bit	*Name*	*R/W*	*Description*	*Default*
6	RECEIVE 10B MODE	R/W	1 = force GMII receive into 10B mode 0 = use normal mode bit	0
5	BYPASS TRANSMIT FIFO modes: (rgmii 10/100/1000 copper) (sgmii 1000 towards copper link partner) (rgmii 100fx towards SerDes link partner)	R/W	1 = transmit FIFO bypassed 0 = normal operation	0
4	SAME SCRAMBLER SEEDS	R/W	1 = receive scrambler uses transmit seed 0 = normal operation	0
3	JITTER TEST MODE	R/W	1 = jitter test mode 0 = normal operation	0
2	TEST ATMP COUNTER	R/W	1 = force master slave seed attempt counter into test mode 0 = normal mode	0
1	LATENCY MEASURE	R/W	1 = send special short packet to measure receive latency on remote PHY (hold high) 0 = normal operation	0
0	DISABLE ACTIVE HYBRID	R/W	1 = active hybrid disabled 0 = active hybrid enabled	0

Clause 45 Registers

Clause 45 registers are accessed in the Clause 22 register space according to the method in Clause 22.2.4.3.11-12 and Annex 22D.

> ***Example:*** To advertise that EEE mode is supported for 1000BASE-T and 100BASE-TX, Clause 45 Dev 7.3Ch bits 2:1 must be set to 11. To perform this write, the following sequence may be used:
>
> - In register 0Dh, write bits 15:14 = 00 (set the function field to Address) and set bits 4:0 to '00111' (set DEVAD to 7) (write 007h to Register 0Dh).
> - In register 0Eh, write the desired address value. In this example, write 003Ch to Register 0Eh
> - In register 0Dh, write the Function field (bits 15:14) to 01 for Data, no post increment. Set DEVAD to 7. (Write 4007h to Register 0Dh)
> - In register 0Eh, write the content of the register. (Write to Register 0Eh to 0006h to set bits 2:1 = 11)

To read the EEE Resolution Status Register (Clause 45 DEV 7 Reg803Eh), the following register sequence may be followed:

- Write bits 15:14 to 00 to register 0Dh to set the function field to Address, and set bits 4:0 to '00111' to set DEVAD to 7
- Write 803Eh to register 0Eh to select the EEE Resolution Status Register

- Write 4007h to register 0Dh to set the Function to Data, no post increment and to set the Device Address to 7
- Read register 0Eh to read the content of the selected Clause 45 register (EEE Resolution Status Register in this case).

Clause 45 Register Dev 3 Reg14h (20d): EEE Capability Register

Table 106: Clause 45 Register Dev 3 Reg14h: EEE Capability Register

Bit	Name	R/W	Description	Default
15:3	Reserved	RO	ignore on read	0
2	1000BASE-T EEE	RO	1 = EEE is supported for 1000BASE-T 0 = EEE is not supported for 1000BASE-T	0
1	100BASE-TX EEE	RO	1 = EEE is supported for 100BASE-TX 0 = EEE is not supported for 100BASE-TX	0
0	Reserved	RO	Write as 0, ignore on read	0000h

Clause 45 Register Dev 7 Reg3ch (60d): EEE Advertisement Register

Table 107: Clause 45 Register Dev 7 Reg3Ch: EEE Advertisement Register

Bit	Name	R/W	Description	Default
15:3	Reserved	RW	Write as 0, ignore on read.	0
2	1000BASE-T EEE	RW	1 = Advertise PHY has EEE capability for 1000BASE-T. 0 = Do not advertise PHY has EEE capability for 1000BASE-T.	0
1	100BASE-TX EEE	RW	1 = Advertise PHY has EEE capability for 100BASE-TX. 0 = Do not advertise PHY has EEE capability for 100BASE-TX.	0
0	Reserved	RW	Write as 0, ignore on read.	0000h

1000BASE-T EEE

This bit, when set, turns on advertisement for 1000BASE-T EEE mode. EEE mode is turned off by default. Register 09h bits 9:8 must be set to advertise 1000BASE-T mode and auto-negotiation must be restarted for EEE mode to take effect.

100BASE-TX EEE

This bit, when set, turns on advertisement for 100BASE-TX EEE mode. EEE mode is turned off by default. Register 04h bits 8:7 must be set to advertise 100BASE-TX mode and auto-negotiation must be restarted for EEE mode to take effect.

Clause 45 Register Dev 7 Reg803Eh (32830d): EEE Resolution Status

Table 108: Clause 45 Register Dev 7 Reg803Eh: EEE Resolution Status

Bit	Name	R/W	Description	Default
15:3	Reserved	R/W	Write as 0, ignore on read	0
2	EEE 1000BASE-T Resolution	R/W	1 = Both local device and link partner advertise EEE 1000BASE-T capability 0 = No EEE capability in local device or link partner in 1000BASE-T mode	0
1	EEE 100BASE-TX Resolution	R/W	1 = Local and remote link partner advertise EEE 100BASE-TX mode 0 = No EEE capability in either local or link partner in 100BASE-TX mode	0
0	Reserved	R/W	Write as 0, ignore on read	0000h

EEE 1000BASE-T Resolution

This bit returns the status of the resolution of EEE 1000BASE-T mode. If this bit is set, PHY and remote receiver have auto-negotiation enabled, and both PHY and link partner advertise EEE 1000BASE-T mode. If this bit is cleared, either the local or remote PHY does not have auto-negotiation enabled, or local or remote PHY does not advertise EEE capability in 1000BASE-T mode.

EEE 100BASE-TX Resolution

This bit returns the status of the resolution of EEE 100BASE-TX mode. If this bit is set, PHY and remote receiver have auto-negotiation enabled, and both PHY and link partner advertise EEE 100BASE-TX mode. If this bit is cleared, either the local or remote PHY does not have auto-negotiation enabled, or local or remote PHY does not advertise EEE capability in 100BASE-TX mode.

Clause 45 Register Dev 7 Reg803dh (32817d): EEE Control Register

Table 109: Clause 45 Register Dev 7 Reg803dh: EEE Control Register

Bit	Name	R/W	Description	Default
15	LPI Feature Enable	R/W*	1 = Enable LPI Feature 0 = Disable LPI Feature	0
14	Enables EEE capability using SGMII auto-negotiation	R/W	1 = Enable EEE capability using SGMII auto-negotiation 0 = Do not enable EEE capability using SGMII auto-negotiation	0
13	Reserved	R/W	Write as 0, ignore on read.	0
12	Reserved	R/W	Write as 0, ignore on read.	0
11:0	Reserved	R/W	Write as 0, ignore on read.	0000h

LPI Feature Enable

Setting this bit high enables LPI.

Top Level MII Registers

Top Level MII Registers use the MDIO interface for access. The registers are access through the following registers:

- Address 1Ch, shadow 0Bh of port 5: Top Level MII Access register: used to issue a write/read command and give register offset for the Top Level register to be accessed.
- Address 1Ch, shadow 0Ch of port 6: Top Level MII Data register: used to write 8-bit data to the Top Level register or read 8-bit data back from Top Level register.
- Expansion register 0Bh of port 3: Top Level MII Status register.

Register Write Sequence

A write command is done by:

- Writing B0xxh to Port 6's register 1Ch, where xx is data to be written to the Register Offset.
- Writing {0xAC} to bit [15:8], write {1} to bit 7 and "offset address" to bit [6:0] of Port 5's register 1Ch.
  - This sets bit 7 = 1 to enable a write command and sets bits[6:0] = "Offset Address" for the register to be written.

Register Read Sequence

A read command is done by:

- Writing {0xAC} to bit [15:8], write {0} to bit 7 and "offset address" to bit [6:0] of Port 5's register 1Ch.
  - This sets bit 7 = 0 to enable a read command and sets bits[6:0] = "Offset Address" for the register to be written.
- Read Exp_Reg 0Bh, bit [7:0] of port 3.

Offset Address 00h [7:0]: SyncE Recovered Clock Register

Bit	*Name*	*R/W*	*Description*	*Default*
7	REC_CLK2 Disable	R/W	1 = disable REC_CLK2 output.	1

Bit	Name	R/W	Description	Default
6:4	REC_CLK2 Port Selection	R/W	Select port which REC_CLK2 output is derived from. 000 = port 1 001 = Port 2 010 = Port 3 011 = Port 4 100 = Port 5 101 = Port 6 110 = Port 7 111 = Port 8	000
3	REC_CLK1 Disable	R/W	1 = disable REC_CLK1 output.	1
2:0	REC_CLK1 Port Selection	R/W	Select port which REC_CLK1 output is derived from. 000 = port 1 001 = Port 2 010 = Port 3 011 = Port 4 100 = Port 5 101 = Port 6 110 = Port 7 111 = Port 8	000

Offset Address 01h [15:8]: Temperature Monitor Register

Bit	Name	R/W	Description	Default
7:1	Temperature Monitor Bias Adjust	R/W	Analog Bias Adjust	0000000
0	Temperature Monitor Enable	R/W	1: Enable temperature monitor	0

Offset Address 02h [23:16]: OTP Address Register

Bit	Name	R/W	Description	Default
7	ShadowPHYAddress Select	R/W	This bit is only meaningful for BCM54685. 1: enable Shadow PHY address mode. 0: Disable Shadow PHY address mode.	tpin[9]
6	Select ShadowPHY address	R/W	When top_miireg[23] is enabled. 1: assign phya+7 address to the 8th slice of GPHY 0: assign phya+7 address to QSGMII	0

Bit	Name	R/W	Description	Default
5:3	OTP Data Select	R/W	OTP controller 3-bit data selector on which bank of 32 bits will be read.	0
2:0	OTP Controller address	R/W	3 bit OTP controller address extension	0

Offset Address 03h [31:24]: Tune LPF and TH Pole Register

Bit	Name	R/W	Description	Default
7	Th_pole_reg_ovr	R/W	Enable overwrite th_pole_reg value	0
6	Th_pole_reg_val	R/W	th_pole_reg value	0
5	Tune_lpf_reg_ovr	R/W		0
4:0	Tune_lpf_reg_val[4:0]	R/W		0

Offset Address 04h [39:32]: LED Bias Settings Register

Bit	Name	R/W	Description	Default
7:4	Led_bias1_trim_val	R/W	–	0
3:0	Led_bias0_trim_val	R/W	–	0

Offset Address 05h [47:40]: LED/Misc Control Register

Bit	Name	R/W	Description	Default
7	Legacy Current Mode Enable	R/W	0 = Normal Current Mode. 1 = Legacy Current Mode. When in this mode the LEDs will function the same as Parallel LED mode but with current mode turned on.	0
6:5	Sled_bank_size	R/W	–	0
4:2	Reserved	R/W	–	–
1	Force LED powerdown	R/W	–	0
0	Led_bias0_trim_ovr	R/W		0

Offset Address 06h [55:48]: Frequency Counter Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:6	Reserved	R/W	–	–
5	Cntout_sel	R/W	Output counter select between osc or reference	0
4	Clkosc_sel	R/W	Input oscillator clock select	0
3	Counter_enable	R/W	Frequency counter enable	0
2:0	Pre_scaler_sel	R/W	Pre Scaler Input	0

Offset Address 0Ah [87:80]: Quad Auto Media Detection Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:6	Reserved	R/W	–	–
5	Invert RXC clock to QSGMII and SGMII	R/W	1= use rising edge to send traffic to QSGMII (BCM54685) or SGMII(BCM54640)	0
4	Overwrite package bonding to enable Quad GPHY mode	R/W	1 = Over write package bonding option and enable Quad GPHY mode. For testing only.	0
3:0	Enable Quad GPHY mode	R/W	When bit [4] is set, bit [3:0] is used to enable Quad Auto Media Detection mode.	0000

Offset Address 0Bh ~ 0Eh [119:88]: Voltage Monitor Configuration Register

Bit	*Name*	*R/W*	*Description*	*Default*
31:28	Reserved	R/W	–	0
27:6	Voltage Monitor Conf	R/W	Configuration of voltage monitor	0x004000
5:0	Voltage Monitor Conf	R/W	Configuration of voltage monitor. When top_miiregs, offset addr 16h, bit [0] (176th bit) = 1, this is used to configure voltage monitor, otherwise OTP trm value configure voltage monitor.	0

Offset Address 0Bh ~ 0Eh [119:88]: Voltage Monitor Configuration Register

Bit	*Name*	*R/W*	*Description*	*Default*
31:28	Reserved	R/W		0
27:6	Voltage Monitor Conf	R/W	Configuration of voltage monitor	–
5:0	Voltage Monitor Conf	R/W	Configuration of voltage monitor. When top_miiregs, offset addr 22h, bit [0] (176th bit) = 1, this is used to configure voltage monitor, otherwise OTP trm value configures voltage monitor.	0

Offset Address 0Fh [127:120]: Voltage Monitor Low Threshold_0 Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Voltage Monitor Low Threshold_0	R/W	There are total 9 bit of voltage monitor low threshold_0. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [0] (168th bit)	0

Offset Address 10h [135:128]: Voltage Monitor High Threshold_0 Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Voltage Monitor high Threshold_0	R/W	There are total 9 bit of voltage monitor high threshold_0. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [1] (169th bit)	0

Offset Address 11h [143:136]: Voltage Monitor Low Threshold_1 Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Voltage Monitor Low Threshold_1	R/W	There are total 9 bit of voltage monitor low threshold_1. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [2] (170th bit)	0

Offset Address 12h [151:144]: Voltage Monitor High Threshold_1 Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Voltage Monitor high Threshold_1	R/W	There are total 9 bit of voltage monitor high threshold_1. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [3] (171th bit)	0

Offset Address 13h [159:152]: Temp Monitor Low Threshold Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Temp Monitor Low Threshold	R/W	There are total 9 bit of temp monitor low threshold. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [4] (172th bit)	0

Offset Address 14h [167:160]: Temp Monitor High Threshold Register

Bit	*Name*	*R/W*	*Description*	*Default*
7:0	Temp Monitor high Threshold	R/W	There are total 9 bit of temp monitor high threshold. Bit [7:0] are set by this register. Bit 8 is from top_miiregs, offset addr 15h, bit [5] (173th bit)	0

Offset Address 15h [175:168]: Voltage/Temp Monitor Threshold MSB Register

Bit	*Name*	*R/W*	*Description*	*Default*
7	Voltage Monitor interrupt mask for 3.3v	R/W	1 = interrupt masked (status bits still operate normally but do not generate interrupt output) 0 = interrupt enabled	1
6	Voltage Monitor interrupt mask for 1.2v	R/W	1 = interrupt masked (status bits still operate normally but do not generate interrupt output) 0 = interrupt enabled	1
5:0	MSB of voltage/temp monitor threshold value	R/W	See offset address 10h ~ 14h.	0

Offset Address 16h [183:176]: Voltage Monitor Control Register

Bit	*Name*	*R/W*	*Description*	*Default*
7	Temp monitor interrupt mask	R/W	1 = interrupt masked (status bits still operate normally but do not generate interrupt output) 0 = interrupt enabled	1
6	Temp monitor Strobe	R/W	0: continuos monitor 1: strobe monitor (test mode, not to disclose to customer)	0
5	Temp monitor powerdown	R/W	1: Powerdown Temp Monitor.	0
4:3	Voltage Monitor Mode Select	R/W	00: Monitor 1.2v 01: Monitor 3.3v 1x: Monitor both voltage The monitored 1.2v and 3.3v data are stored at exp_reg0B of slice 7 and slice 8.	00

Bit	Name	R/W	Description	Default
2	Voltage Monitor Strobe	R/W	0: continuos monitor 1: strobe monitor (test mode, not to disclose to customer)	0
1	Voltage Monitor Powerdown	R/W	1: Powerdown Voltage Monitor.	0
0	Voltage Monitor Trim Value Overwrite	R/W	See offset address 0Bh ~ 0Eh bit[5:0]	–

Section 14: Card-Reader GRC Registers

All registers in each Card-Reader (CR) function are mapped to the GRC Register Space Starting from 6000-61FF. The first 256B are unique to each Card-Reader Function and the next 256B from 6100-61FF are common to all Card-Reader Function and it is mainly mapped to the Card-Reader Bridge Functions. There are 3 Card-Reader Functions and SW can use bits 7:6 of Register 6800 to control which Card-Reader function that it wants to access.

Note: Refer to the CR standard register definitions for all CR registers not documented in this Programmer's Guide. One source for this information for SD cards can be found here: http://www.sdcard.org/developers/tech/host_controller/simple_spec.

Card-Reader Register Space (offset: 0x6100-62FF)

The registers in this space are mapped uniquely to each Card-Reader Function. Using bits(7:6) of register 6800 to allow Software to select the registers from which function to read.

SLOT1 Register Mapping

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x000	Slot1 CR IP registers	0x0000	Slot1 CR IP registers
–	Slot1 CR IP registers	–	Slot1 CR IP registers
0x00FF	Slot1 CR IP registers	0x00FC	Slot1 CR IP registers
0x0102	Reserved	0x0100	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x017E	Reserved	0x017C	Reserved
0x0182	Bridge Mode Control Register	0x0180	Bridge Mode Control Register
0x0186	Bridge SDMA Status Register	0x0184	Bridge SDMA Status Register
0x018A	Bridge ADMA Status Register	0x0188	Bridge ADMA Status Register
0x018E	GRC/AHB Status Register	0x018C	GRC Control Register
0x0192	IP Control/Status Register	0x0190	IP Control/Status Register
0x0196	Bridge Static Debug Register	0x0194	Bridge Static Debug Register
0x019A	Reserved	0x0198	Status Mux Control Register
0x019E	RX CPU DMA Control Register	0x019C	RXCPU DMA Control Register

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x01A2	RXCPU DMA Data Register	0x01A0	RXCPU DMA Data Register
0x01A6	Reserved	0x01A4	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x01BE	Reserved	0x01BC	Reserved
0x01C2	Global Interrupt Status Register	0x01C0	Global Interrupt Status Register
0x01C6	Global Interrupt Status Enable Register	0x01C4	Global Interrupt Status Enable Register
0x01CA	Global Interrupt Status Enable Register	0x01C8	Global Interrupt Status Enable Register
0x01D2	Reserved	0x01D0	Reserved
–	Reserved	–	Reserved
0x01FF	Reserved	0x01FC	Reserved

Slot2 Register Mapping

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x000	Slot2 CR IP registers	0x0000	Slot2 CR IP registers
–	Slot2 CR IP registers	–	Slot2 CR IP registers
0x00FF	Slot2 CR IP registers	0x00FC	Slot2CR IP registers
0x0102	Reserved	0x0100	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x017E	Reserved	0x017C	Reserved
0x0182	Bridge Mode Control Register	0x0180	Bridge Mode Control Register
0x0186	Bridge SDMA Status Register	0x0184	Bridge SDMA Status Register
0x018A	Bridge ADMA Status Register	0x0188	Bridge ADMA Status Register
0x018E	GRC/AHB Status Register	0x018C	GRC Control Register
0x0192	IP Control/Status Register	0x0190	IP Control/Status Register
0x0196	Bridge Static Debug Register	0x0194	Bridge Static Debug Register
0x019A	Reserved	0x0198	Status Mux Control Register
0x019E	RX CPU DMA Control Register	0x019C	RXCPU DMA Control Register

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x01A2	RXCPU DMA Data Register	0x01A0	RXCPU DMA Data Register
0x01A6	Reserved	0x01A4	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x01BE	Reserved	0x01BC	Reserved
0x01C2	Global Interrupt Status Register	0x01C0	Global Interrupt Status Register
0x01C6	Global Interrupt Status Enable Register	0x01C4	Global Interrupt Status Enable Register
0x01CA	Global Interrupt Status Enable Register	0x01C8	Global Interrupt Status Enable Register
0x01D2	Reserved	0x01D0	Reserved
–	Reserved	–	Reserved
0x01FF	Reserved	0x01FC	Reserved

Slot3 Register Mapping

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x000	Slot3 CR IP registers	0x0000	Slot3 CR IP registers
–	Slot3 CR IP registers	–	Slot3 CR IP registers
0x0102	Slot3 CR IP registers	0x0100	Slot3 CR IP registers
0x0106	Reserved	0x0104	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x017E	Reserved	0x017C	Reserved
0x0182	Bridge Mode Control Register	0x0180	Bridge Mode Control Register
0x0186	Bridge SDMA Status Register	0x0184	Bridge SDMA Status Register
0x018A	Bridge ADMA Status Register	0x0188	Bridge ADMA Status Register
0x018E	GRC/AHB Status Register	0x018C	GRC Control Register
0x0192	IP Control/Status Register	0x0190	IP Control/Status Register
0x0196	Bridge Static Debug Register	0x0194	Bridge Static Debug Register
0x019A	Reserved	0x0198	Status Mux Control Register
0x019E	RX CPU DMA Control Register	0x019C	RXCPU DMA Control Register

Addr Offset	*Bits[31:24] : Bits[23:16]*	*Addr Offset*	*Bits[15:8] : Bits[7:0]*
0x01A2	RXCPU DMA Data Register	0x01A0	RXCPU DMA Data Register
0x01A6	Reserved	0x01A4	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
–	Reserved	–	Reserved
0x01BE	Reserved	0x01BC	Reserved
0x01C2	Global Interrupt Status Register	0x01C0	Global Interrupt Status Register
0x01C6	Global Interrupt Status Enable Register	0x01C4	Global Interrupt Status Enable Register
0x01CA	Global Interrupt Status Enable Register	0x01C8	Global Interrupt Status Enable Register
0x01D2	Reserved	0x01D0	Reserved
–	Reserved	–	Reserved
0x01FF	Reserved	0x01FC	Reserved

SLOT1 Register Bit Definitions

Bridge Mode Control Register[0x183-0x180]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
SOB Disable Mode	31	RW	1	Start of Block for SDMA operation. 0: SDMA in SOB mode 1: SDMA in Basic/Normal mode of operation
SOD Disable Mode	30	RW	1	Start of a Descriptor fro ADMA1/ADMA operation. 0: ADMA in SOD mode 1: ADMA in Basic/Normal mode of operation
Reserved	29:24	RSVD	0	Reserved bits
DMA Write Combine mode	23:22	RW	01	Write combine DMA writes when SOB or SOD mode of operation is active 00: Reserved 01 : Combine AHB write bursts to max 128B packets 10: Combine AHB write burst to max 64B packets 11: Reserved Default: 01

Signal Name	***Bit Field***	***Attribute***	***Reset Value***	***Description***
DMA Read Pre fetch mode	20:21	RW	0	Read pre fetch DMA Reads when SOB or SOD mode of operation is active. The default max size Read request is 512B(SDMA) and 1KB(ADMA). One read request at a time issued from CR bridge to PCIe. 00: 1KB/512B-ADMA/SDMA 01: Split 1KB Read in to two 512B and 512B Rd in to two 256B Requests. 10: Split 1KB Read in to four 256B requests and 512B Read in to two 256B Requests 11: Reserved.
Reserved	19:16	RSVD	0	Reserved

DMA WR Byte Swap	15	RW	1	Selects DMA write data Byte swap format to WDMA
				0:LittleEndian{B3B2B1B0}
				1:BigEndian{B0B1B2B3}

DMA Read Response Byte Swap	14	RW	1	Selects DMA Read response data byte swap format to CR AHB Slave
				0:LittleEndian{B3B2B1B0}
				1:BigEndian{B0B1B2B3}

GRC Write Byte Swap	13	RW	0	Selects DMA Read response data byte swap format to CR AHB Master
				0:LittleEndian{B3B2B1B0}
				1:BigEndian{B0B1B2B3}

GRC Read Response Byte Swap	12	RW	0	Selects DMA Read response data byte swap format to GRC Bus.
				0:LittleEndian{B3B2B1B0}
				1:BigEndian{B0B1B2B3}

DMA WR Word Swap	11	RW	0	Selects DMA write data word swap format to WDMA
				0: Normal{B3B2B1B0}
				1: Word Swap{B1B0B3B2}

DMA Read Response Word Swap	10	RW	0	Selects DMA read response data word swap format to CR AHB Slave
				0: Normal{B3B2B1B0}
				1: Word Swap{B1B0B3B2}

Reserved	9:3	RSVD	0	Reserved
CR DMA F3 Disable	2	RW	0	Stop F3 DMA RD and WRITE request to PCIe 0: Enable the DMA Rd/Wr 1: Disable DMA Rd/Wr
CR DMA F2 Disable	1	RW	0	Stop F2 DMA RD and WRITE request to PCIe 0: Enable the DMA Rd/Wr 1: Disable DMA Rd/Wr

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
CR DMA F1 Disable	0	RW	0	Stop F1 DMA RD and WRITE request to PCIe 0: Enable the DMA Rd/Wr 1: Disable DMA Rd/Wr

Bridge SDMA Status Register[0x187-0x184]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
CR block Size	31:20	R	0	SDMA BLOCK SIZE latched with SOB signal
Reserved	19	RSVD	0	Reserved
CR Page size	18:16	R	0	SDMA PAGE SIZE latched with SOB signal
CR block count	15:0	R	0	SDMA BLOCK COUNT latched with SOB signal

Bridge ADMA Status Register[0x18B-0x188]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
CR Descriptor size	31:16	R	0	ADMA DESCRIPTOR SIZE latched with SOD signal
Reserved	15:0	RSVD	0	Reserved

GRC Control Register[0x18D-0x18C]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
P1GRC Turn off to AHB	0	RW	0	Terminate Host GRC requests in the CR Bridge. 0: Forward Host GRC requests to AHB Master 1: Stop HOST GRC requests to AHB master and locally terminate with dummy ACK's
P2GRC Turn off to AHB	1	RW	0	Terminate RX CPU GRC requests in the CR Bridge. 0: Forward RXCPU GRC requests to AHB Master 1: Stop RX CPU GRC requests to AHB master and locally terminate with dummy ACK's
Disable Global Interrupt	2	RW	0	Disables the Global interrupt offset(0x400 ~ 0x4-B). Mapped in the upper 512B range(0x1C0~0x1CB). 0: Forward the above range to AHB 1: Terminate above range with in CR bridge.

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
GRC Unsupported Request Enable	3	RW		Controls the GRC Read completion with Unsupported Request for the above 512B address space fro all three functions. 0: Unsupported request disabled 1: Unsupported request enabled
Reserved	15:4	RSVD	0	Reserved

GRC/AHB Status Register[0x18F-0x18E]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
P1GRC completion UR status	16	RW1C	0	Indicates the unsupported request (UR) status sent on the grc response ack. 0: No UR 1: UR status
P1 GRC Completion CA Status	17	RW1C	0	Indicates the completion abort (CA) status sent on the grc response ack. 0: No CA 1: CA status Unsupported Request status will be generated when the GRC Read request received with address range above 512B address (0x0200 ~0xFFFF) range and Bit 3 of GRC control register is set to 1. Completion Abort status will be generated When AHB Slave returns ERROR response for the Read completion on the AHB Bus.
GRC_ahb response err	18	RW1C	0	Indicates the error received in thRSVPRC AHB slave response. 0: No Error 1: ERROR received
GRC ahb response retry	19	RW1C	0	Indicates the retry received in the GRC AHB slave response. 0: No Retry 1: Retry received.
Reserved	31:20	RSVD	0	Reserved

IP Control/Status Register[0x193-0x190]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Slot1 Int status	0	R	0	Indicates SLOT 1 interrupt status: 0: INT Not Asserted 1: INT Asserted
Slot2 Int Status	1	R	0	Indicates SLOT 2 interrupt status 0: INT Not Asserted 1: INT Asserted
Slot3 Int Status	2	R	0	Indicates SLOT 3 interrupt status 0: INT Not Asserted 1: INT Asserted

Signal Name	***Bit Field***	***Attribute***	***Reset Value***	***Description***
Int_to_arm_status	3	R	0	Indicates Global interrupt status 0: INT Not Asserted 1: INT Asserted
Slot1 Active Status	4	R	0	Indicates Slot1 is active 0: Slot Not active 1: Slot active
Slot2 Active Status	5	R	0	Indicates Slot2 is active 0: Slot Not active 1: Slot active
Slot3 Active Status	6	R	0	Indicates Slot3 is active 0: Slot Not active 1: Slot active
Led1 Status	7	R	0	Indicates LED1 status
Led2 Status	8	R	0	Indicates LED2 status
Reserved	12:9	RSVD	0	Reserved
SD detect status	13	R	0	Indicates the Slot1 card detect status 0: No SD/MMC card 1: SD/MMC Card Inserted
MS detect status	14	R		Indicates the Slot1 card detect status 0: No MS/MSPRO card 1: MS/MSPRO Card Inserted
xD detect Status	15	R		Indicates the Slot2 card detect status 0: No xD card 1: xD Card Inserted
Reserved	28:16	RSVD	0	Reserved
Slot3 wakeup interrupt enable	29	R	0	Enables slot3 interrupt for the slot3 wakeup event 0: Disables the forwarding of slot3 wakeup event to slot3 interrupt. 1: Enables the forwarding of slot3 wakeup event to slot3 interrupt.
Slot2 wakeup interrupt enable	30	R	0	Enables slot2 interrupt for the slot2 wakeup event 0: Disables the forwarding of slot2 wakeup event to slot2 interrupt. 1: Enables the forwarding of slot2 wakeup event to slot2 interrupt.
Slot1 wakeup interrupt enable	31	R	0	Enables slot1 interrupt for the slt1 wakeup event 0: Disables the forwarding of slot1 wakeup event to slot1 interrupt. 1: Enables the forwarding of slot1 wakeup event to slot1 interrupt.

Bridge Static Debug Register[0x197-0x194]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
CR debug reg	31:0	R	0	Internal Static Debug Status Register

CR Status Mux Control Register[0x199-0x198]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Static Mux Select	5:0	RW	0	Controls the internal static bit register mux
Reserved	7:6	RSVD	0	Reserved
Dynamic Mux Select	11:8	RW	0	Controls the dynamic register mux
Reserved	15:12	RSVD	0	Reserved

Clock Control/Status Register [0x19F-0x19C]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Reserved for Internal Debug SD3.0	31:0	RW	0	Internal debug for SD3.0

Capability Register Slot1-SD/MMC [0x1A3-0x1A0]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
DDR50 Support	31	RW	1	1: DDR50 is supported 0: DDR50 is not supported
SDR104 Support	30	RW	1	1: SDR104 is supported 0: SDR104 is not supported
SDR50 Support	29	RW	1	If SDR104 is supported, this bit shall be set to 1. Bit 40 indicates whether SDR50 requires tuning or not. 1: SDR50 is supported 0: SDR50 is not supported
Slot Type	28:27	RW	0	This field indicates usage of a slot by a specific host system. (A host controller register set is defined per slot.) Embedded slot for one device (01b) means that only one nonremovable device is connected to a SD bus slot. Shared bus slot (10b) can be set if host controller supports Shared BusControl register. The standard host driver controls only a removable card or one embedded device is connected to a SD bus slot. If a slot is configured for shared bus (10b), the standard host driver does not control embedded devices connected to a shared bus. Shared bus slot is controlled by a specific host driver developed by a host system. 00: Removable card slot 01: Embedded slot for one device 10: Shared bus slot 11: Reserved
Asynchronous Interrupt Support	26	RW	0	Refer to SDIO Specification version 3.00 about asynchronous interrupt. 1: Asynchronous interrupt supported 0: Asynchronous interrupt not supported
64-bit System Bus Support	25	RW	0	1: Supports 64 bit system address 0: Does not support 64 bit system address
Voltage Support 1.8V	24	RW	1	0: 1.8V not supported 1: 1.8V supported
Voltage Support 3.0V	23	RW	1	0: 3.0V not supported 1: 3.0V supported
Voltage Support 3.3V	22	RW	1	0: 3.3V not supported 1: 3.3V supported

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Suspend/Resume Support	21	RW	1	This bit indicates whether the HC supports suspend/resume functionality. If this bit is 0, the suspend and resume mechanism is not supported, the HD does not issue either suspend/resume commands. 0: Not supported 1: Supported
SDMA Support	20	RW	1	This bit indicates whether the HC is capable of using DMA to transfer data between system memory and the HC directly. 0: SDMA not supported 1: SDMA supported
High Speed Support	19	RW	1	This bit indicates whether the HC and the host system support High Speed mode and they can supply SD clock frequency from 25 MHz to 50 MHz (for SD)/ 20 MHz to 52 MHz (for MMC). 0: High speed not supported 1: High speed supported
ADMA2 Support	18	RW	1	1: ADMA2 supported 0: ADMA2 not supported
Extended media Support	17	RW	1	This bit indicates whether the host controller is capable of using 8-bit bus width mode. This bit is not effective when slot type is set to 10b. In this case, refer to bus width preset in the Shared Bus resister. 1: Extended media bus supported 0: Extended media bus not supported
Max Block Length	16:15	RW	2'b10	This value indicates the maximum block size that the HD can read and write to the buffer in the HC. The buffer shall transfer this block size without wait cycles. Three sizes can be defined as indicated below: 00: 512 byte 01: 1024 byte 10: 2048 byte 11: 4096 byte

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Base Clock Frequency for SD Clock	14:7	RW	8'h32	(1) 6-bit base clock freq. This mode is supported by the host controller version 1.00 and 2.00. Upper 2-bit is not effective and always 0. Unit values are 1 MHz. The supported clock range is 10 MHz to 63 MHz: 11xx xxxx: Not supported 0011 1111: 63 MHz 0000 0010: 2 MHz 0000 0001: 1 MHz 0000 0000: Get information via another method (2) 8-bit base clock freq. This mode is supported by the host controller version 3.00. Unit values are 1 MHz. The supported clock range is 10 MHz to 255 MHz: 8'hFF: 255 MHz ……… ……… ………. 8'h02: 2 MHz 8'h01: 1 MHz 8'h00: Get information via another method If the real frequency is 16.5 MHz, the lager value is set 00010001b (17 MHz) because the host driver uses this value to calculate the clock divider value (refer to the SDCLK Frequency Select in the Clock Control register), and it does not exceed upper limit of the SD clock frequency. If these bits are all 0, the host system has to get information via another method.
Timeout Clock Unit	6	RW	1	This bit shows the unit of base clock frequency used to detect data timeout error. 0: kHz 1: MHz
Timeout Clock Frequency	5:0	RW	6'h30	This bit shows the base clock frequency used to detect data timeout error. Not 0: 1 kHz to 63 kHz or 1 MHz to 63 MHz 000000: Get Information via another method

Capability Register Slot1-SD/MMC [0x1A7-0x1A4]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
SD base clockfreq override bit	31	RW	0	0: Selects the SD clock base frequency value based on the CPMU clock control two bits 1: Selects the SD clock base frequency value programmed in the offset 0x1A0[14:7]
Reserved	30:20	R	0	Reserved
SPI block mode support	19	RW	1	Spi block mode 0: Not Supported 1: Supported
SPI mode support	18	RW	1	Spi mode 0: Not supported 1: Supported
Clock Multiplier	17:10	RW	0	This field indicates clock multiplier value of programmable clock generator. Refer to the Clock Control register. Setting 00h means that host controller does not support programmable clock generator. 8'hFF: Clock multiplier M = 256 .... 8'h02: Clock multiplier M = 3 8'h01: Clock multiplier M = 2 8'h00: Clock multiplier is not supported
Re-Tuning modes	9:8	RW	0	This field defines the retuning capability of a host controller and how to manage the data transfer length and a retuning timer by the host driver: 00: Mode1 01: Mode2 10: Mode3 11: Mode4 There are two retuning timings: Retuning Request and expiration of a retuning timer. By receiving either timing, the host driver executes the retuning procedure just before a next command issue.
Use Tuning for SDR50	7	RW	1	If this bit is set to 1, this host controller requires tuning to operate SDR50. (Tuning is always required to operate SDR104.) 1: SDR50 requires tuning 0: SDR50 does not require tuning

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Timer count for Retuning	6:3	RW	4'h1	This field indicates an initial value of the retuning timer for retuning mode 1 to 3. 4'h0: Get information via other source 4'h1: 1 seconds 4'h2: 2 seconds 4'h3: 4 seconds 4'h4: 8 seconds ……. …….. ……… n = 2(n-1) seconds ……. ……. 4'hB: 1024 seconds 4'hC–4'hF:Reserved
Driver Type D Support	2	RW	1	This bit indicates support of driver type D for 1.8 signaling. 1: Driver type D is supported. 0: Driver type D is not supported.
Driver Type C Support	1	RW	1	This bit indicates support of driver type C for 1.8 signaling. 1: Driver type C is supported. 0: Driver type C is not supported.
Driver Type A Support	0	RW	1	This bit indicates support of driver type A for 1.8 Signaling. 1: Driver type A is supported. 0: Driver type A is not supported.

Capability Register Slot2-MS/MSPro [0x1AB-0x1A8]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
DDR50 Support	31	RW	1	1: DDR50 is supported 0: DDR50 is not Supported
SDR104 Support	30	RW	1	1: SDR104 is supported 0: SDR104 is not supported
SDR50 Support	29	RW	1	If SDR104 is supported, this bit is set to 1. Bit 40 indicates whether SDR50 requires tuning or not. 1: SDR50 is supported 0: SDR50 is not supported
Slot Type	28:27	RW	0	This field indicates usage of a slot by a specific host system. (A host controller register set is defined per slot.) Embedded slot for one device (01b) means that only one nonremovable device is connected to a SD bus slot. Shared bus slot (10b) can be set if host controller supports Shared BusControl register. The standard host Driver controls only a removable card or one embedded device is connected to a SD bus slot. If a slot is configured for shared bus (10b), the standard host driver does not control embedded devices connected to a shared bus. Shared bus slot is controlled by a specific host driver developed by a host system. 00: Removable card slot 01: Embedded slot for one device 10: Shared bus slot 11: Reserved
Asynchronous Interrupt Support	26	RW	0	Refer to SDIO Specification version 3.00 about asynchronous interrupt. 1: Asynchronous interrupt supported 0: Asynchronous interrupt not supported
64-bit System Bus Support	25	RW	0	1: Supports 64 bit system address 0: Does not support 64 bit system address
Voltage Support 1.8V	24	RW	1	0: 1.8V not supported 1: 1.8V supported
Voltage Support 3.0V	23	RW	1	0: 3.0V not supported 1: 3.0V supported
Voltage Support 3.3V	22	RW	1	0: 3.3V not supported 1: 3.3V supported

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Suspend/Resume Support	21	RW	1	This bit indicates whether the HC supports suspend/resume functionality. If this bit is 0, the suspend and resume mechanism is not supported, and the HD does not issue either suspend/resume commands. 0: Not supported 1: Supported
SDMA Support	20	RW	1	This bit indicates whether the HC is capable of using DMA to transfer data between system memory and the HC directly. 0: SDMA not supported 1: SDMA supported.
High Speed Support	19	RW	1	This bit indicates whether the HC and the host system support High Speed mode and they can supply SD clock frequency from 25 MHz to 50 MHz (for SD)/ 20 MHz to 52 MHz (for MMC). 0: High speed not supported 1: High speed supported
ADMA2 Support	18	RW	1	1: ADMA2 support 0: ADMA2 not support
Extended Media Support	17	RW	1	This bit indicates whether the host controller is capable of using 8-bit bus width mode. This bit is not effective when slot type is set to 10b. In this case, refer to bus width preset in the Shared Bus resister. 1: Extended media bus supported 0: Extended media bus not supported
Max Block Length	16:15	RW	2'b10	This value indicates the maximum block size that the HD can read and write to the buffer in the HC. The buffer shall transfer this block size without wait cycles. Three sizes can be defined as indicated below. 00: 512 byte 01: 1024 byte 10: 2048 byte 11: 4096 byte

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Base Clock Frequency for SD Clock	14:7	RW	8'h32	(1) 6-bit base clock freq. This mode is supported by the host controller version 1.00 and 2.00. Upper 2-bit is not effective and always 0. Unit values are 1 MHz. The supported clock range is 10 MHz to 63 MHz: 11xx xxxx: Not supported 0011 1111: 63 MHz 0000 0010: 2 MHz 0000 0001: 1 MHz 0000 0000: Get information via another method (2) 8-bit base clock freq. This mode is supported by the host controller version 3.00. Unit values are 1 MHz. The supported clock range is 10 MHz to 255 MHz: 8'hFF: 255 MHz ……… ……… ………. 8'h02: 2 MHz 8'h01: 1 MHz 8'h00: Get information via another method If the real frequency is 16.5 MHz, the lager value is set 00010001b (17 MHz) because the host driver uses this value to calculate the clock divider value (refer to the SDCLK Frequency Select in the Clock Control register), and it does not exceed upper limit of the SD clock frequency. If these bits are all 0, the host system has to get information via another method.
Timeout Clock Unit	6	RW	1	This bit shows the unit of base clock frequency used to detect data timeout error. 0: kHz 1: MHz
Timeout Clock Frequency	5:0	RW	6'h30	This bit shows the base clock frequency used to detect data timeout error. Not 0: 1 kHz to 63 kHz or 1 MHz to 63 MHz 000000: Get Information via another method

Capability Register Slot2-MS/MS Pro [0x1AF-0x1AC]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Host spec version control Register Override bit for MS and MSPro	31	RW	0	0: Selects the host control version based on the CPMU clock control bits 1: Selects the Host version control value from offset 0x1AC[30:23]
Host spec version control value	30:23	RW	0	8-bit spec version control value
Reserved	22:20	R	0	Reserved
SPI block mode support	19	RW	1	Spi block mode: 0: Not supported 1: Supported
SPI mode support	18	RW	1	Spi mode: 0: Not supported 1: Supported
Clock Multiplier	17:10	RW	0	This field indicates clock multiplier value of programmable clock generator. Refer to the Clock Control register. Setting 00h means that host controller does not support programmable clock generator. 8'hFF: Clock multiplier M = 256 .... 8'h02: Clock multiplier M = 3 8'h01: Clock multiplier M = 2 8'h00: Clock multiplier is not supported
Re-Tuning modes	9:8	RW	0	This field defines the retuning capability of a host controller and how to manage the data transfer length and a retuning timer by the host driver: 00: Mode1 01: Mode2 10: Mode3 11: Mode4 There are two retuning timings: Retuning request and expiration of a retuning timer. By receiving either timing, the host driver executes the retuning procedure just before a next command issue.

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Use Tuning for SDR50	7	RW	1	If this bit is set to 1, this host controller requires tuning to operate SDR50. (Tuning is always required to operate SDR104.) 1: SDR50 requires tuning 0: SDR50 does not require tuning
Timer count for retuning	6:3	RW	4'h1	This field indicates an initial value of the retuning timer for retuning mode 1 to 3. 4'h0: Get information via other source 4'h1: 1 seconds 4'h2: 2 seconds 4'h3: 4 seconds 4'h4: 8 seconds ……. …….. ……… n = 2(n-1) seconds ……. ……. 4'hB: 1024 seconds 4'hC - 4'hF: Reserved
Driver Type D Support	2	RW	1	This bit indicates support of driver type D for 1.8 signaling. 1: Driver type D is supported 0: Driver type D is not supported
Driver Type C Support	1	RW	1	This bit indicates support of driver type C for 1.8 signaling. 1: Driver type C is supported 0: Driver type C is not supported
Driver Type A Support	0	RW	1	This bit indicates support of driver type A for 1.8 signaling. 1: Driver type A is supported 0: Driver type A is not supported

Preset Registers_set0 [0x1BF-0x1B0]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Reserved for Internal Debug SD3.0	31:0	RW	0	Preset values for SD3.0

4.15: Preset Registers_set1 [0x1DF-0x1D0]

Signal Name	*Bit Field*	*Attribute*	*Reset Value*	*Description*
Reserved for Internal Debug SD3.0	31:0	RW	0	Preset values for SD3.0

Appendix A: Flow Control

Notes

Developers can refer to the IEEE 802.3 Annex 31B specification for detailed information on Ethernet flow control mechanisms.

- Flow control frames use a well-known multicast address, defined in the 802.1D Bridging specification The MAC destination address is 01-80-C2-00-00-01
- Bridges and Switches will not forward pause frames to downstream ports.
- A pause frame contains a request_operand that contains a pause_time field. Pause_time specifies the number of quanta, which transmission should be inhibited.
- Pause frames cannot inhibit MAC control Frames
- Pause_time is a two-octet field, which represents a quanta value. The quanta value is based on bit/slot times for the connection speed. Valid pause_times vary from 0 to 65535.
- The pause frame contains a MAC control opcode. 00-01 is reserved for PAUSE MAC control functions.
- MAC control layers will provide two indicators—paused and not paused.
- The Enet source address equals the unicast address of the MAC sublayer, which transmits the pause_frame.
- The receive engine will set a countdown timer, based on the value of pause_time. When the timer expires, the transmit engine may resume send operation
- A Mac sublayer may transmit pause frames with pause_time = 0. The zero value will stop a pause count down, executed by the MAC's link partner. Effectively, a value of zero restarts a link partner's transmit engine, assuming the link partner was inhibited by a previous pause operation.

Flow Control Scenario

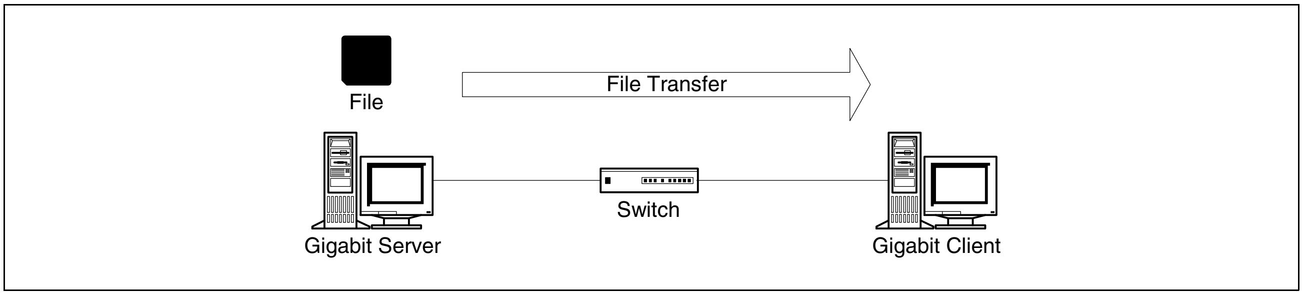
This scenario assumes that the Gigabit switch has a 1:1 port mapping, between the Gigabit Server and Client. The switch does not implement header aligned blocking, nor will it drop packets to alleviate buffer pressure. The following constraints are placed on this scenario:

- Client
  - Full-duplex connection at Gigabit speed.
  - Implements flow control.
  - Flow control enabled.
- Switch
  - Does not drop packets.
  - Full-duplex connection to Client.
- Server
  - Gigabit connection.
  - Either half/full-duplex connection at Gigabit speed (i.e., this scenario will cover two subcases).

File Transfer

The client begins a FTP session (see Figure 64). The file size is very large and will take several minutes for a complete transfer, even at gigabit wire speed.

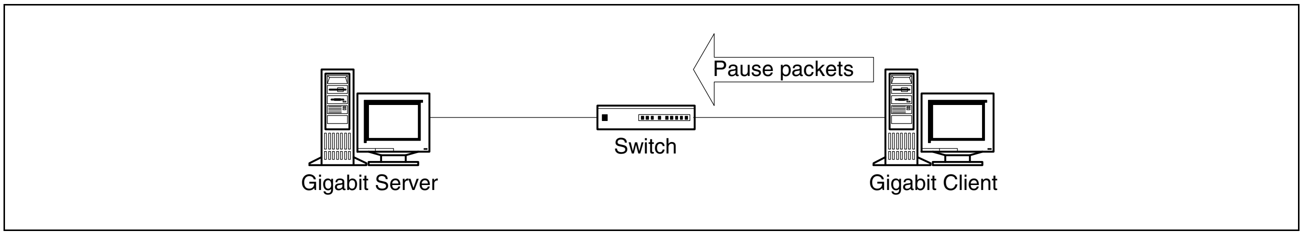
Figure 64: File Transfer Scenario: FTP Session Begins



Speed Mismatch

The Client sends pause frame(s) to the switch (see Figure 65). The Client's pipe has been saturated, and the RX buffers are almost exhausted. The Client begins sending pause frames, when the RX buffer high-water mark/threshold is hit. Any number of reasons can account for the RX buffer issue. The assumption will be made that the Client PCI bus lack bandwidth to DMA packets, at wire speed, to host memory. The user may be playing a DVD, for example.

Figure 65: File Transfer Scenario: Speed Mismatch

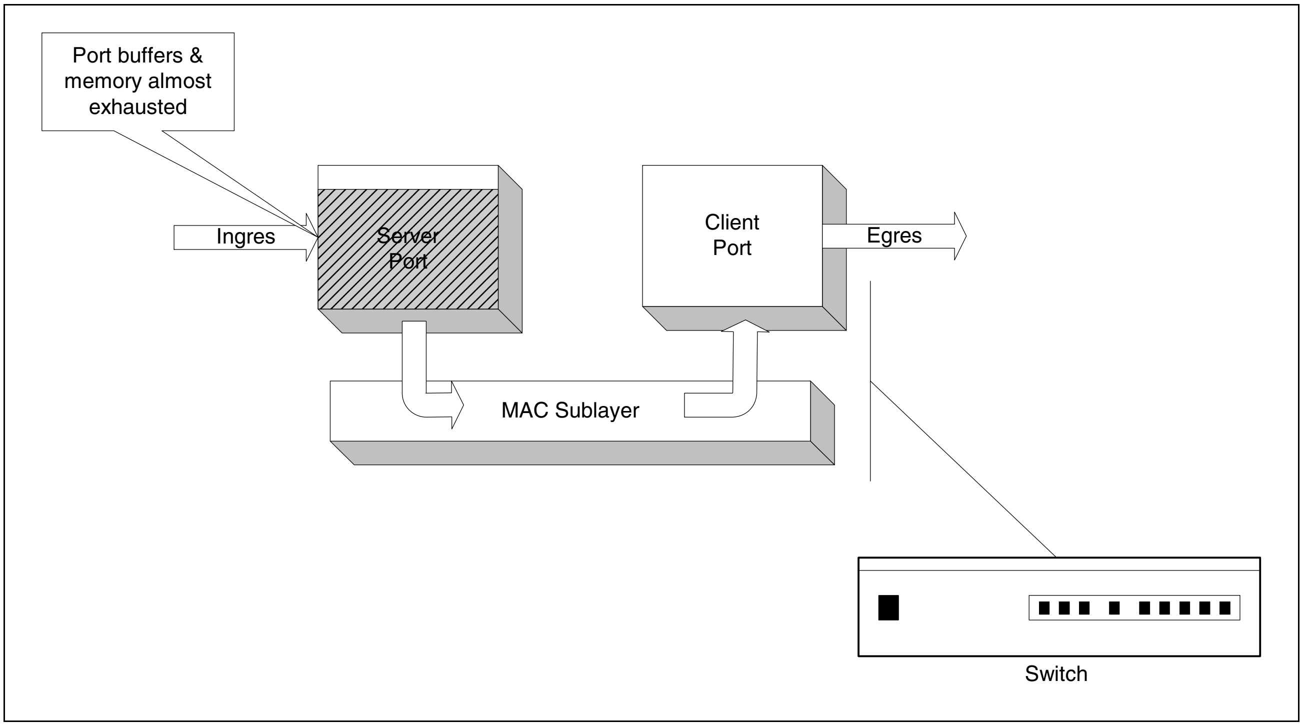


Switch Buffers Run Low

The switch must wait/inhibit transmission to the Client (see Figure 66). During the pause interval, the Server is still sending packets to the Switch. The Switch will buffer some packets, but will eventually hit an internal threshold; memory will run short. Since dropping packets is undesirable, the Switch must slow incoming packets from the Server. The duplex mode of the Server's connection will dictate how the switch slows traffic. There are two options:

- Jamming (half-duplex)
- Pause frames (full-duplex)

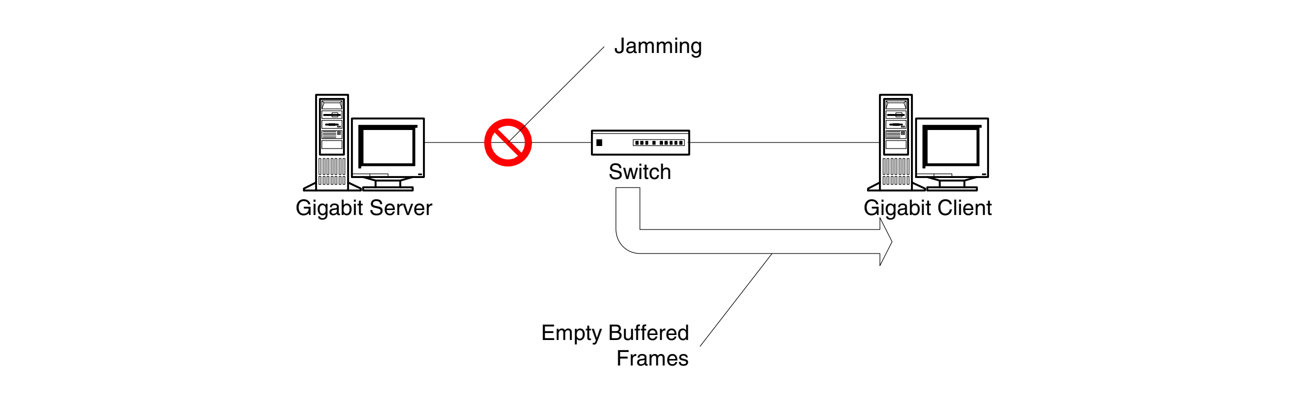
Figure 66: File Transfer Scenario: Speed Buffers Run Low



Switch Backpressure

The Switch will jam ports configured with half-duplex link to slow frame transmission (see Figure 67). In this case, the Server connection must be half-duplex, and then the switch may apply backpressure to the port. The Switch will transmit a jamming pattern, which will prevent the Server from transmitting further packets. The Server's MAC will detect a collision situation, and will back off for a specified interval. The Switch will continue to apply backpressure to the Server Ingress, until the Client egress is available. The Client port will be available when the pause interval expires, and no further pause packets are sent by the Gigabit Client.

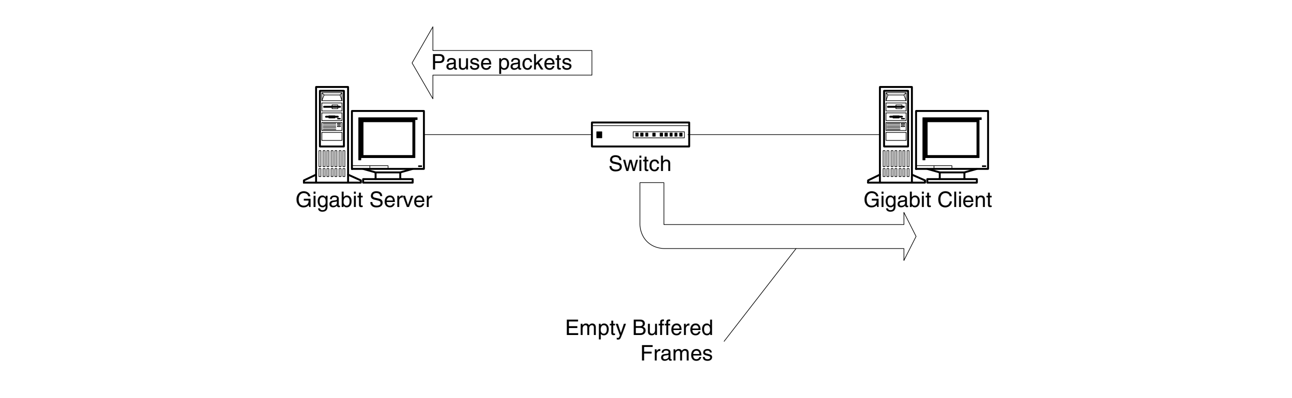
Figure 67: File Transfer Scenario: Switch Backpressure



Switch Flow Control

The Switch can only use IEEE 802.3x flow control when the link is configured for full-duplex operation (see Figure 68). When buffers are near exhaustion, the switch will send a pause frame to the Server. The Server's MAC will be inhibited for a pause_time interval. During the pause interval, the Switch has the opportunity to empty the buffered packets. Once the buffered packets fall below the high water mark, the Switch may send another pause frame, with pause_time = 0, to terminate the Server's pause interval. The Switch may also allow the Server's pause interval to expire. Either way, the Switch no longer will inhibit the Server from sending packets.

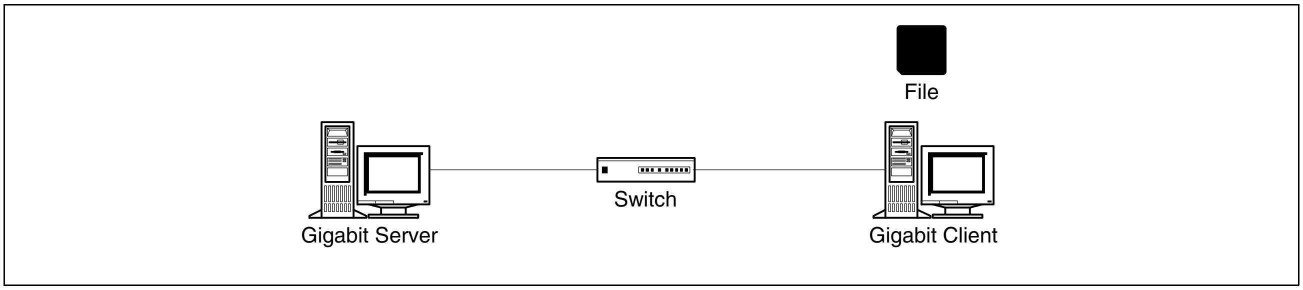
Figure 68: File Transfer Scenario: Switch Flow Control



File Transfer Complete

The Client has caught up with the transmission flow of the Server (see Figure 69). The Client's RX buffers/ memory is below the flow control threshold. The file is transfer is complete. This scenario was a worst-case cascade, where the pause delay propagated through the LAN. The Switch could absorb the Client's pause delay, without having to flow control the Server.

Figure 69: File Transfer Scenario: File Transfer Complete

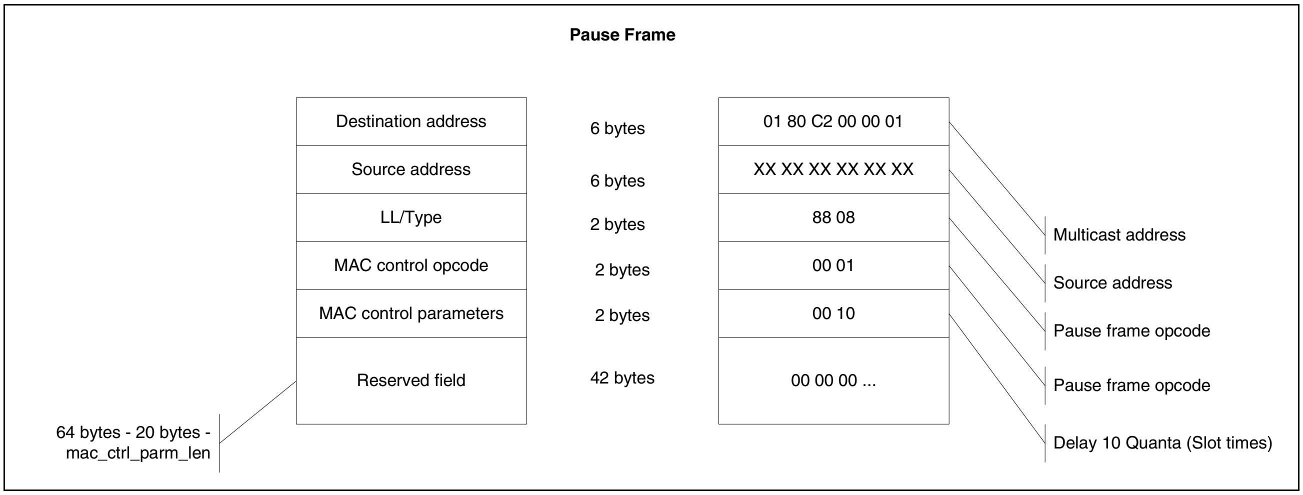


Pause Control Frame

The minimum size frame is 512 bits or 64 bytes (see Figure 70). MAC control frames must pad zeros into the unused portion of the payload. A flow control frame contains the following fields:

- Destination address field, set to 01-80-C2-00-00-01
- Source address field set to unique MAC address of sender
- LL/Type field set to the 802_3_MAC_CONTROL value, set to 88–08
- MAC control pause opcode (00-01), pause_time, and reserved field (zeros)

Figure 70: Pause Control Frame


Pause Frame

Destination address	6 bytes	01 80 C2 00 00 01
Source address	6 bytes	XX XX XX XX XX XX
LL/Type	2 bytes	88 08
MAC control opcode	2 bytes	00 01
MAC control parameters	2 bytes	00 10
Reserved field	42 bytes	00 00 00 ...

64 bytes - 20 bytes - mac_ctrl_parm_len
Multicast address
Source address
Pause frame opcode
Pause frame opcode
Delay 10 Quanta (Slot times)

Appendix B: Acronyms and Abbreviations

Table 110 defines acronyms and abbreviations used throughout this document.

Table 110: Terms and Acronyms

Term/Acronym	***Definition***
BD	Buffer Descriptor.
Deferred Procedure Call (DPC)	The ISR may schedule a O/S callback to process interrupts at a later time.
Expansion ROM	PCI devices may optionally expose device specific programs to BIOS. For example, network devices may place PXE boot code in their expansion ROM region.
Host Coalescing	A hardware block which the Ethernet controller status block. The hardware will drive a line interrupt or MSI.
Interrupt Distribution Queue	The Ethernet controller supports four interrupt distribution queues per class of service. The rules engine may place traffic into RX return rings based on rules checking. Within each class of service, the traffic may further be organized in Interrupt Distribution Queues. For example, frames with errors may be given lower data path priority over frames without errors, all within the same class of service (RX Return Ring).
Interrupt Service Routine (ISR)	A procedure where device interrupts are processed.
Pre-boot execution (PXE)	An industry-standard client/server interface that allows networked computers that are not yet loaded with an operating system to be configured and booted remotely.
Receive BD Initiator	The hardware block that DMA's BDs when receive ring indices are written.
Receive Data and Receive BD Initiator	The hardware block the updates packet buffers, in host memory, after an Ethernet frame is received. The hardware block will also update the BD with information like checksum and VLAN Tags.
Receive Data Completion	The hardware block that updates the host coalescing engine after the packet buffers and BD are DMAed to host memory.
Receive Queue Placement	The hardware block that routes a categorized frame to one of sixteen RX Return rings.
Send BD Initiator	The hardware block that is activated when a Send producer index is updated by host software. The hardware block will DMA a BD from host memory.
Send Data Initiator	The hardware block updates the DMAs in the packet buffers from host memory. The packet buffers are DMAed after the BD has been moved to device local memory.