NetXtreme®/NetLink™ BCM5718 Family Programmer's Guide

5718-PG107-R

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

July 17, 2013

Revision History

Revision	*Date*	*Change Description*
5718-PG-107-R	07/17/13	**Updated:** • "Send Rings" on page 106 • "Initialization Procedure" on page 140 • Table 49: "GPIO Usage for Power Management for Broadcom Drivers," on page 192 • Table 101: "Multiple Send Ring Mail Boxes," on page 357 • "Send BD Ring Host Producer Index Register (offset: 0x5900)" on page 465 • "Send BD Ring NIC Producer Index Register (offset: 0x5980)" on page 466 • Table 121: "GbE Port Internal PHY Register Map," on page 553 • Table 127: "AUTONEG LINK PARTNER ABILITY," on page 559 **Added:** • Table 124: "02h: PHY_Identifier_MSB_Register," on page 558 • Table 125: "03h: PHY_Identifier_LSB_Register," on page 558
5718-PG-106-R	06/25/12	**Updated:** • "Base Address Register 1 (offset: 0x10)" on page 275 • "Base Address Register 2 (offset: 0x14)" on page 275 • "Base Address Register 3 (offset: 0x18)" on page 275 • "Base Address Register 4 (offset: 0x1c)" on page 276 • "Mode Control Register (offset: 0x6800)" on page 475 **Added:** • Section 8: "IEEE1588," on page 152 • "RX TIME STAMP LSB REG [Offset 0X06B0]" on page 163 • "RX TIME STAMP MSB REG [Offset 0x06B4]" on page 163 • "RX PTP SEQUENCE ID REG [Offset 0X06B8]" on page 163 • "RX LOCK TIMER LSB REG [Offset 0x06C0]" on page 164 • "RX LOCK TIMER MSB REG [Offset 0x06C4]" on page 164 • "RX PTP CONTROL REG [Offset 0X06C8]" on page 164 • Section 12: "IO Virtualization (IOV)," on page 264 • "Perfect Match Destination Address Registers" on page 465 • "VRQ Filter Set Registers" on page 461 • "VRQ Mapper Registers" on page 462 • "Base Address Register 5 (offset: 0x20)" on page 276 • "Base Address Register 6 (offset: 0x24)" on page 277

Revision	*Date*	*Change Description*
5718-PG-105-R	02/24/12	**Updated:** • Table 3: "Family Revision Levels," on page 48 • Table 5: "Flag Fields for a Ring," on page 70 • Figure 24: "Ring Control Block," on page 118 • "Summary of Register Settings to Support Jumbo Frames" on page 126 • "Initialization Procedure" on page 136 • "Reading a PHY Register" on page 186 • "Writing a PHY Register" on page 187 • "Subsystem ID/Vendor ID Register (offset: 0x2C)" on page 251 • "DMA Read/Write Control Register (Offset: 0x6c)" on page 258 • "PCI State Register (offset: 0x70)" on page 259 • "Receive BD Standard Producer Ring Index Register (offset: 0x268-0x26f)" on page 281 • "Transmit MAC Status Register (offset: 0x460)" on page 296 • "Receive MAC Mode Register (offset: 0x468)" on page 297 • "Statistics Registers" on page 319 • "H2B Statistics Registers" on page 320 • "Receive Data and Receive BD Initiator Mode Register (offset: 0x2400)" on page 341 • "Link Speed 10 MB/No Link Power Mode Clock Policy Register (offset: 0x3604)" on page 356 • "Link Speed 100 MB Power Mode Clock Policy Register (offset: 0x3608)" on page 357 • "Link Aware Power Mode Clock Policy Register (offset: 0x3610)" on page 359 • "D0u Clock Policy Register (offset: 0x3614)" on page 360 • "Link Idle Power Mode Clock Policy Register (offset: 0x3618)" on page 360 • "APE CLK Policy Register (offset: 0x361C)" on page 361 • "APE Sleep State Clock Policy Register (offset: 0x3620)" on page 363 • "Clock Speed Override Policy Register (offset: 0x3624)" on page 364 • "Clock Status Register (offset: 0x3630)" on page 367 • "Padring Control Register (offset: 0x3668)" on page 376 • "Receive Coalescing Ticks Register (offset: 0x3C08)" on page 393 • "Send Coalescing Ticks Register (offset: 0x3C0C)" on page 394 • "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 395 • "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 397 • "Status Block Host Address Register (offset: 0x3C38)" on page 400

Revision	Date	Change Description
		• "Status Block Base Address Register (offset: 0x3C44)" on page 426 • "BM Hardware Diagnostic 2 Register (offset: 0x4450)" on page 436 • "LSO Read DMA Mode Register (offset: 0x4800)" on page 438 • "LSO Read DMA Reserved Control Register (offset: 0x4900)" on page 445 • "LSO Read DMA Flow Reserved Control Register (offset: 0x4904)" on page 446 • "LSO/Non-LSO/BD Read DMA Corruption Enable Control Register (offset: 0x4910)" on page 446 • "BD Read DMA Mode Register (Offset: 0x4A00)" on page 449 • "BD READ DMA Reserved Control Register (offset: 0x4A70)" on page 456 • "BD READ DMA Flow Reserved Control Register (offset: 0x4A74)" on page 457 • "BD READ DMA Corruption Enable Control Register (offset: 0x4A78)" on page 457 • "Non_LSO Read DMA Mode Register (offset: 0x4B00)" on page 458 • "Non-LSO Read DMA Reserved Control Register (offset: 0x4B74)" on page 461 • "Non-LSO Read DMA Corruption Enable Control Register (offset: 0X4B7C)" on page 462 • "Write DMA Mode Register (offset: 0x4C00)" on page 464 • "Low Priority Mailboxes" on page 469 • "Interrupt Mailbox 0 Register (offset: 0x5800)" on page 469 • "Other Interrupt Mailbox Register (offset: 0x5808–0x5818)" on page 469 • "General Mailbox Registers 1-8 (offset: 0x5820–0x5824)" on page 469 • "Receive BD Standard Producer Ring Index Register (offset: 0x5868)" on page 470 • "Receive BD Return Ring 0 Consumer Index Register (offset: 0x5880-0x5887)" on page 470 • "Receive BD Return Ring 0 Consumer Index Register (offset: 0x5880-0x5887)" on page 470 • "Send BD Ring Consumer Index Register (offset: 0x5900)" on page 471 • "NVM Write Register (offset: 0x7008)" on page 497 • "NVM Address Register (offset: 0x700C)" on page 497 • "NVM Read Register (offset: 0x7010)" on page 498 • "NVM Config 1 Register (offset: 0x7014)" on page 498 • "NVM Access Register (offset: 0x7024)" on page 502 • "00h: MII_Control_Register" on page 513 • "03h: PHY_Identifier_LSB_Register" on page 515 • "04h: Auto_Negot_Advertisement_Register" on page 515 • "09h: 1000Base_T_Control_Register" on page 518 • "10h: PHY_Extended_Control_Register" on page 522 • "18h: Auxiliary Control Register (Shadow Register Selector = "000")" on page 526

Revision	Date	Change Description
		• "18h: Miscellaneous Control Register (Shadow Register Selector = "111")" on page 533 • "1Ch: Cabletron LED Register (Shadow Register Selector = "00h")" on page 538 • "1Ch: Spare Control 4 Register (Shadow Register Selector = "0bh")" on page 547 • "1Ch: External Serdes Control Register (Shadow Register Selector = "14h")" on page 557 • "1Ch: SGMII Slave Register (Shadow Register Selector = "15h")" on page 559 • "1Ch: Misc 1000-X Control 2 Register (Shadow Register Selector = "16h")" on page 561 • "1Ch: Misc 1000-X Control Register (Shadow Register Selector = "17h")" on page 563 • "1Ch: Auto-Detect SGMII/GBIC Register (Shadow Register Selector = "18h")" on page 564 • "1Ch: Auto-Detect Medium Register (Shadow Register Selector = "1eh")" on page 572 • "1Ch: Mode Control Register (Shadow Register Selector = "1fh")" on page 573 **Added:** • Table 1: "Register Access Methods," on page 46 • "Device Reset Procedure" on page 146 • "PHY Loopback Configuration" on page 205 • "PHY Configuration Auto-Negotiation (10/100/1000 Speed with Half and Full Duplex Support)" on page 206 • "MSI-X Capabilities Registers" on page 281 • "PCIe Capabilities Registers" on page 282 • "VRQ Flush Control Register (Offset: 0x2410)" on page 369 • "VRQ Flush Timer Register (offset: 0x2414)" on page 370 • "RDI B2HRX Hardware Debugging Register (offset: 0x2418)" on page 370 • "Receive BD Ring Initiator Local NIC Standard Receive BD Consumer Index (offset: 0x2474)" on page 373 • "B2HRX Byte-count Statistics Count (offset: 0x24D0)" on page 374 • "B2HRX Unicast Statistics Count (offset: 0x24D4)" on page 374 • "B2HRX Multicast Statistics Count (offset: 0x24D8)" on page 374 • "B2HRX Broadcast Statistics Count (offset: 0x24DC))" on page 374 • "B2HRX Drop Packet Count (offset: 0x24E0)" on page 374 • "B2HRX Drop Packet Byte Count (offset: 0x24E4)" on page 374 • "B2HRX APE Byte-count Statistics Count (offset: 0x24E8)" on page 375 • "B2HRX APE Unicast Statistics Count (offset: 0x24EC)" on page 375 • "B2HRX APE Multicast Statistics Count (offset: 0x24F0)" on page 375 • "B2HRX APE Broadcast Statistics Count (offset: 0x24F4)" on page 375 • "B2HRX APE Drop Packet Count (offset: 0x24F8)" on page 375 • "B2HRX APE Drop Packet Byte Count (offset: 0x24FC)" on page 375

Revision	Date	Change Description
		• "Receive Max Coalesced BD Count During Interrupt Register (offset: 0x3C18)" on page 424 • "Send Max Coalesced BD Count During Interrupt Register (offset: 0x3C1C)" on page 425 • "NIC Mini Receive BD Consumer Index (offset: 0x3c58)" on page 428 • "Send BD Ring Producer Index Register (offset: 0x5980)" on page 471 • "DMA Completion Mode Register (Offset: 0x6400)" on page 477 • Figure 58: "Copper PHY Register Mapping Table," on page 511 • Figure 59: "Serdes PHY Register Map," on page 512 • "Clause 45 Registers" on page 601 • "SerDes PHY Register Definitions" on page 578 • "PHY 0x18 Shadow 0x1 register read Procedure" on page 527 • Added PHY 0x1C Shadow 0x1 register read Procedure information to "1Ch: Cabletron LED Register (Shadow Register Selector = "00h")" on page 538 • Added Clause 45 register Dev3 Reg803Eh read Procedure to "Clause 45 Register Dev 3 Reg14h (20d): EEE Capability Register" on page 601 • NIC Ring Addresses information to Memory map tables in Appendix C: "Device Register and Memory Map," on page 611 **Deleted** • Section 11: Host to/from BMC Pass Through • Appendix D: Appendix • Top Level MII Registers
5718-PG104-R	06/29/11	**Updated:** • Table 27: "Flag Field Description," on page 113 • Table 31: "Send Buffer Descriptor Flags," on page 123 • "Clock Control" on page 191 • Table 47: "Ethernet Controller Power Pins," on page 191 • "Internal Memory" on page 214 • "ISR Flow" on page 230 • Table 82: "Interrupt-Related Registers," on page 235 • "Status Register (offset: 0x362C)" on page 391 • "Clock Status Register (offset: 0x3630)" on page 393 • "LSO Read DMA Mode Register (offset: 0x4800)" on page 438 • "NVM Write Register (offset: 0x7008)" on page 497 **Added:** • "Device Closing Procedure" on page 147 • "TX TIME STAMP LSB REG (offset: 0x5C0)" on page 327 • "TX TIME STAMP MSB REG (offset: 0x5C4)" on page 327

Revision	*Date*	*Change Description*
5718-PG103-R	01/26/11	**Updated:** • Added BCM5720 to Section 1: "Introduction," on page 49. • Added BCM5720 to "Introduction" on page 49. • Added Host to BMC to "Transmit MAC Mode Register (offset: 0x45C)" on page 317. • Added Host to BMC to "Transmit MAC Lengths Register (offset: 0x464)" on page 319. • Added Host to BMC to "Mode Control Register (offset: 0x6800)" on page 477. **Added** • "HTX2B Perfect Match[1–4] HI Reg (offset: 0x4880, 0x4888, 0x4890, 0x4898)" on page 330. • "HTX2B Perfect Match[1–4] LO Reg (offset: 0x4884, 0x488C, 0x4894, 0x489C)" on page 330. • "HTX2B Protocol Filter Reg (offset: 0x6D0)" on page 331. • "HTX2B Global Filter Reg (address: 0x6D4)" on page 333. • "H2B Statistics Registers" on page 346. • "HTX2B Statistics" on page 347 • "B2HRX Statistics" on page 347 • "RMU Registers" on page 504 • "RMU_EGRESS_DA1_MATCH[1-8]_REG (offsets: 0x00B0, 0x00B8, 0x00C0, 0x00C8 … 0xE8)" on page 504 • "RMU_EGRESS_DA2_MATCH[1-8]_REG (Offsets 0x00B4, 0x00BC, 0x00C4, 0xCC …0xEC)" on page 504 • "RMU_EGRESS_STATUS_REG (Offset 0x0000)" on page 504

Revision	*Date*	*Change Description*
5718-PG102-R	12/16/10	**Updated:** • Added BCM5719 to "Introduction" on page 39. • Added BCM5719 to "Related Documents" on page 39. • Added BCM5719 to Table 1: "BCM5718 Family Product Features," on page 40. • Removed PHY core column and added BCM5717 B0, BCM5718 BO, and BCM5719 to Table 2: "Family Revision Levels," on page 42. • Updated note in "Revision Levels" on page 42. • Added Memory Arbiter to Figure 1: "Individual Port Functional Block Diagram," on page 45. • Added BCM5719 to "Overview of Features" on page 46. • Added note about BCM5719 to Figure 2: "High-Level System Functional Block Diagram," on page 47. • Added max ring sizes to "Ring Control Block" on page 99. • Added BCM5719 toTable 6: "Defined Flags for Send Buffer Descriptors," on page 67. • Updated Host Ring Size to Table 7: "Receive Return Rings," on page 70. • Corrected typo in Figure 26: "Send Driver Interface," on page 119. • Corrected typo in Figure 27: "Receive Producer Interface," on page 120. • Corrected typo in Figure 28: "Receive Return Interface," on page 121. • Updated Step 36 in "Initialization Procedure" on page 137. • Added BCM5719 to "Description" on page 168. • Corrected typo in "PCI Classcode and Revision ID Register (offset: 0x08) — Function 0" on page 255 • Corrected typos in "Power Management Control/Status Register (offset: 0x4C) — Function 0" on page 261 • Added note to Enable Endian Byte Swap in "Miscellaneous Host Control Register (offset: 0x68)" on page 264. • Updated all Indirection Table register descriptions in "RSS Registers" on page 276. • Added BCM5717 and BCM5718 values to "CPMU Control Register (offset: 0x3600)" on page 346. • Added BCM5719 to "Link Aware Power Mode Clock Policy Register (offset: 0x3610)" on page 349. • Added BCM5719 to "APE CLK Policy Register (offset: 0x361C)" on page 352.

Revision	*Date*	*Change Description*
		Updated (continued): • Added BCM5718 to "Clock Speed Override Policy Register (offset: 0x3624) for BCM5718" on page 354 • Added BCM5718 to "Clock Status Register (offset: 0x3630)" on page 358 • Added Reserved for BCM5719 to "PCIE Status Register (offset: 0x3634)" on page 359 • Added BCM5719 to "GPHY Control/Status Register (offset: 0x3638)" on page 360 • Updated introduction to "PCIE Idle Detection De-Bounce Control Register (offset: 0x364C)" on page 362 • Corrected typo and added BCM5719 to "DLL Lock Timer Register (offset: 0x3654)" on page 364 • Updated Chip ID default value in "CHIP ID Register (offset: 0x3658)" on page 365 • Added BCM5719 to "Padring Control Register (offset: 0x3668)" on page 367 • Added BCM5719 to "Reserved (offset: 0x366C)" on page 368 • Added BCM5719 to "Reserved (offset: 0x367C)" on page 373 • Added BCM5719 to "Read DMA Mode Register (offset: 0x4800)" on page 398 • "LSO Read DMA Corruption Enable Control Register (offset: 0x4910)" on page 413 • Added BCM5719 to "Write DMA Mode Register (offset: 0x4C00)" on page 443 • Added BCM5719 to "MSI Mode Register (offset: 0x6000)" on page 454 **Added:** • "Receive BD Standard Producer Ring Index (High Priority Mailbox) Register (offset: 0x268-0x26f)" on page 276 • "TX Time Stamp LSB Reg (offset: 0x5C0)" on page 282 • "TX Time Stamp MSB Reg (offset: 0x5C4)" on page 283 • "RX Time Stamp LSB Reg (offset 0x06B0)" on page 308 • "RX Time Stamp MSB Reg (offset 0x06B4)" on page 308 • "RX PTP Sequence ID Reg (offset 0x06B8)" on page 308 • "RX Lock Timer LSB Reg (offset 0x6C0)" on page 309 • "RX Lock Timer MSB Reg (offset 0x06C4)" on page 309 • "RX PTP Control Reg (offset: 0x6C8)" on page 310 • "Clock Speed Override Policy Register (offset: 0x3624)" on page 355 • "Clock Status Register (offset: 0x3630)" on page 358 • "Global Mutex Request Register (offset: 0x36F0)" on page 381 • "Global Mutex Grant Register (offset: 0x36F4)" on page 381 • "Temperature Monitor Control Register (offset: 0x36FC)" on page 382 • "BCM5719 Registers" on page 469 **Removed:** • "Reserved (offset: 0x378C)"

Revision	*Date*	*Change Description*
5718-PG101-R	11/12/10	**Updated:** • Table 1: “BCM5718 Family Product Features,” on page 40 • IP cksum description in “Receive Buffer Descriptors” on page 70 • “Extended RX Buffer Descriptor (BD)” on page 110 • Table title for Table 34: “Jumbo Producer Ring Host Address Low Register (offset: 0x2444),” on page 123 • Default value and description in Table 36: “Jumbo Producer Ring NIC Address Register (offset: 0x244C),” on page 123 • NIC ring address values in Table 45: “NIC Ring Addresses,” on page 126 • PCI version in “Description” on page 148 • “EMAC Status Register (offset: 0x404)” on page 279 • “DMA Flag Register for TCP Segmentation (offset: 0xCEC)” on page 321 • “Jumbo Producer Ring NIC Address Register (offset: 0x244C)” on page 337 • “Receive Producer Length/Flags Register (offset: 0x2458)” on page 337 • “Receive Producer Ring NIC Address Register (offset: 0x245C)” on page 338 • “GPHY Strap Register (offset: 0x3664)” on page 366 • “Read DMA Mode Register (offset: 0x4800)” on page 398 • “BCM5718 Family MII Bus PHY Addressing” on page 496 **Added:** • Section 8: “Device Control,” on page 137 • Registers 0x00 to 0x3c and 0x48 to 0x64 to “PCI Configuration Registers” on page 254 • Section 14: “Transceiver Registers,” on page 496 **Removed:** • References to BCM5724 throughout • Column from Table 1: “BCM5718 Family Product Features,” on page 40 • Register control mode from “MDI Register Access” on page 187 • MDI Control Register (offset: 0x6844)
5718-PG100-R	04/13/10	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 BCM5718 family of NetXtreme®/NetLink® Ethernet controllers. This family of controllers includes the following devices:

- BCM5717
- BCM5718
- BCM5719
- BCM5720

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 50) complements this document.

Acronyms and Abbreviations

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

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]

Table 1: Register Access Methods

Convention	*Description*
A/C	Clear contents after read
RO	Read-Only access, Write has no effect
RW	Full Read and Write access

Table 1: Register Access Methods (Cont.)

Convention	*Description*
WO	Write-Only access, Read returns garbage
W1C	Write a value 1 to clear the bit.
W1S	Write a value 1 to set the bit
WZO	Write 0 only to avoid side effects, Read returns garbage
RW1C	Readable / Write a 1'b1 value to clear the bit.
RW1S	Readable / Write a 1'b1 to set the bit
RW/S	Readable / Writeable but is not affected by any CPU reset or AHB reset.

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] BCM57XX NetXtreme® Programmer's Guide: Programming details for the BCM5700, BCM5701, BCM5702, BCM5703, BCM5704, BCM5705, BCM5721, BCM5751, BCM5752, BCM5714, BCM5715, and BCM57XX devices	57XX-PG1XX-R	Broadcom CSP
[2] x2 PCI Express Dual-Port Gigabit Ethernet Controller	5717-DS0X-R	Broadcom CSP
[3] x2 PCI Express Dual-Port Gigabit Ethernet Controller	5718-DS0X-R	Broadcom CSP
[4] x4 PCI Express® Quad-Port Dual-Media Gigabit Ethernet Controller	5719-DS0X-R	Broadcom CSP
[5] x2 PCI Express® Dual-Port Dual-Media Gigabit Ethernet Controller	5720-DS0X-R	Broadcom CSP
[6] x2 PCI Express® Dual-Port Gigabit Ethernet Controller Errata (A0 and B0)	5717-ES10X-R	Broadcom CSP
[7] BCM5718 Revision A0, B0	5718-ES10X-R	Broadcom CSP
[8] BCM5719 A0 Errata	5719-ES10X-R	Broadcom CSP
[9] BCM5720 Errata	5720-ES10X-R	Broadcom CSP
[10] Self Boot Option	5754X_5787X-AN10X-R	Broadcom CSP

Document (or Item) Name (Cont.)	*Number*	*Source*
[11] NetXtreme/NetLink Software Self-Boot NVRAM	NetXtreme-AN40X-R	Broadcom CSP
[12] NetXtreme®/NetLink® NVRAM Access	NetXtreme-AN50X-R	Broadcom CSP
[13] NetXtreme/NetLink NVRAM Configuration Options	NetXtreme-AN60X-R	Broadcom CSP
[14] NetXtreme/NetLink Shared Memory Communication	NetXtreme-AN80X-R	Broadcom CSP
Other Items		
[15] NC-SI (Network Controller–Sideband Interface) Specification	–	(http://www.dmtf.org/home)

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

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 2: BCM5718 Family Product Features

Feature	*BCM5717 Dual-Port Copper*	*BCM5718 Dual-Port Copper/Serdes*	*BCM5719 Quad-Port Copper/Serdes*	*BCM5720 Dual-Port Copper/Serdes*
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	32 KB RX, 29 KB TX	32 KB RX, 29 KB TX	40 KB RX, 29 KB TX	40 KB RX, 29 KB TX
TCP checksum offload (hardware based) on Tx/Rx over IPv4/IPv6	Yes	Yes	Yes	Yes
UDP checksum offload (hardware based) on Tx/Rx over IPv4/IPv6	Yes	Yes	Yes	Yes
IP checksum offload on Tx/Rx over IPv4/IPv6	Yes	Yes	Yes	Yes
Hardware TCP segmentation offload over IPv4/IPv6	Yes	Yes	Yes	Yes
Jumbo frame support	Yes	Yes	Yes	Yes
Receive-side scaling (RSS)	Yes	Yes	Yes	Yes
UDP Receive-side scaling (UDP RSS)	No	No	Yes	Yes

Table 2: BCM5718 Family Product Features (Cont.)

Feature	*BCM5717 Dual-Port Copper*	*BCM5718 Dual-Port Copper/Serdes*	*BCM5719 Quad-Port Copper/Serdes*	*BCM5720 Dual-Port Copper/Serdes*
Transmit-side scaling (TSS)	Yes	Yes	Yes	Yes
Multiple receive descriptor queues	Yes	Yes	Yes	Yes
IOV support (I/O Virtualization) for: • VMWare® NetQueue • Microsoft® Virtual Machine Queue (VMQ)	No	Yes	Yes	Yes
MSI	Yes	Yes	Yes	Yes
MSI-X	Yes (5 Vectors)	Yes (17 Vectors)	Yes (17 Vectors)	Yes (17 Vectors)
Function Level Reset	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	No	No	No	No
iSCSI Boot	Yes	Yes	Yes	Yes
Teaming	Yes	Yes	Yes	Yes
Host Bus Interfaces				
PCIe 2.0 x2	Yes	Yes	Yes	Yes
PCIe 2.1 x4	No	No	Yes	No
LAN Interfaces				
Integrated 10/100/1000 transceiver	Yes	Yes	Yes	Yes
Internal MII/GMII Interface	Yes	Yes	Yes	Yes
Other Bus Interfaces				
NC-SI (version 1.0.0a)	Yes	Yes	Yes	Yes
Host 2 BMC	No	No	No	Yes
SMBus 2.0 interface	Yes	Yes	Yes	Yes
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
JTAG support	Yes	Yes	Yes	Yes
Technology				
High-performance, low-overhead software/hardware interface	Yes	Yes	Yes	Yes

Table 2: BCM5718 Family Product Features (Cont.)

Feature	*BCM5717 Dual-Port Copper*	*BCM5718 Dual-Port Copper/Serdes*	*BCM5719 Quad-Port Copper/Serdes*	*BCM5720 Dual-Port Copper/Serdes*
High-speed on-chip RISC processors (one per port)	Yes	Yes	Yes	Yes
High-speed on-chip Application Processor Engine (APE)	Yes	Yes	Yes	Yes
Wake-on-LAN (WOL)	Yes	Yes	Yes	Yes
Energy Efficient Ethernet (EEE)	No	Yes	Yes	Yes
Ethernet Audio Video (EAV)	No	No	No	No
Secure Digital (SD) Card Reader	No	No	No	No
IEEE 1588 / IEEE 802.1AS Timestamp Support	No	No	Yes	Yes
On-chip temperature monitor	No	No	Yes	Yes
Integrated Trusted Platform Module (TPM) Security Engine	No	No	No	No
Process voltage	1.2V	1.2V	1.2V	1.2V
CMOS linewidth	65 nm	65 nm	65 nm	65 nm

Revision Levels

See Table 3 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 3 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 68 for more details).

Table 3: Family Revision Levels

Family Member	*Device ID*[a]	*Revision Level*	*PCI Revision ID*[b]	*Chip ID*[c]	*Product ASIC ID*[d]	*Errata Sheet*[e]
BCM5717	0x1655	A0	0x00	0xF000xxxx	0X5717000	5717-ES1xx-R
BCM5717	0x1655	B0	0x10	0xF100xxxx	0X5717100	5717-ES1xx-R
BCM5718	0x1656	A0	0x00	0xF000xxxx	0X5717000	5718-ES1xx-R
BCM5718	0x1656	B0	0x10	0xF100xxxx	0X5717100	5718-ES1xx-R
BCM5719	0x1657	A0	0x00	0xF000xxxx	0X5719000	5719-ES1xx-R
BCM5719	0x1657	A1	0x01	0XF100XXXX	0X5719100	5719-ES1XX-R

Table 3: Family Revision Levels (Cont.)

Family Member	*Device ID[a]*	*Revision Level*	*PCI Revision ID[b]*	*Chip ID[c]*	*Product ASIC ID[d]*	*Errata Sheet[e]*
BCM5720	0X165F	A0	0x00	0xF000xxxx	0X5720000	5720-ES1xx-R

a. See Device ID and Vendor ID Register (Offset: 0x00) — Function 0", per PCI specification.
b. See "PCI Classcode and Revision ID Register (offset: 0x8) — Function 0" as per the PCI specification. 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)" on page 283. The lower 16 bits are don't cares for determining chip id.
d. See "Product ASIC ID (offset: 0xF4)" on page 301 or determining ASIC ID.
e. See the appropriate errata documentation for the errata information and resolutions.

Note: If you are using silicon revision A0 of the BCM5718, you must load a "patch" image (ap5718.012) into the boot code NVRAM in addition to the usual legacy boot code image common to all NetXtreme/NetLink controllers. This extra patch image, which masquerades as "Management Firmware", works around an issue with A0 silicon in which the device may fail to load reliably and execute the primary boot code image. This issue is fixed in BCM5718 B0 and is not an issue in BCM5719 or BCM5720.

Programming the Ethernet Controllers

See Table 3 on page 50 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 "Mailbox Registers" on page 93).
- Indirect mode uses the low-priority mailboxes (see "Mailbox Registers" on page 93).

The reference documents for Ethernet controller software development include this manual and the errata documentation (see "Revision Levels" on page 50) 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®).

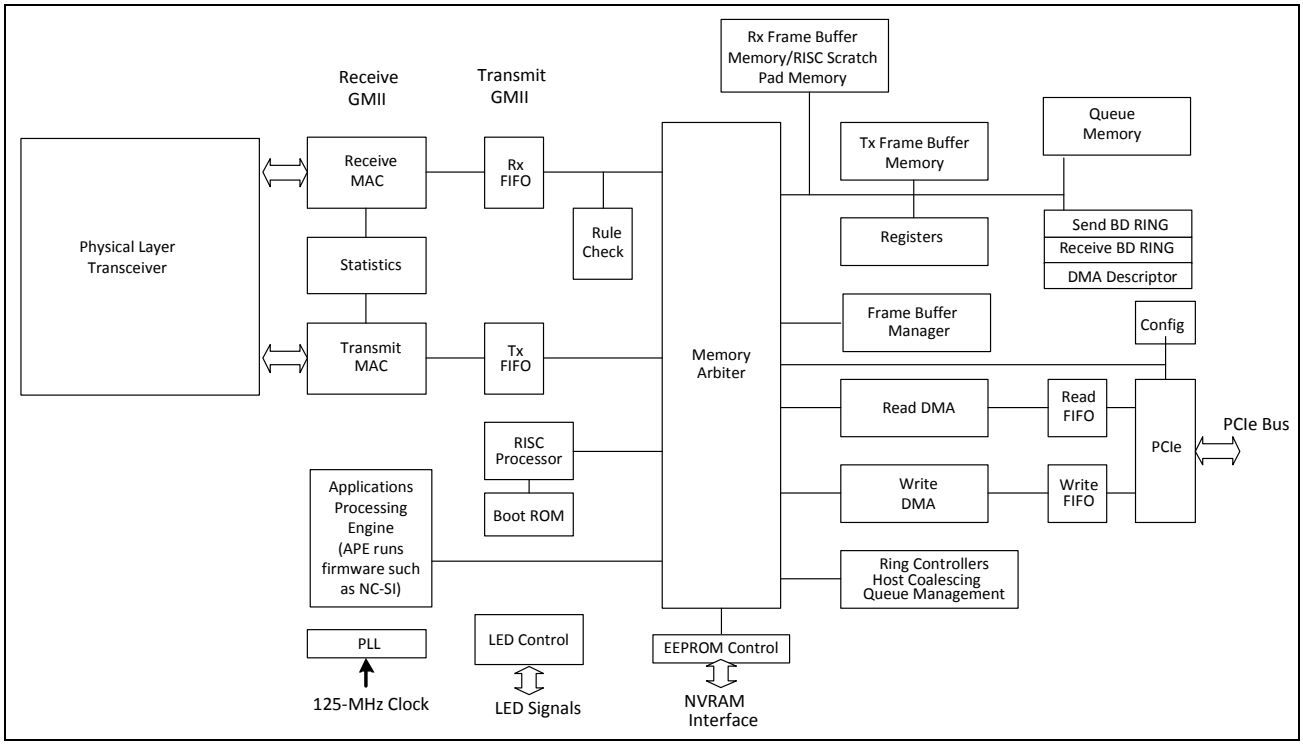
Section 2: Hardware Architecture

Theory of Operation

Figure 1 shows the major functional blocks and interfaces of the Ethernet controllers covered in this document. Only a single port is illustrated in Figure 1. The dual-port controllers in this family of controllers essentially replicate a second instance of the major areas of functionality shown in the diagram below. The dual-port controllers have only a single PCIe and NVRAM interfaces.

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: Individual Port Functional Block Diagram



Overview of Features

The BCM5718 family of controllers represents the third generation of Broadcom NetXtreme multi-port gigabit Ethernet controllers. This family is the successor to the BCM5714/BCM5715 family. The BCM5717, BCM5718, and BCM5720 feature two independent 1 Gb Ethernet ports on the network side. A host computer can communicate with the controller over a single PCIe link. However, the two network controller ports appear as two independent PCIe functions to the host operating system (four functions in the quad-port BCM5719). In essence, the chip consists of a single PCIe interface controller that offers multiple function-level interfaces, and each function-level interface is further attached to an independent DMA engine which in turn feeds an independent Ethernet Media Access Controller (EMAC).

Attached to the other end of each EMAC is the respective 802.3 Ethernet physical media interface. The controller essentially consists of multiple DMA+EMAC logic instances, multiple Ethernet physical interfaces, and a single instance of a PCIe core that is shared by the DMA blocks.

The BCM5718 is available as four SKUs, BCM5717, BCM5718, BCM5719 and BCM5720. These are also referred to as the Dual-Copper SKU and the Dual-Media SKU. For the BCM5718, BCM5719, and BCM5720 part, any of the network ports may be independently configured as a copper-based (1000BASE-T) or as SerDes-based (1000BASE-X/SGMII) media interfaces. The choice of media interface on each port is configurable via a power-on strapping option.

The BCM5717 part is permanently bonded as a copper (1000BASE-T) only device.

For the Dual-Media SKU (BCM5718 and BCM5719), whenever a port is configured as a SerDes medium, there are two protocol choices: 1000BASE-X or SGMII. This choice can be made by an Auto-Detect feature or by explicit software programming.

The software driver for this device is capable of loading or unloading each network port independently. The DMA+EMAC associated with each port is also able to acquire different ACPI power states irrespective of the other; however, the power state of the PCIe link may not necessarily follow that of either one or both ports. This is a significant behavioral difference of a dual-port controllers compared to single-port. The BCM5718 family hardware and firmware is cognizant of this effect and adds the necessary intelligence to handle it. This functionality remains transparent to device driver software.

On top of the basic dual-port network controller functionality, the BCM5718, BCM5719, and BCM5720 also support an advanced feature known as IO Virtualization (IOV).

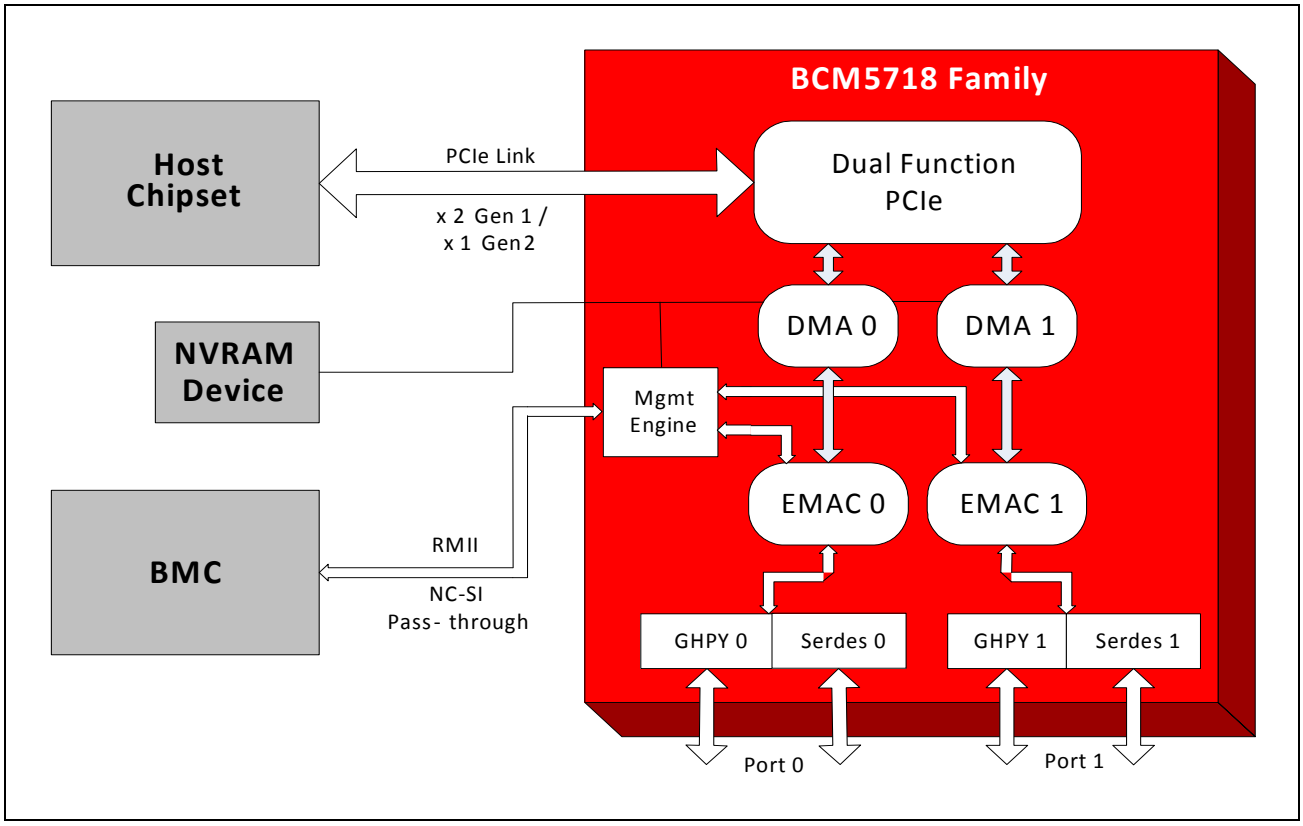
Network Controller Sideband Interface (NC-SI) pass-through functionality is the Server Management solution offered by the BCM5718 family of controllers. NC-SI pass-through is a part of the Server Management infrastructure. Such technology offers server platform management via a BMC (baseboard management controller) chip. A server platform and, in turn, a BMC is typically administered remotely over the network, but BMCs are not equipped with direct network connectivity. Here, a network controller chip comes into the picture—a pass-through-capable NIC chip offers a sideband packet interface to the BMC. Over this interface the BMC can send and receive Ethernet packets to and from the network. In essence, the network adapter functionality of the network controller chip is shared by the host computer system and the BMC chip. NC-SI pass-through protocol is an industry standard (DMTF) for a side band interface between the BMC and network controller. The physical and L2 layers of NC-SI are upwardly compliant to the respective sections of the IEEE 802.3 specification, thus allowing the exchange of Ethernet packets between the network port and BMC without any protocol transformation whatsoever.

In an environment of several network servers, typically no server is dedicated to running only the management console. Host to BMC pass-through functionality identifies host to local BMC-bound packets and routes them internally to the chip. Similarly, BMC to local host-bound packets are also identified and routed internally to the NIC chip. Host to BMC pass-through functionality is offered by the BCM5720 controller.

In addition to IEEE 802.3 standard size Ethernet frames, the BCM5718 family also supports jumbo frames of sizes up to 9622 bytes.

The BCM5718 family of controllers replaces the traditional PNP-based linear regulator with a more efficient switching regulator. This regulator steps down system supplied 3.3V rail to 1.2V, which it supplies to the chip's core.

Figure 2: High-Level System Functional Block Diagram



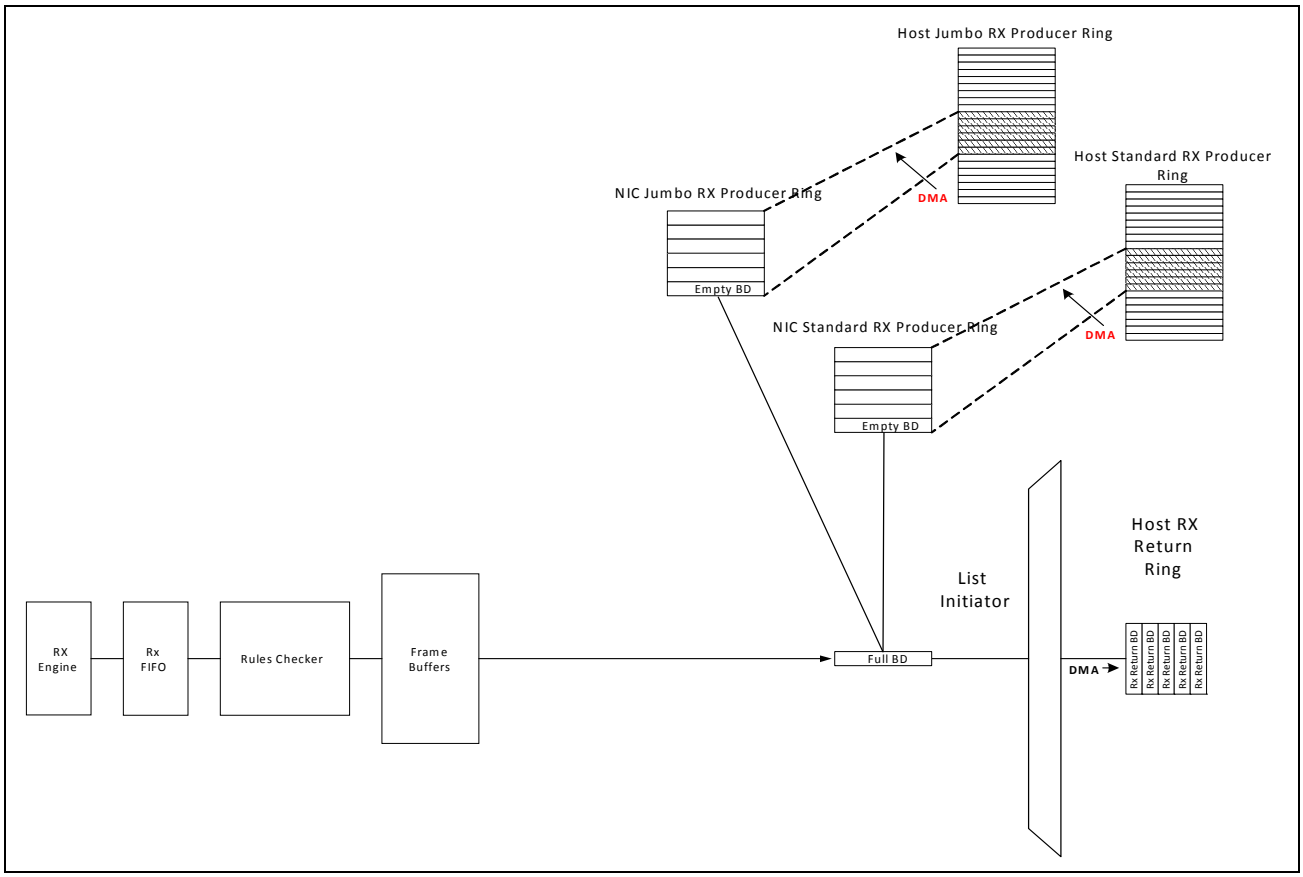
Note: BCM5719 has the same general architecture, but four ports (ports 0, 1, 2, 3) and a quad-function PCIe interface.

Receive Data Path

RX Engine

The receive engine (see Figure 3) activates whenever a packet arrives from the PHY.

Figure 3: 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

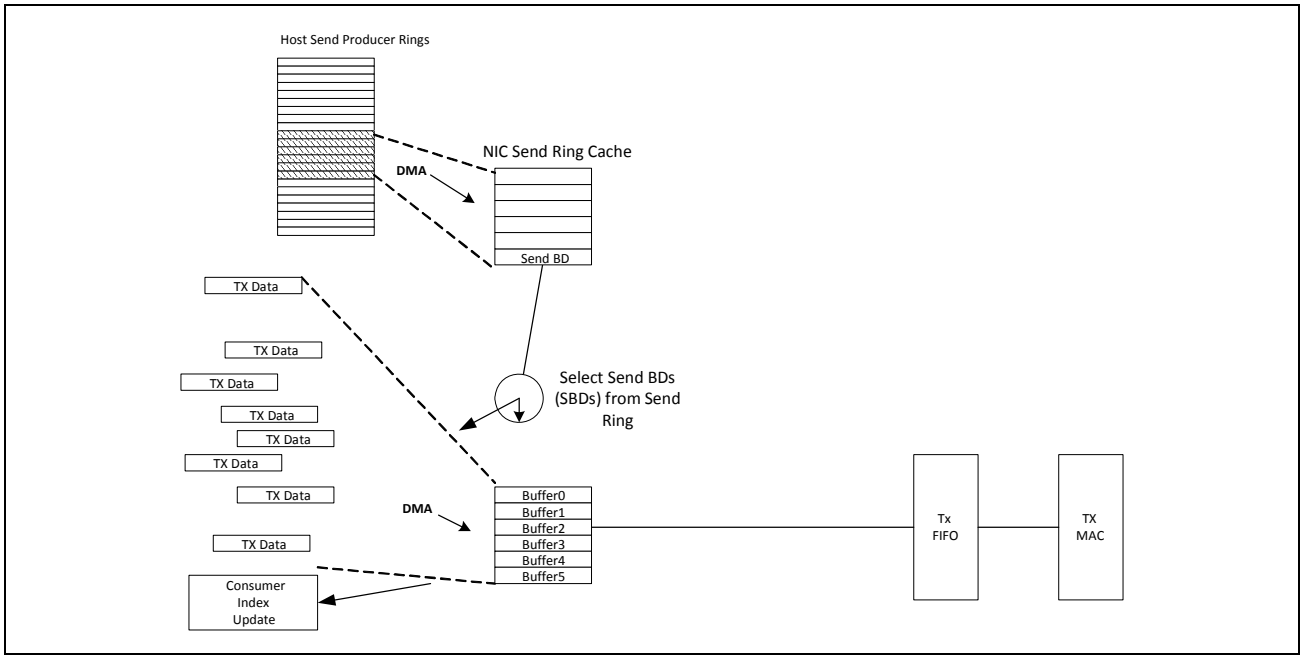
The RX List Initiator function activates whenever the receive producer index for any of receive buffer descriptor (BD) rings is written. This value is located in one of the receive BD producer mailboxes. The host software writes to the producer mailbox and causes the RX Initiator function to enqueue 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 4). When the entire packet is available, the transmit MAC is activated.

Figure 4: 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

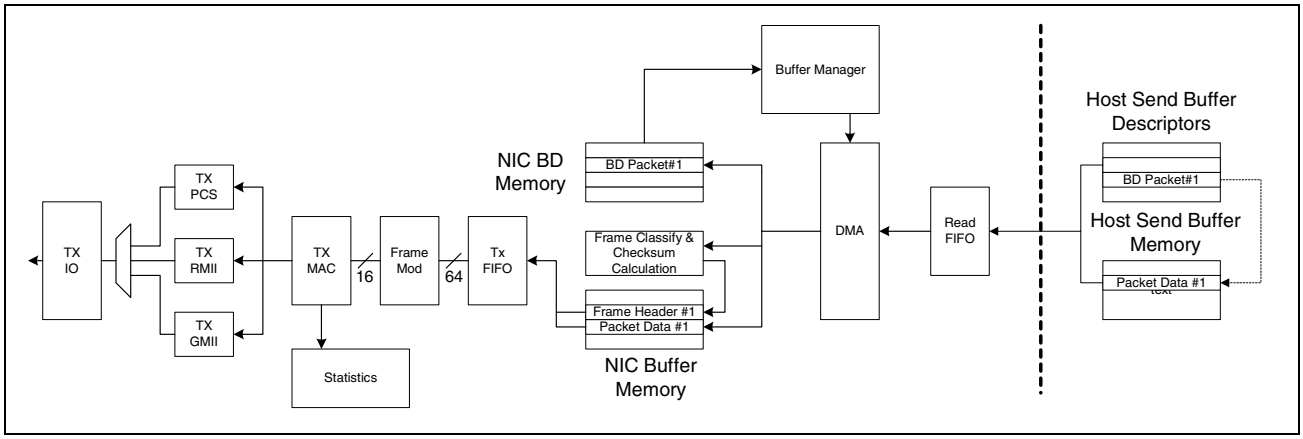
The TX FIFO provides elasticity while data is moved from device internal memory to PHY. There are no programmable settings for the TX FIFO. This FIFO's operation is completely transparent to host software.

DMA Read

Read Engine

The DMA read engine (see Figure 5) activates whenever a host read is initiated by the send or receive data paths.

Figure 5: DMA Read Engine



The DMA read engine dequeues 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

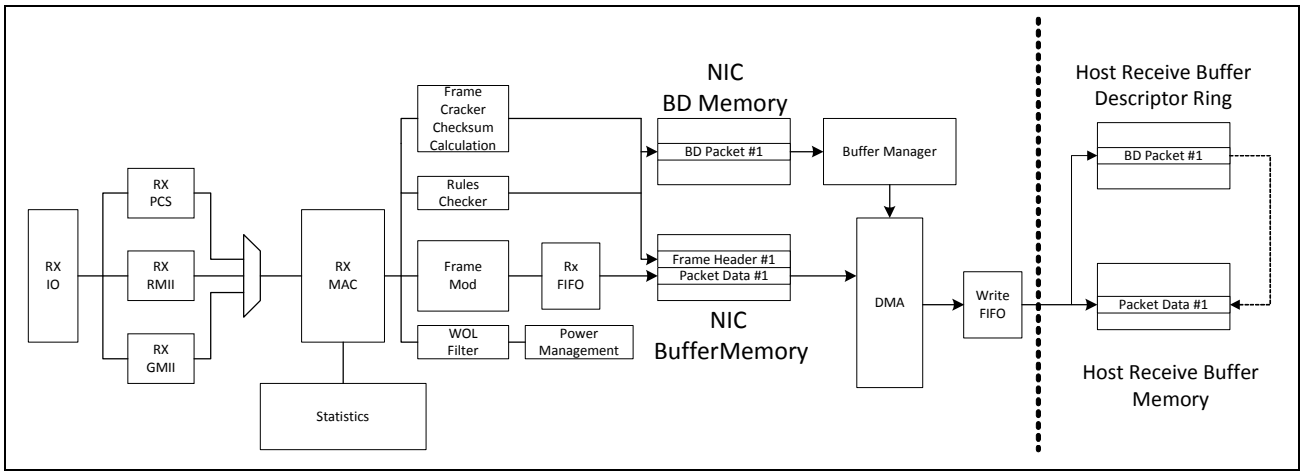
The buffer manager maintains pools of internal memory used by transmit and receive engines. The buffer manager has logic blocks for allocation, free, control, and initialization of internal memory pools. The DMA read engine requests internal memory for BDs and frame data. Figure 5 on page 58 shows the 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 6, activates when a host write is initiated by the send or receive data paths.

Figure 6: DMA Write Engine



The DMA write engine dequeues 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

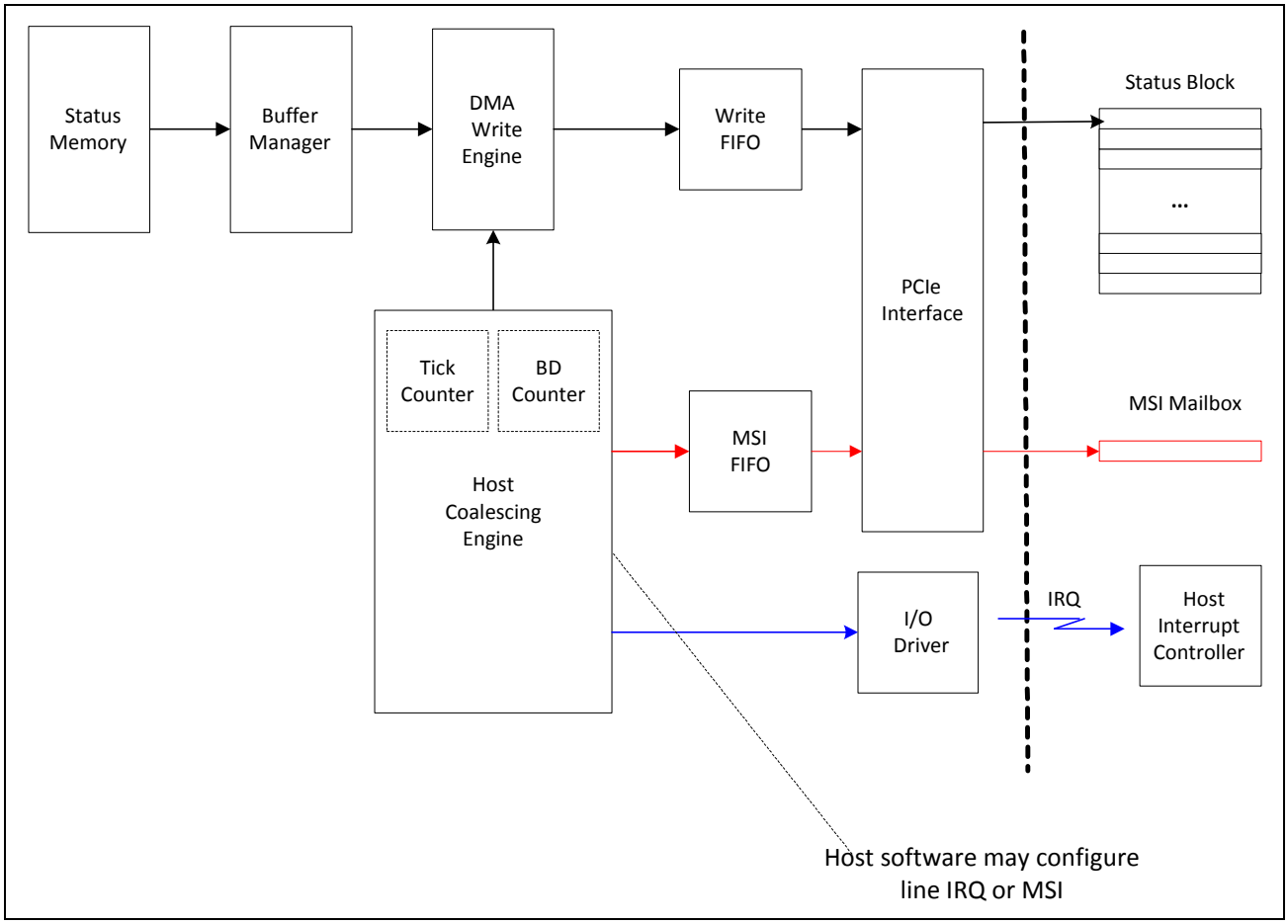
The Memory Arbiter (MA) is a gatekeeper for internal memory access. The MA is responsible for decoding the internal memory addresses that correspond to Ethernet controller data structures and control maps. If a functional block faults or traps during access to internal memory, the MA handles the failing 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 7: 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 255).
- 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 Register (Offset: 0x3c14)" on page 256). Updates can occur when the number of BDs (not frames) meets the thresholds set in the various coalescing registers (see Section 11: "Interrupt Processing," on page 230 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 255).
- 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 84 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

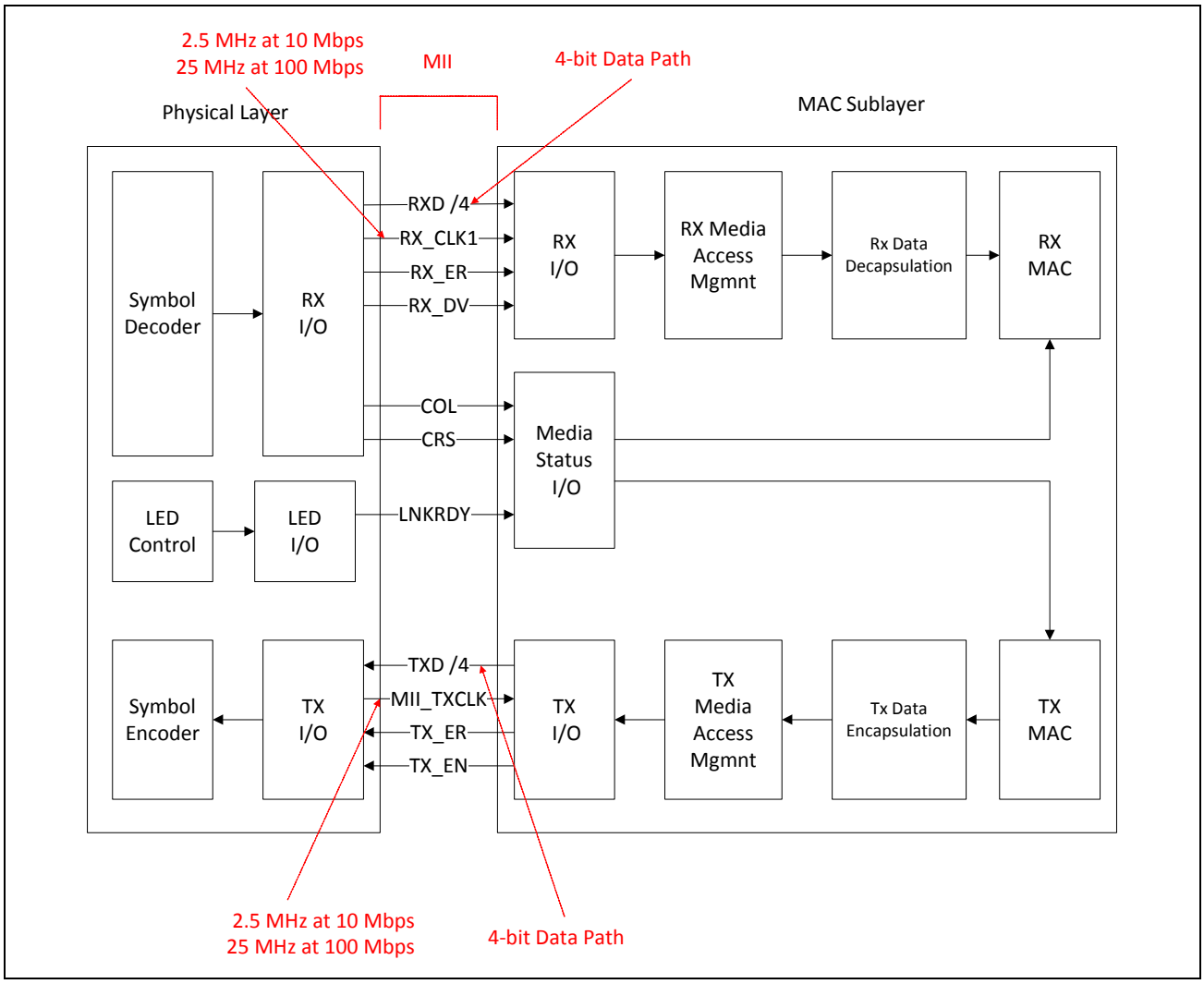
The NetXtreme/NetLink Ethernet controller supports the following physical layer interfaces:

- 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 8 on page 64).

Figure 8: 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 9 on page 66); the send and receive data paths operate independently.

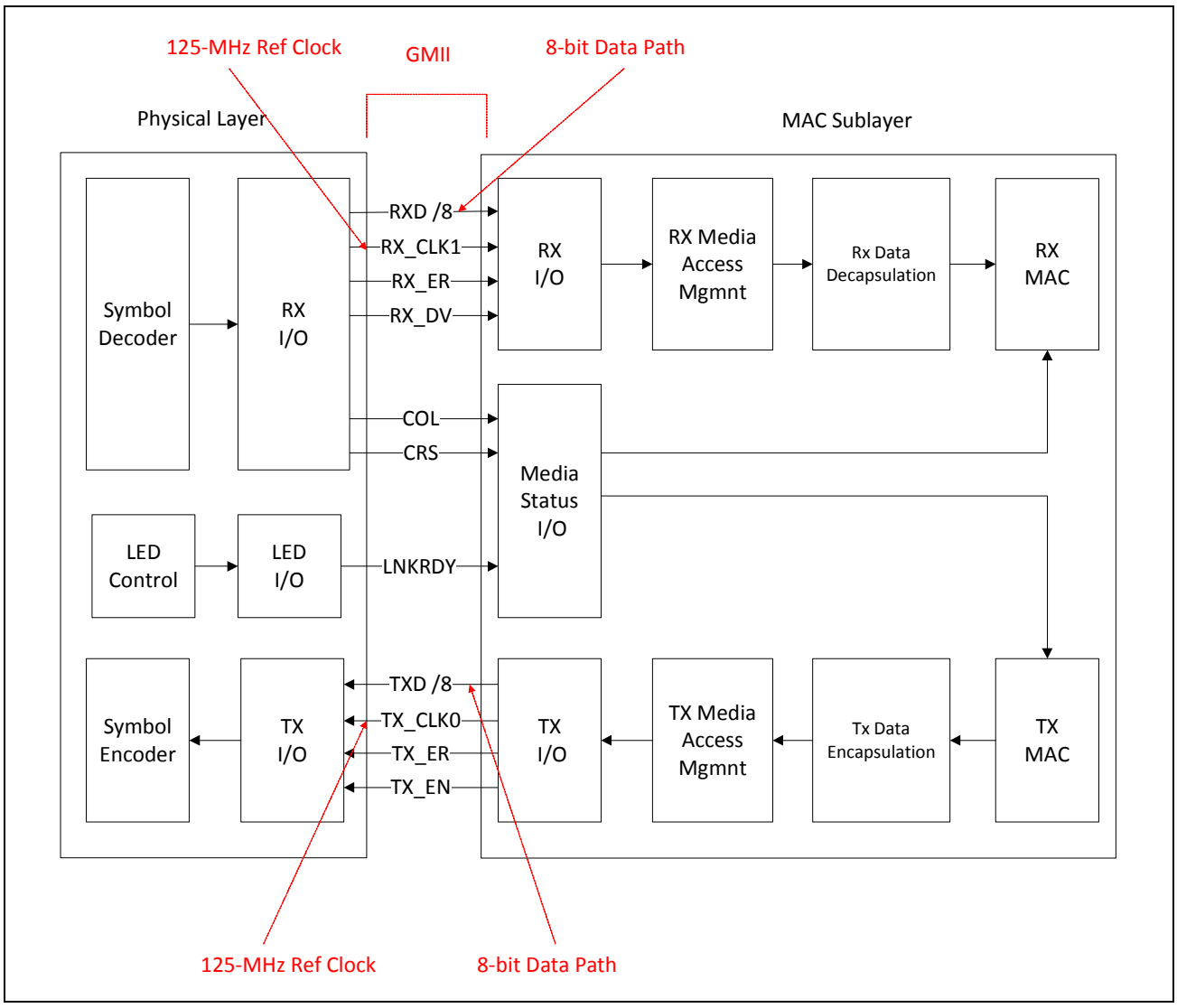
The transmit signals TXD[7:0] create a eight-bit wide data path. The TXD[7:0] signals are synchronized to the reference clockTX_CLK0. The TX_CLK0 clock runs at 125 MHz and is sourced by the MAC sublayer. Transmit Error (TX_ER) is asserted by the MAC sublayer. The PHY will transmit a bad code until TX_ER 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.

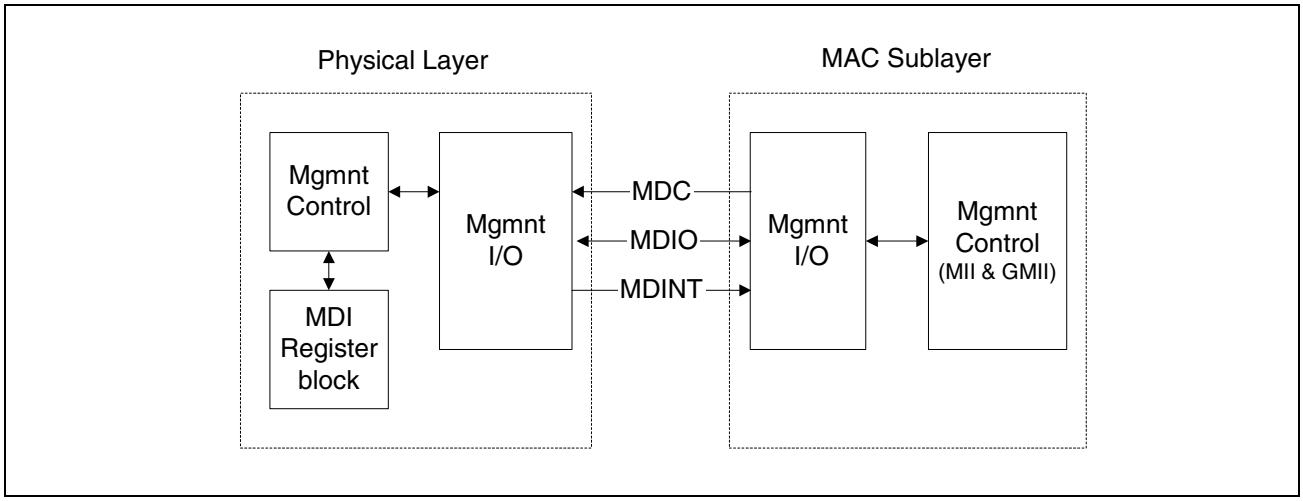
Figure 9: GMII Block



MDIO Register Interface

Figure 10 shows the MDI register interface.

Figure 10: 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 BCM5718 family 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) for the purpose of housing OEM-programmable configuration items (refer to Application Note NetXtreme-AN60x-R "NetXtreme/NetLink NVRAM Configuration Options").

Note: If using self-boot, the design must use a very small (approximately 256 byte) external NVRAM to hold configuration items and self-boot code patches.

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

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:

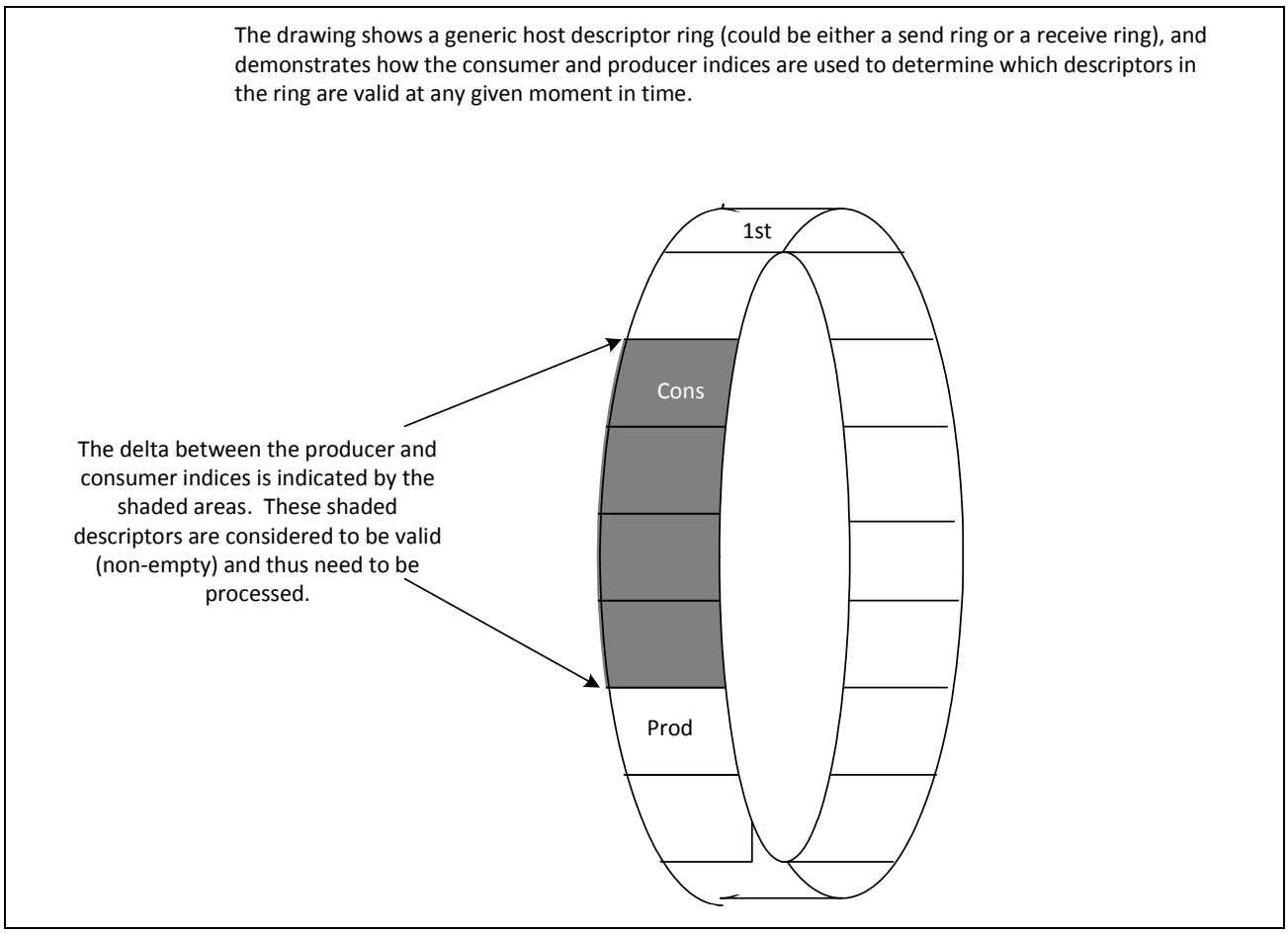
- Send Rings
- Receive Producer Rings
- Receive Return Rings

The TX/RX ring base requires an 8 byte alignment. The receive buffer address (recorded in SBD/RBD) cannot cross 4G.

Producer and Consumer Indices

The Producer Index and the Consumer Index control which descriptors are valid for a given ring. Each ring will have 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 mark which descriptors are currently valid in the ring (see Figure 11). 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 11: Generic Ring Diagram



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	Reserved			

The fields are defined as follows:

- The Host Ring Address field contains the 64-bit host address of the first element in the ring. Basically, this field tells the controller precisely where in host memory the ring is located. This field only applies to rings that are 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 flags that are defined.

Table 5: Flag Fields for a Ring

Bits	*Name*	*Description*
0	Reserved	Should be set to 0.
1	RCB_FLAG_RING_DISABLED	Indicates that the ring is not in use.
15:2	STD_MAX_PACKET_SIZE	Indicates maximum frame size for the ring.

- 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, 512, 1024, 2048, and 4096.
- Max ring sizes supported in the BCM5718 Family are shown below.
  - Rx Return: 4096
  - Rx Producer: 2048
  - Rx Producer Jumbo: 1024
  - Tx Producer: 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 "PCI" on page 168.

Send Ring Control Blocks

The format of the Send RCB remains unchanged, only 15 more are added as shown in the Table below.

Table 6: Send RCBs for Multiple Rings

Send Ring#	*Register Name*	*Send Ring RCB Register Address (**)*	*Usable in*
Legacy / 1	Host Address High	0x100	Legacy Mode, RSS Mode & IOV Mode
	Host Address Low	0x104	
	Max Length/Flag	0x108	
	NIC Address	0x10C	
2	Host Address High	0x110	RSS Mode & IOV Mode
	Host Address Low	0x114	
	Max Length/Flag	0x118	
	NIC Address	0x11C	
3			
4			
5			IOV Mode Only
….	……	……	
16	Host Address High	0x1F0	
	Host Address Low	0x1F4	
	Max Length/Flag	0x1F8	
	NIC Address	0x1FC	
** These are Memory Mapped registers			

Note: Address range [0x100 - 0x10F] is the legacy single RCB address and is also used by RCB1.

Receive Ring Control Blocks

Table 7 lists all of the receive RCB Register Addresses.

Table 7: High Priority Mail Box Registers for VRQ Rings

VRQ #	*Register Name*	*Standard Ring RCB Register Address*	*Jumbo Ring RCB Register Address*	*Return Ring RCB Register Address (**)*
0 (Default)	Host Address High	0x2450	0x2440	0x200
	Host Address Low	0x2454	0x2444	0x204
	Max Length/Flag	0x2458	0x2448	0x208
	NIC Address	0x245C	0x244C	0x20C

Table 7: High Priority Mail Box Registers for VRQ Rings (Cont.)

VRQ #	Register Name	Standard Ring RCB Register Address	Jumbo Ring RCB Register Address	Return Ring RCB Register Address (**)
1	Host Address High	0x2510	0x2500	0x210
	Host Address Low	0x2514	0x2504	0x214
	Max Length/Flag	0x2518	0x2508	0x218
	NIC Address	0x251C	0x250C	0x21C
2	Host Address High	0x2530	0x2520	0x220
	Host Address Low	0x2534	0x2524	0x224
	Max Length/Flag	0x2538	0x2528	0x228
	NIC Address	0x253C	0x252C	0x22C
3	Host Address High	0x2550	0x2540	0x230
	Host Address Low	0x2554	0x2544	0x234
	Max Length/Flag	0x2558	0x2548	0x238
	NIC Address	0x255C	0x254C	0x23C
4	Host Address High	0x2570	0x2560	0x240
.....	.....	.....	.....	.....
16	Host Address High	0x26F0	0x26E0	0x300
	Host Address Low	0x26F4	0x26E4	0x304
	Max Length/Flag	0x26F8	0x26E8	0x308
	NIC Address	0x26FC	0x26EC	0x30C

Note: [0x2450 .. 0x245C] and [0x2440 .. 0x244C] are legacy RCB addresses and are being assigned to VRQ Ring# 0.

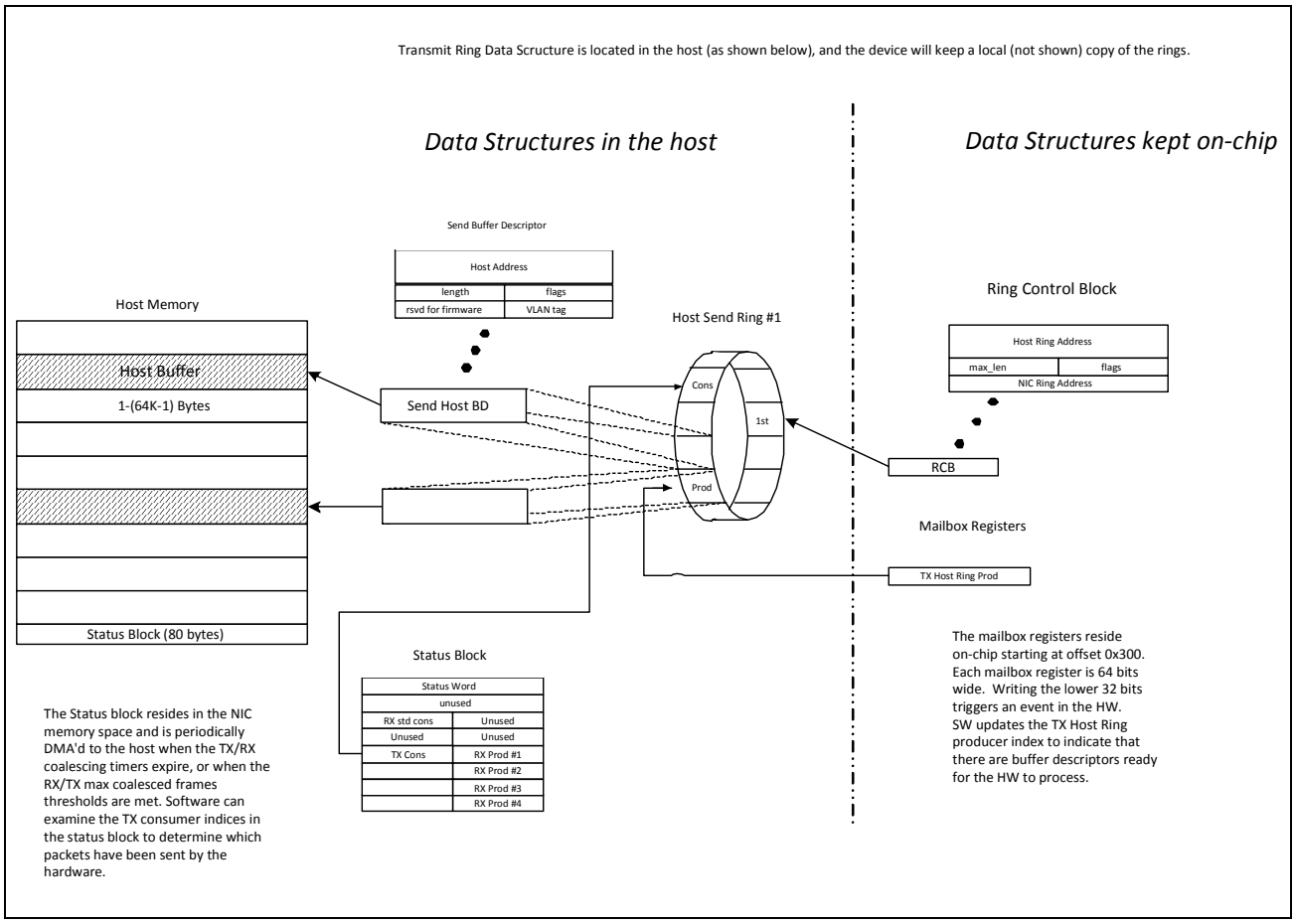
Note: The Return Ring RCBs are Memory Mapped. Memory Address [0x200 .. 0x23C] are legacy Return Ring RCB addresses and are being assigned to VRQ Return Ring# 0 through Ring# 3

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 12 on page 75). 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 12: Transmit Ring Data Structure Architecture Diagram



Send Buffer Descriptors

Standard (Not Large Segment Offload)

The format of an individual send buffer descriptor is shown in Table 8.

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 9.

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.
3	Jumbo Frame	The 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 Ether+IP+TCP Headers to be replicated in each segment arising out of a Large TCP Segment (LSO).
5	Capture Time Stamp	(BCM5719/5720 only) If this bit is 1, this frame's launch time shall be captured in the TX Time-Stamp Register.
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.

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

Bits	Name	Description
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.
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.
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 is 256B.
11	HDRLEN[4]	–
12	HDRLEN[5]	–
13	HDRLEN[6]	–
14	HDRLEN[7]	–
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).

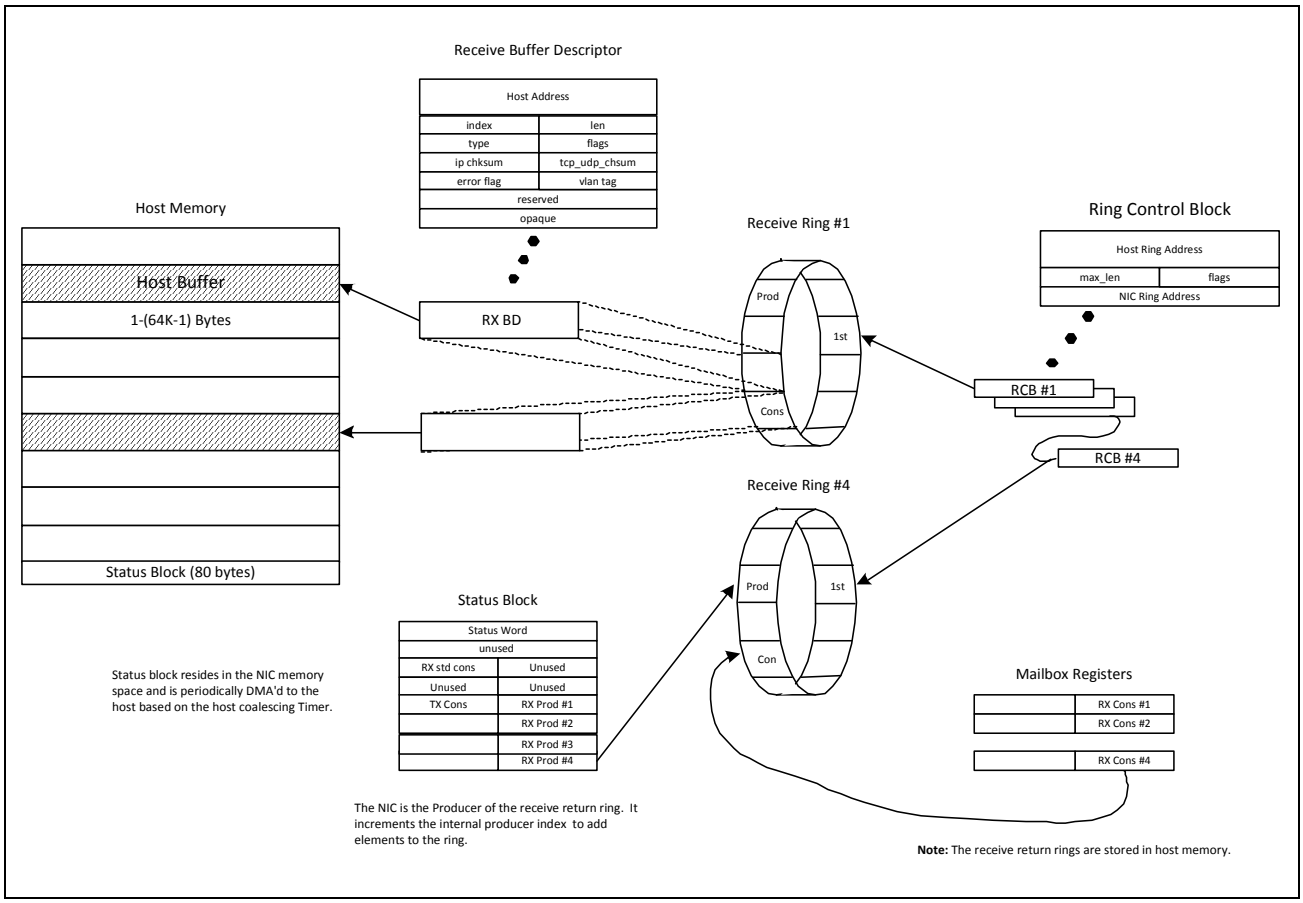
Large Segment Offload (LSO) Send BD

See "Large Segment Offload" on page 111.

Receive Rings

The Ethernet controllers support two types of Receive Descriptor Rings: Producer Rings and Return Rings (see Figure 13 on page 78). Unlike previous NetXtreme controllers, the BCM5718 family adds a second Receive Producer ring dedicated to jumbo frame reception. 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 13: Receive Return Ring Memory Architecture Diagram



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 89). 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	*BCM5718 Family*
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 or 1024 or 208 or 4096
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)	*3116*	*150*
0x00 0x04	Host Address	
0x08	Index	Length
0x0c	Type	Flags
0x10	IP_Cksum	TCP_UDP_Cksum
0x14	Error_Flags	VLAN tag
0x18	RSS Hash	
0x1C	Opaque	

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 13.

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.
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 323 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.

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

Bits	Name	Description
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 472) 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 — Host software should zero this value in the producer ring descriptors. For Receive Return Ring descriptors, if using IP checksum offload, the host driver software should rely on the Flags bit IP_CHECKSUM (Flags bit 12) to determine if the IP checksum in the received packet is correct. This field used to contain the actual IP checksum value but that is not true for the BCM5718 family of controllers. Only the Flags bit IP_CHECKSUM should be relied on by host driver software as is done by Broadcom drivers.
- 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 317). 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.

Table 13: Defined Error Flags for Receive Buffers (Cont.)

Bits	Name	Description
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.

Additional Ring Information for the BCM5718 Family

Rings in the BCM5718 Family are more complicated than in previous NetXtreme controllers. Before considering ring structure details, first choose the mode of operation:

- Legacy
- RSS
- RSS+TSS
- IOV

Note: RSS (and/or TSS) and IOV are mutually exclusive.

Once the mode of operation is decided, determine if jumbo frames are to be used.

Setting ring sizes involves setting some ring sizes (such as the Rx Producer Rings) in registers and setting other ring sizes (such as the Rx Return Rings) in memory locations (for example, the upper 16 bits of the 32 bit value at memory offset 0x208 for Rx Return Ring number 0, rather than in a register). See Appendix C: "Device Register and Memory Map," on page 588 for more information. A Ring Control Block (RCB) consists of four 32 bit words:

- host address high
- host address low
- len/flags
- NIC ring address

For example, the RX Return Ring size is set in the high 16 bits of 32bit value in device memory offset 0x208.

If using RSS or IOV, there are either 4 or 17 Rx Return and Producer rings. The 17th Rx ring is a default throw-away ring for any Rx traffic that does not map to rings 0-3 (RSS) or 0-15 (IOV). Driver software only pulls valid Rx BDs from 16 Rx Ret Rings (rings 0-15) in IOV mode.

For Rx Producer Rings, there are only 1 or 2 in non-IOV mode (the second one dedicated to the driver to supply jumbo Rx BDs to the chip, if using jumbo frames). In RSS mode, there are up to 4. In IOV mode, there are up to 16 (17, including the default ring).

The Tx path is simpler. Tx is 1 or 4 or 16 rings (legacy, TSS, or IOV) for giving Send BDs to the chip. Jumbo BDs go onto the same Send ring as non-jumbos. There are no separate jumbo-dedicated Send ring(s) like there are for Rx producer rings.

The max sizes of the various rings is as follow:

- Rx Ret: 4096
- Rx Prod: 2048
- Rx Prod Jumbo: 1024
- Tx Prod: 512

As a maximum use example, there can be an IOV implementation with the following total max ring size/count arrangement:

- 16 Send rings (size=512 each ring)
- 16 Rx Prod rings (size=2048 each ring)
- 16 Rx Prod jumbo rings (size=1024 each ring)
- 17 Rx Ret rings (size=4096 each ring)

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 61). 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 70. 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 and the 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 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 — 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 16: Status Block [0] Format (MSI-X Multivector 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	Reserved 0x0
0x14	Reserved 0x0	Receive Jumbo Producer Ring Consumer Index

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

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

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

- Bit [0]: Update bit
- Bit [31:1]: Reserved 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 18: Status Block Host Addresses and INT MailBox Addresses

		RSS Mode		
Status Block #	***Status Block Host Address Register (64-Bit)***	***INT MailBox Register Address***	***Indication Items***	***Comments***
Legacy	0x3C3C, 0x3C38	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	Used in all MSI-X modes for Vector#0
1	0x3D00, 0x3D04	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	
3	0x3D10, 0x3D14	0x218	Rx Return Ring 2 Prod Index	
4	0x3D18, 0x3D1C	0x220	Rx Return Ring 3 Prod Index	
5	0x3D20, 0x3D24	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 19 on page 87.

Table 19: 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 312. See "Wake on LAN Mode/Low-Power" on page 213 for a description of the PHY setup required when link state changes.

Table 19: 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 424. • MAC_ATTN—Events from the MAC block (see "EMAC Status Register (offset: 0x404)" on page 312. • DMA_EVENT—Events from the following blocks: – MSI (see "MSI Status Register (offset: 0x6004)" on page 470. – DMA_RD (see "LSO Read DMA Status Register (offset: 0x4804)" on page 437. – DMA_WR (see "Write DMA Status Register (offset: 0x4C04)" on page 453. • RXCP_ATTN—Events from RX RISC (see "RX RISC Status Register (offset: 0x5004)" on page 455.

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)" on page 283) is set to 1. This Status Tag can be returned to the Mailbox 0 register at location 31:24 (see"Interrupt Mailbox 0 Register (offset: 0x5800)" on page 463) 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

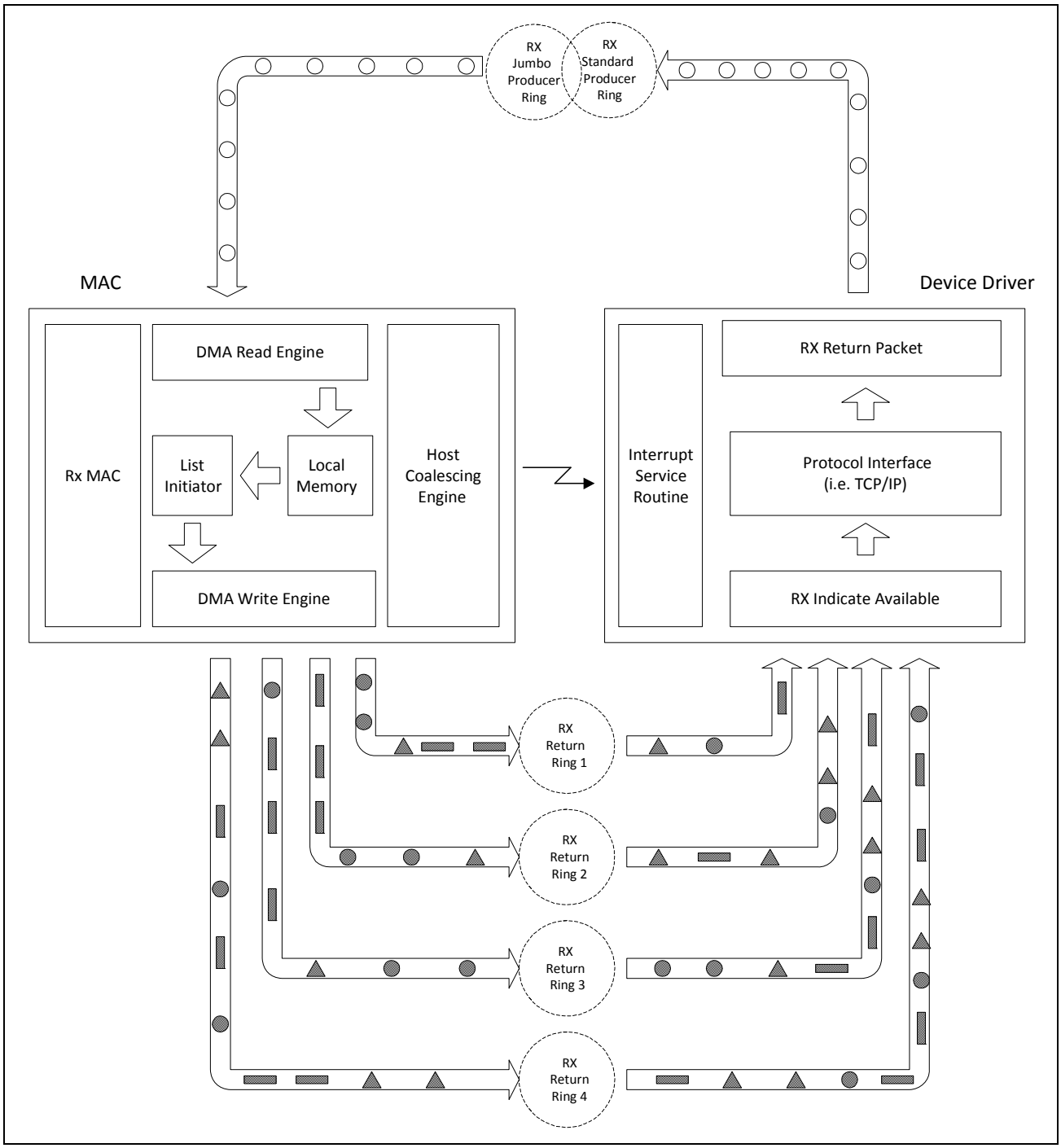
The RX MAC pulls BDs from RX producer rings. The RX BD specifies the location(s) in host memory where packet data may be written. Figure 14 on page 90 shows the receive buffer descriptor cycle.

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 96). 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 235 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 14: 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 a portion 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.

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 set of registers (or internal device memory offsets) 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 375
  - "Receive Data Completion Control Registers" on page 373.

  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 317). 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. The BCM5717 does not support jumbo frames.
- Receive MAC Mode register ("Receive MAC Mode Register (offset: 0x468)" on page 323). 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

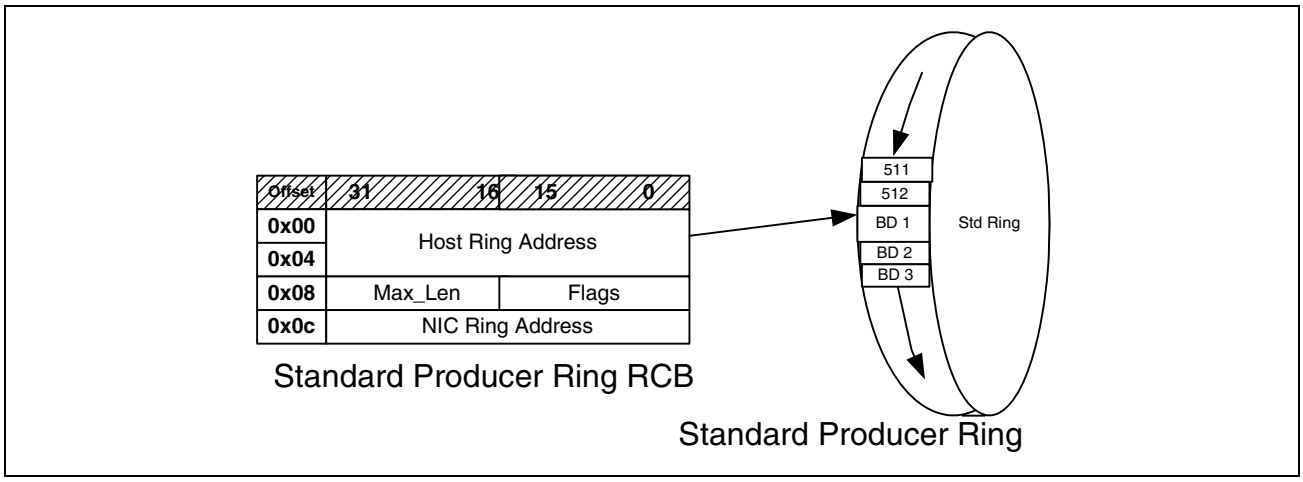
An example of setting up receive producer ring RCB:

```
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 15 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 MTU Size Register (offset: 0x43C)" on page 317). 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 317) specifies the largest packet accepted by the RX MAC; packets larger than the Receive MTU are marked oversized and are discarded.

Figure 15: Receive Producer Ring RCB Setup



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 on page 79. 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 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 83.

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 235) 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 20 on page 93) 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 20: 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 20: Mailbox Registers (Cont.)

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.

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 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. Figure 13 on page 78 shows receive return rings.

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 22). 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 21) 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 21: Receive Rules Configuration Register

Bits	*Field*	*Access*
31:8	Reserved	RO
7:3	Specifies the default class (ring) if no rules are matched	RW
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 22) and the Receive BD Rules Value/Mask register (see Table 24). 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 22: 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							

Table 23: Receive List Placement Rules Array (memory offset 0x480–0x4ff)

Bit	Name	RW	Description	Default
31	E	RW	Enable. Enabled if set to 1	–
30	&	RW	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	RW	If the rule matches, the processor is activated in the queue descriptor for the Receive List Placement state machine.	–
28	P2	RW	If the rule matches, the processor is activated in the queue descriptor for the Receive Data and Receive BD Initiator state machine.	–
27	P3	RW	If the rule matches, the processor is activated in the queue descriptor for the Receive Data Completion state machine.	–
26	M	RW	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	RW	Discard Frame if it matches the rule.	–
24	Map	RW	Map Use the masked value and map it to the class.	–
23:18	Reserved	RW	Must be set to zero.	0
17:16	Op	RW	Comparison Operator specifies how to determine the match: • 00 = Equal • 01 = Not Equal • 10 = Greater than • 11 = Less Than	–

Table 23: Receive List Placement Rules Array (memory offset 0x480–0x4ff) (Cont.)

Bit	Name	RW	Description	Default
15:13	Header	RW	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	RW	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	RW	Number of bytes offset specified by the header type.	–

Table 24: 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															

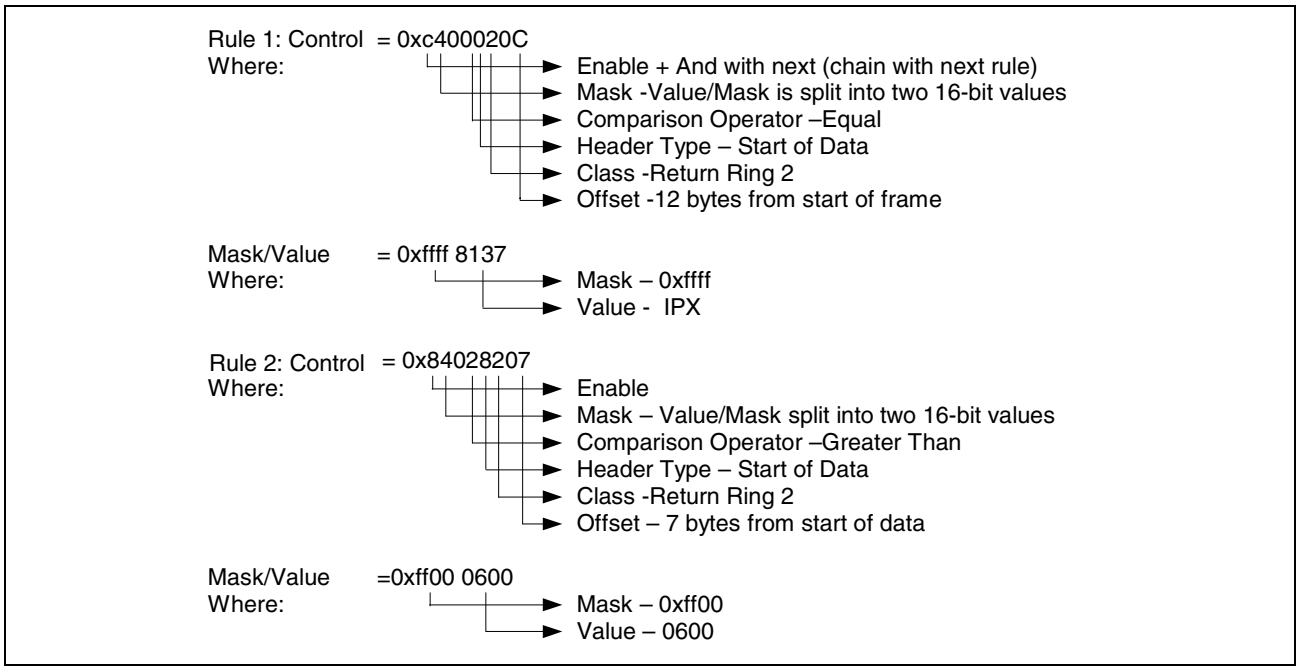
Bit	Name	RW	Description	Default
31:16	Mask	–	–	–
15:0	Value	–	–	–

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 16 on page 99.

Figure 16: Class of Service Example



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 472) 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 25 shows the frame format with IEEE 802.1Q VLAN tag inserted.

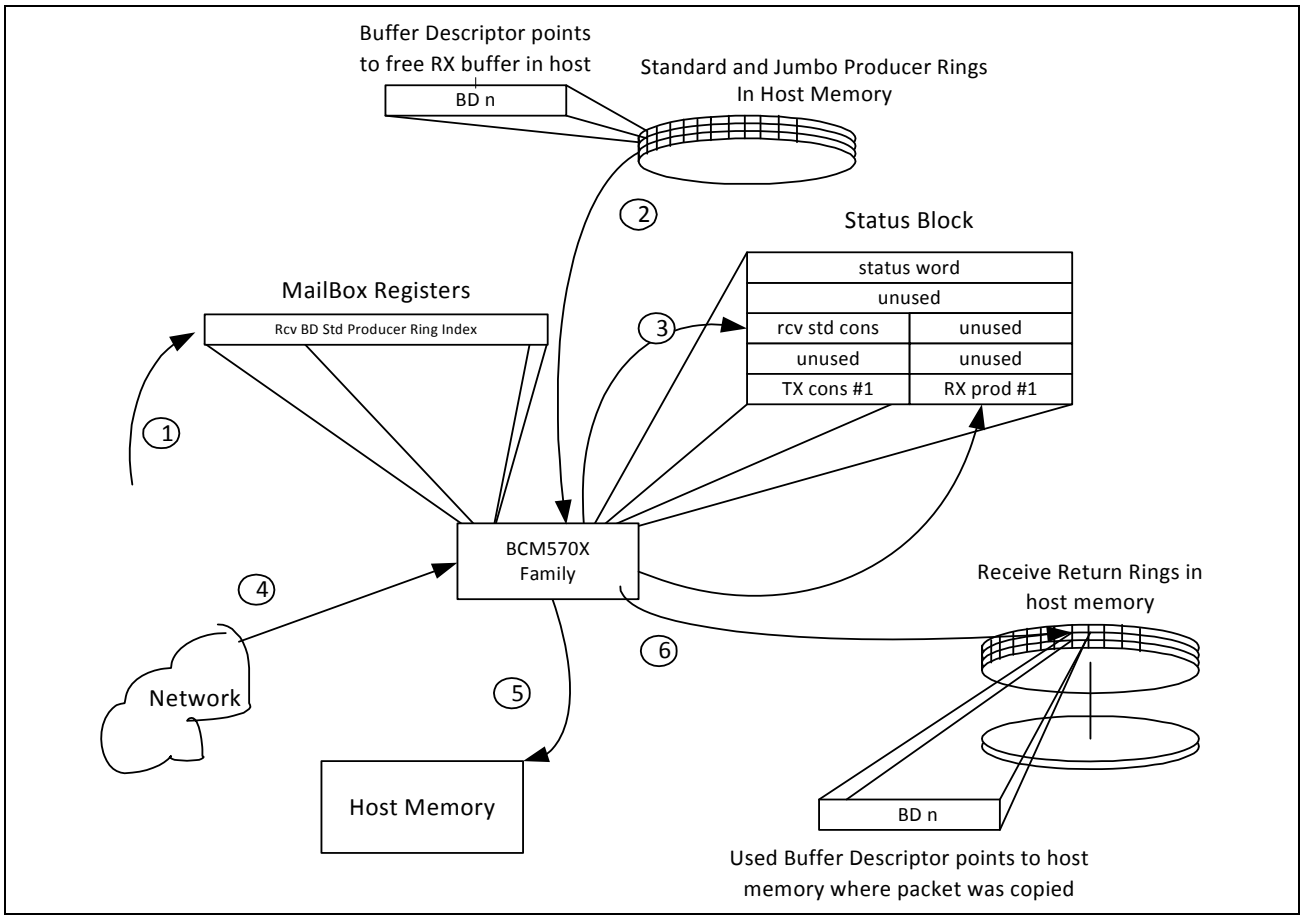
Table 25: Frame Format with 802.1Q VLAN Tag Inserted

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 17. 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 17: Overview Diagram of RX Flow



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

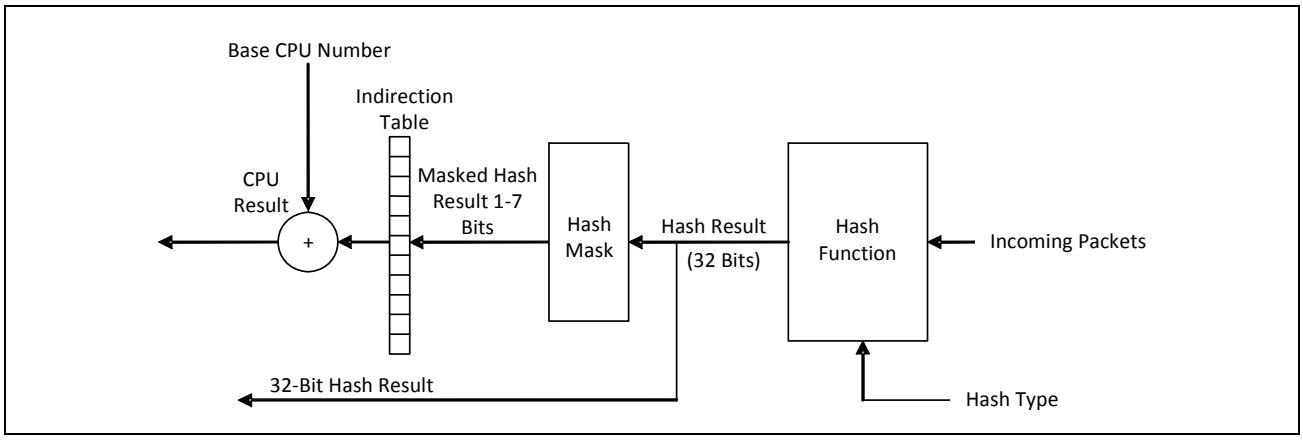
RSS is a scalable networking technology that enables receive packet processing to be balanced across multiple processors in the system while maintaining in-order delivery of the data. The RSS enables packets from a single network adapter to be processed in parallel on multiple CPUs/cores while preserving 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 18: 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.

Note: The BCM5719 and BCM5720 add support for UDP RSS hash functionality, in addition to the legacy TCP RSS hash functionality.

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 multiprocessor 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).

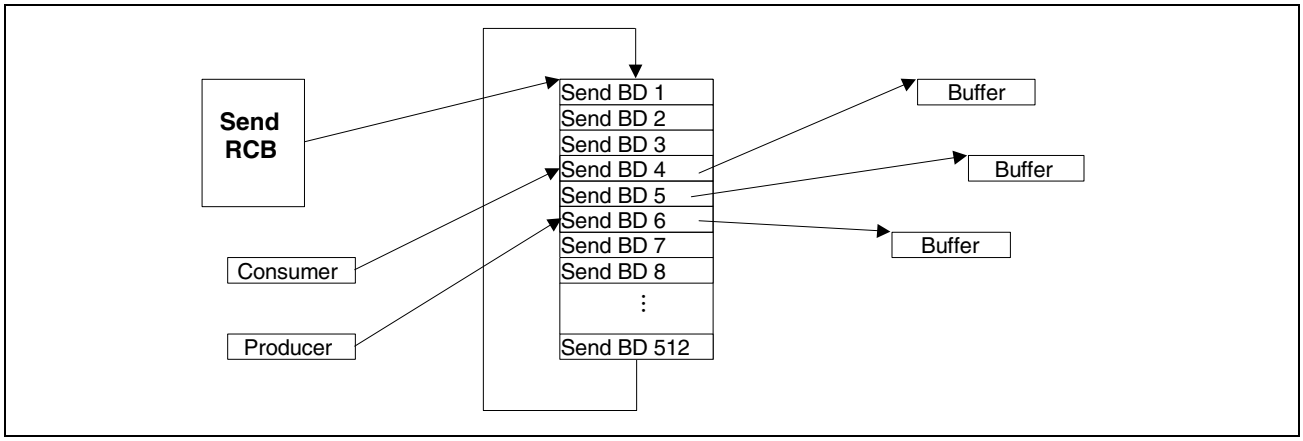
Associated with each ring are two indices that control its operation. These indices are the producer index and the consumer index, which are not shared between the host software and the Ethernet controller. In the case of send rings, the host software controls the producer index by adding elements (initializing 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 registers "Send BD Ring Host Producer Index Register (offset: 0x5900)" on page 465 through "Send BD Ring Host Producer Index Register (offset: 0x300–0x307)" on page 309. 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 83.

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 19 illustrates the relationships between all the components of a send ring.

Figure 19: Relationships Between All Components of a Send Ring

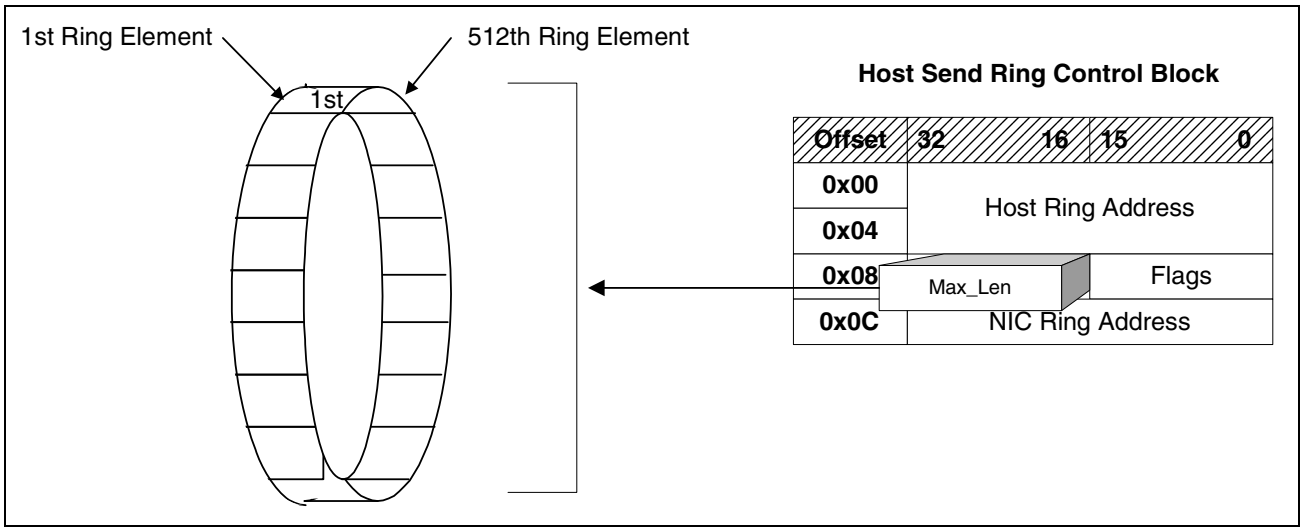


Ring Control Block

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 106 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 20) indicates the maximum number of BDs in the Send Ring. This field can be programmed to either 32, 64, 128, 256, 512, 1024, 2048, or 4096, depending on the type of ring for which the corresponding RCB applies.

Figure 20: Max_Len Field in Ring Control Block



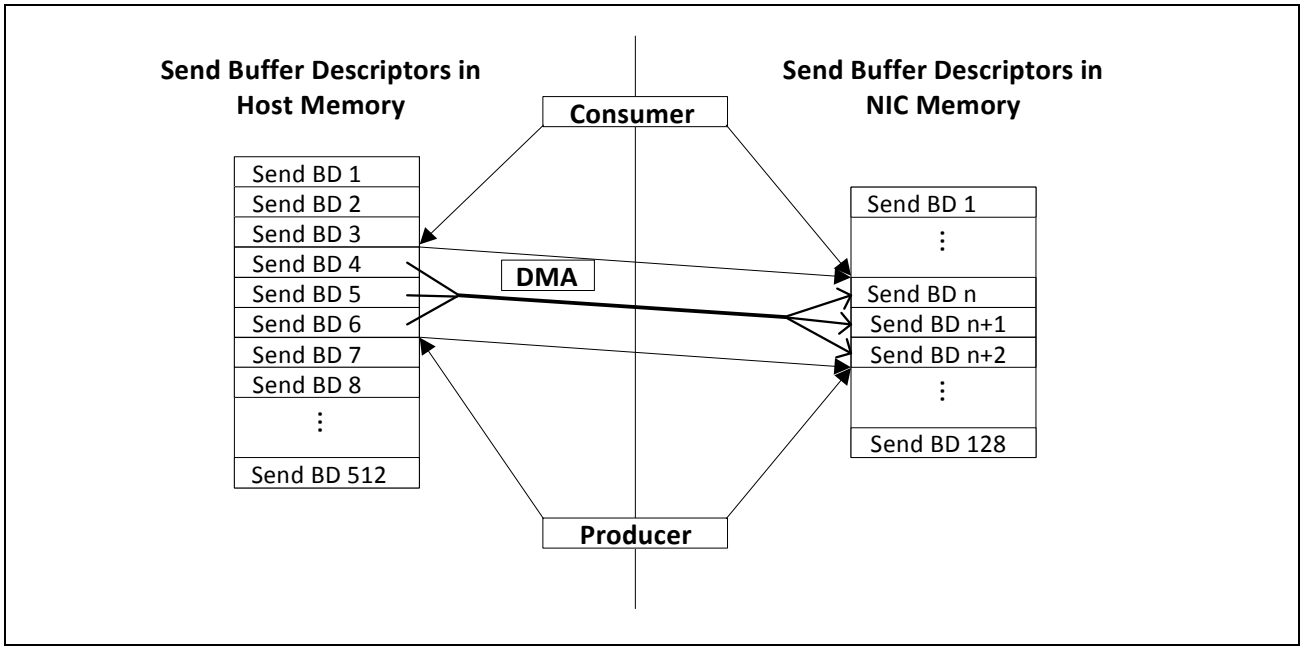
Host-Based Send Ring

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 21 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 (see "Send BD Ring Host Producer Index Register (offset: 0x300–0x307)" on page 309), 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 21: 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.

Note: In LSO enabled case, a driver needs to zero the TCP checksum field coming from upper layer of OS when doing TX CSO. Also, make sure to set bit-27 and bit-28 of 0x4800 when doing TX CSO in both LSO/Non LSO mode.

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 64 KB (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 BCM5718, 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.

Note: The BCM5718 family added to ability to use jumbo sized frames simultaneous with LSO.

QuickStart

Follow the steps to enable LSO:

1. Zero TCP checksum field in offloaded packet (leave IP checksum field alone)
2. Set register 0x0C00[3]=1: Enable hardware 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 26: 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 27: 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 28: 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 BCM5718.
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 29: 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 BCM5718.
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.

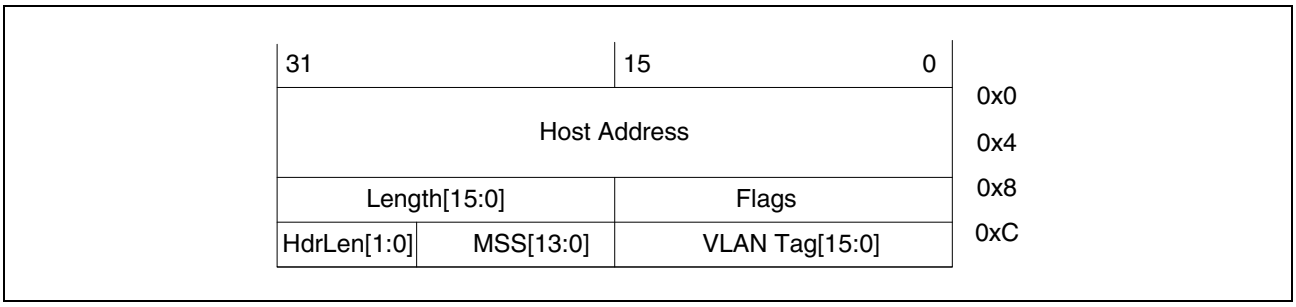
The ISO related registers are to allow for processing of ISO Ethernet Audio Video (EAV)-related TX traffic. A physically separate isochronous TX queue exists in some Broadcom Ethernet controllers to support audio/video traffic applications, which require very precisely timed isochronous launch of TX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

Note: LSO using jumbo frames is permissible on some Broadcom controllers (i.e., BCM5718). 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 22: 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 sub-field 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., BCM5718) to hold a value specifying a jumbo frame size.

Flags

See the table below.

Table 30: 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.

Table 30: Flag Field Description (Cont.)

Bit #	Flag Name	Flag Description
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	Capture Time Stamp (BCM5719/5720 only)	If this bit is 1, this frame's launch time shall be captured in the TX Time-Stamp Register.
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.
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 BCM5718 family supports jumbo Ethernet frames in both the receive (RX) and transmit (TX) paths and in conjunction with LSO. 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 BCM5718 family are described in this section.

Following are the feature highlights.

The maximum jumbo frame length supported by the BCM5718 family 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.

Other:

- Miscellaneous BD memory has been increased over legacy NetXtreme controllers from 6K (4K receive BD + 1K send BD + 1K gencomm) to 51 KB (17 KB standard receive BD + 17K jumbo receive BD + 16 KB 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.

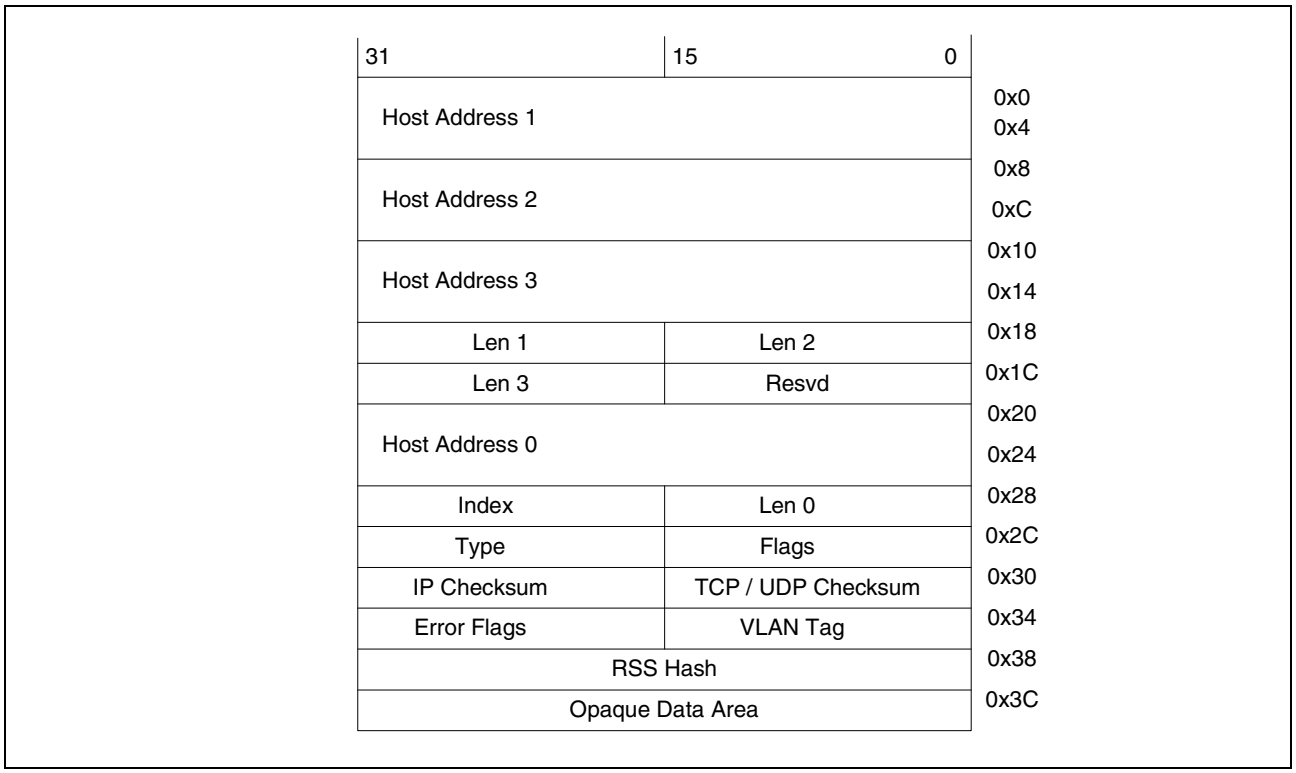
Affected Data Structures

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 23: "Extended RX Buffer Descriptor," on page 119. 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 23: Extended RX Buffer Descriptor



- 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. Len1, Len2, or Len 3 may be set to 0, but hardware ignores Len2 and Len3 when Len1 = 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 32 on page 121.
- For Receive Return Ring descriptors, if using IP checksum offload, the host driver software should rely on the Flags bit IP_CHECKSUM (Flags bit 12) to determine if the IP checksum in the received packet is correct. This field used to contain the actual IP checksum value, but that is not true for the BCM5718 Family of controllers. Only the Flags bit IP_CHECKSUM should be relied on by host driver software as is done by Broadcom drivers.
- 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 32 on page 121) is set in the Flags field.

Table 31: 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 32: 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 BCM5718 family 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 123).

There is also a jumbo ring replenishment threshold register 0x2C1C. The controller DMAs 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.

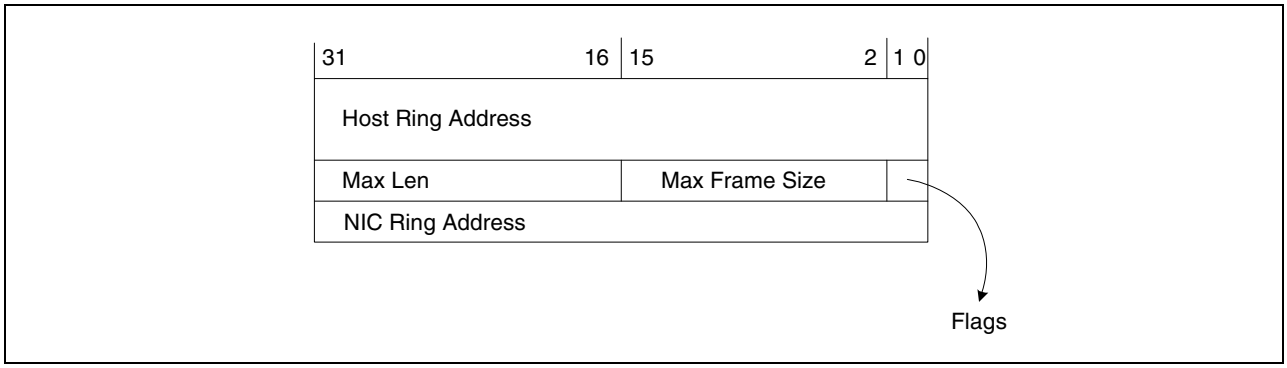
The maximum number of BDs that can be held in the jumbo ring is programmed in that ring's ring control block RCB.

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 BCM5718 family, 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 BCM5718 family 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 24: 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 BCM5718 family 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 BCM5718 family 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, 512, 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 33.

Table 33: Receive BD 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)

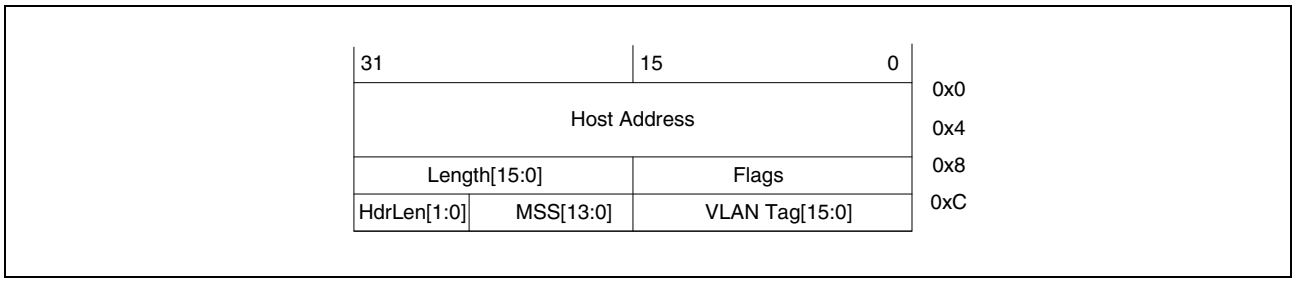
There are no structural changes to the return rings in the BCM5718 family. There are no functional changes from standard size frames or from a Receive Side Scaling (RSS) perspective. The impact of the jumbo frame feature to the return rings is clarified here:

- 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 24).
- 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 25 below illustrates the updated SBD format.

Figure 25: 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 34 below.

Table 34: 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	Capture Time Stamp (BCM5719/5720 only)	If this bit is 1, this frame's launch time shall be captured in the TX Time-Stamp Register.
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.
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.

Table 34: Send Buffer Descriptor Flags (Cont.)

Bit #	Flag Name	Flag Description
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 BCM5718 family 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 BCM5718 family's Legacy RSS mode status block. The updated structure of the status block is shown in Table 35 below.

Note that there are multiple formats of status blocks.

Table 35: Status Block

Offset	31 16	15 0
0x00	Status Word	
0x04	Reserved 0x0	Status Tag[7:0]
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

Table 35: Status Block (Cont.)

Offset	*31* *16*	*15* *0*
0x14	Reserved 0x0	Receive Jumbo Producer Ring Consumer Index

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

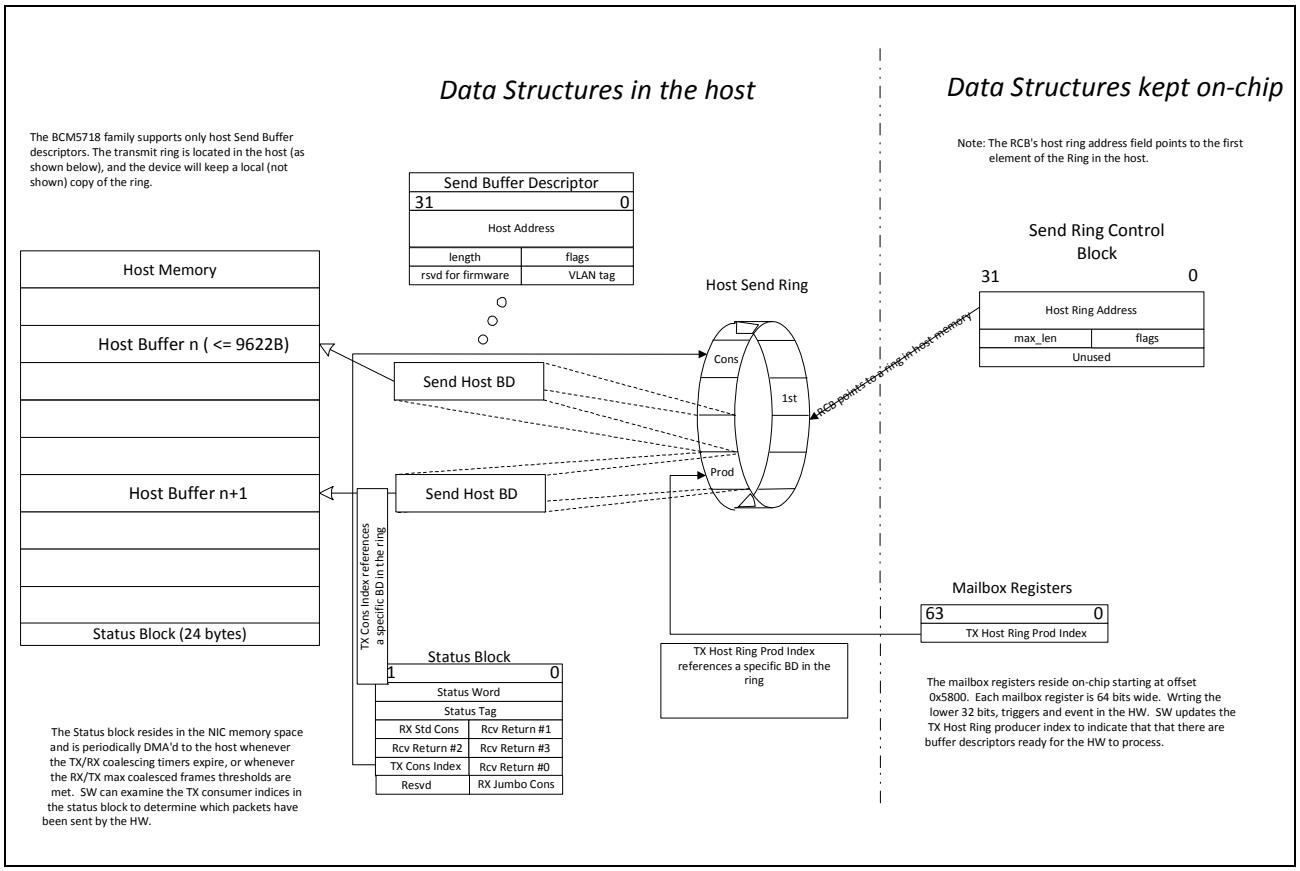
The driver essentially does not see any change in the send interface due to the jumbo frame feature. The only difference is that the stack is now allowed to construct larger frames (i.e., up to 9622 bytes long) out of send buffers.

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 26 below.

Figure 26: Send Driver Interface



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.

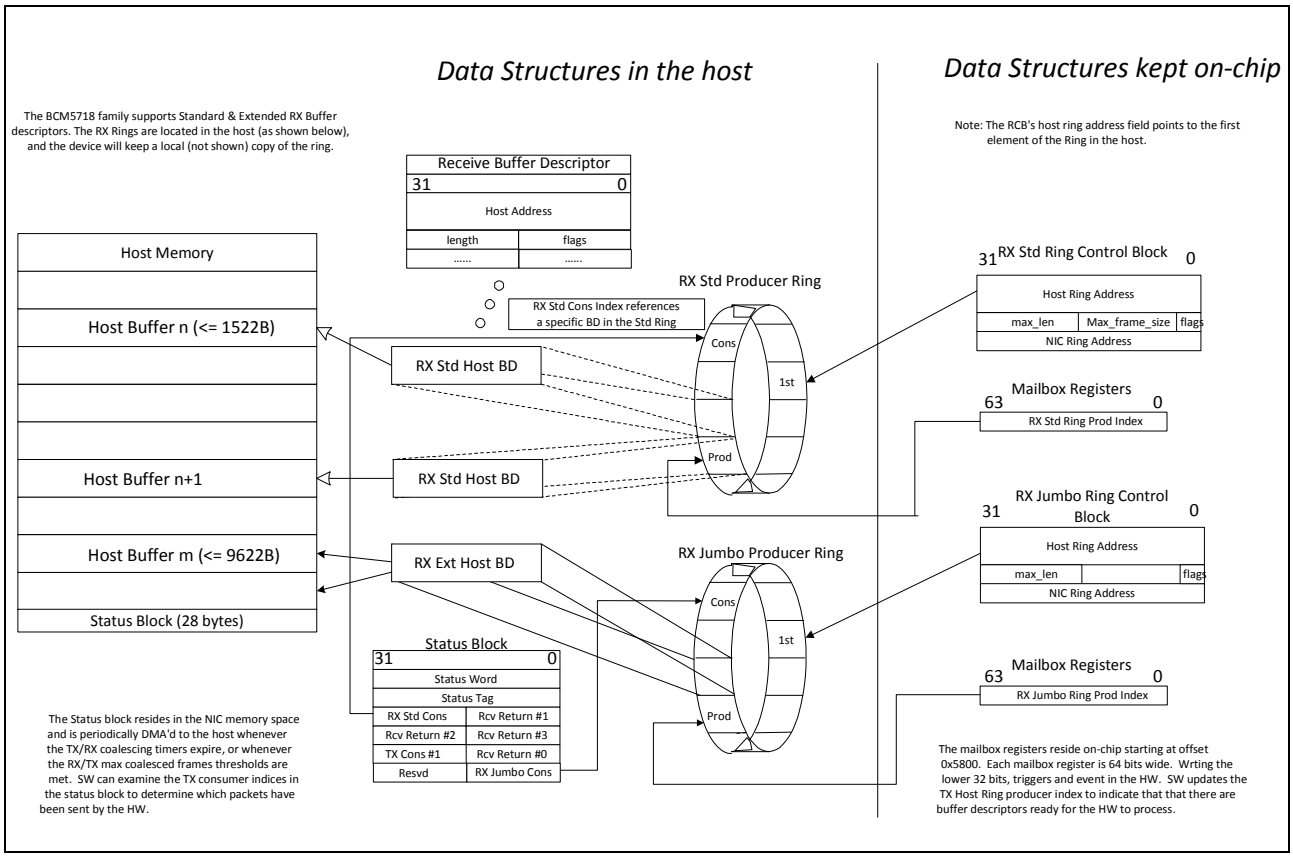
The controller fetches RX BDs in anticipation of RX frames and caches the BDs in the controller. The cached BDs are stored in the Misc BD Memory. There is a set of rules that govern placement of packets in buffers:

- 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 27 below.

Figure 27: Receive Producer Interface



The BCM5718 family 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 123).

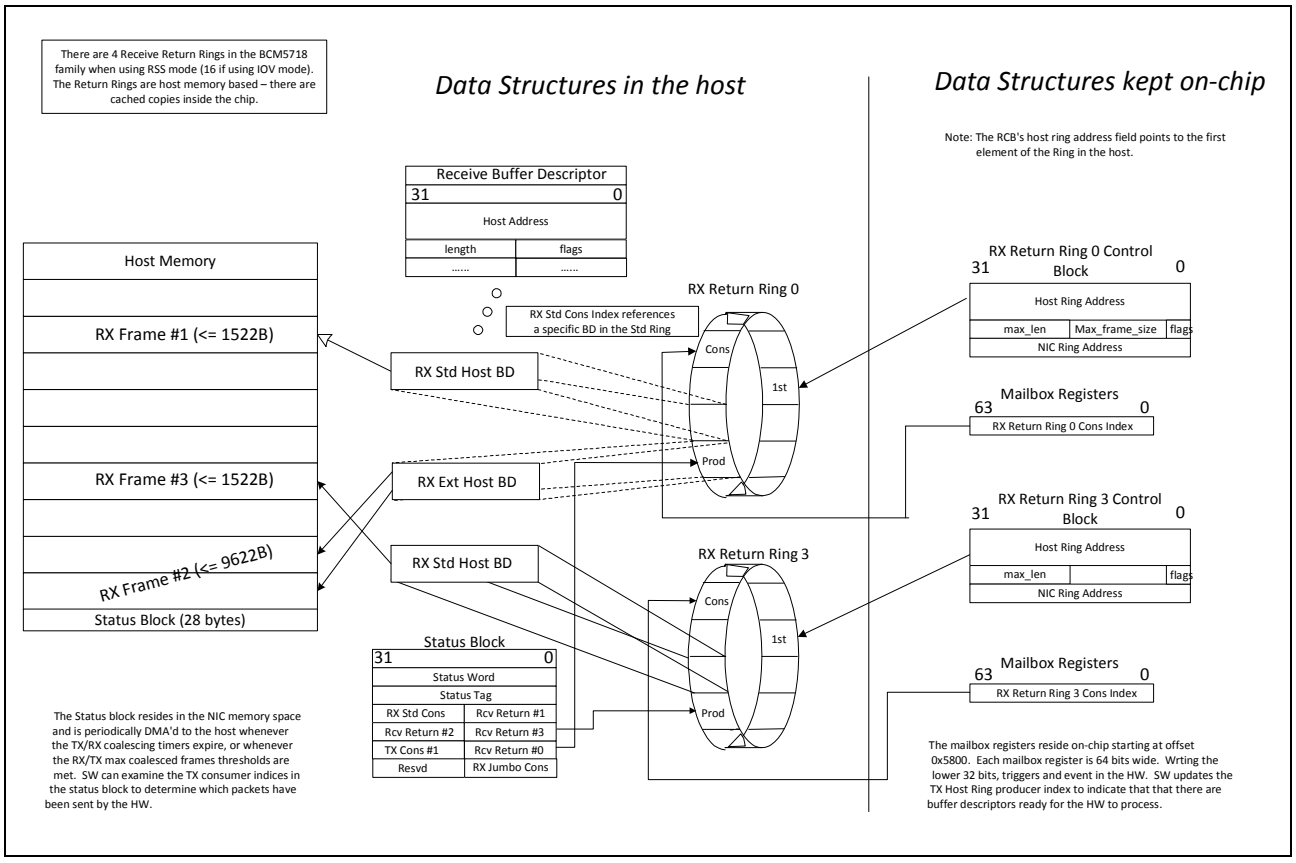
To support RSS, there are four return rings in the BCM5718 family. However, when RSS is disabled all RX frames are posted to Ring 0 while the other three rings remain inactive. In any case, standard and jumbo frames may be intermixed in any return ring as the order of placement strictly follows the order 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 28 below.

Figure 28: Receive Return Interface



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 123).

The legacy NetXtreme design has limitations in LSO hardware. In the case of the BCM5718 family, 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.

Summary of Register Settings to Support Jumbo Frames

- Standard receive producer ring 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)
[15:2]: std_max_packet_size = 0x5EE (1518 decimal)
[1:0] = 0
0x245C: Standard NIC address = 0x400000

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 receive producer ring 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 0x00044400

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.

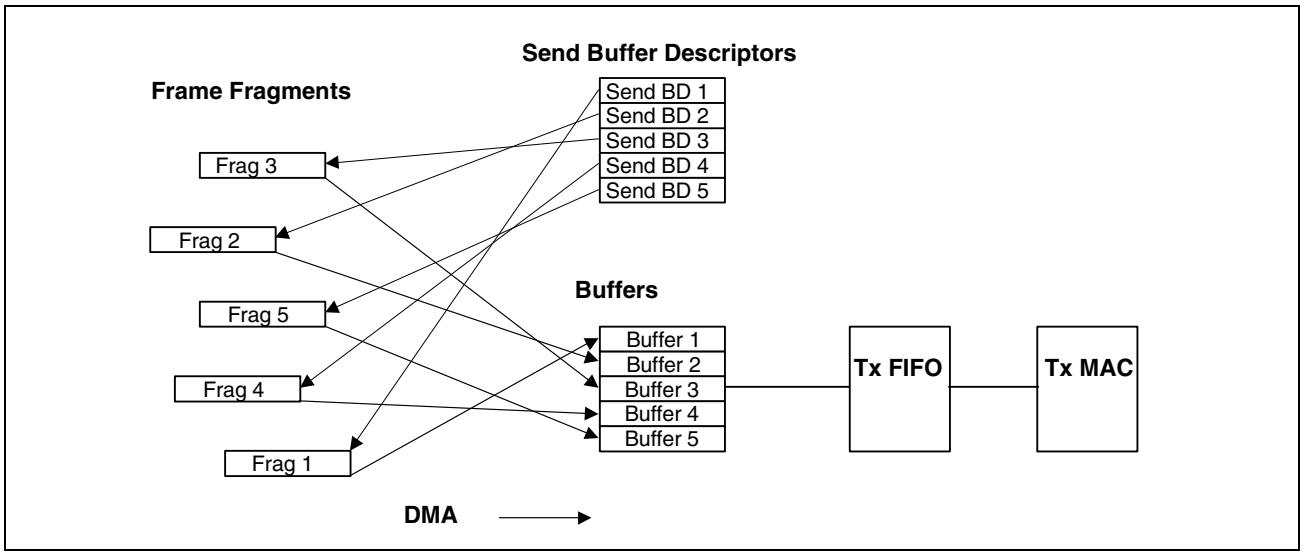
The ability to scatter/gather a frame lessens the restriction on the host software and increases overall system performance.

> ***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 29 illustrates the relationship between a frame consisting of multiple fragments and their corresponding send buffer descriptors.

Figure 29: 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 25 on page 100 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 106).

TX Data Flow Diagram

Figure 30 on page 133 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 83).
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 30 and Figure 31 on page 134 show the basic driver flow to send a packet.

Figure 30: Transmit Data Flow

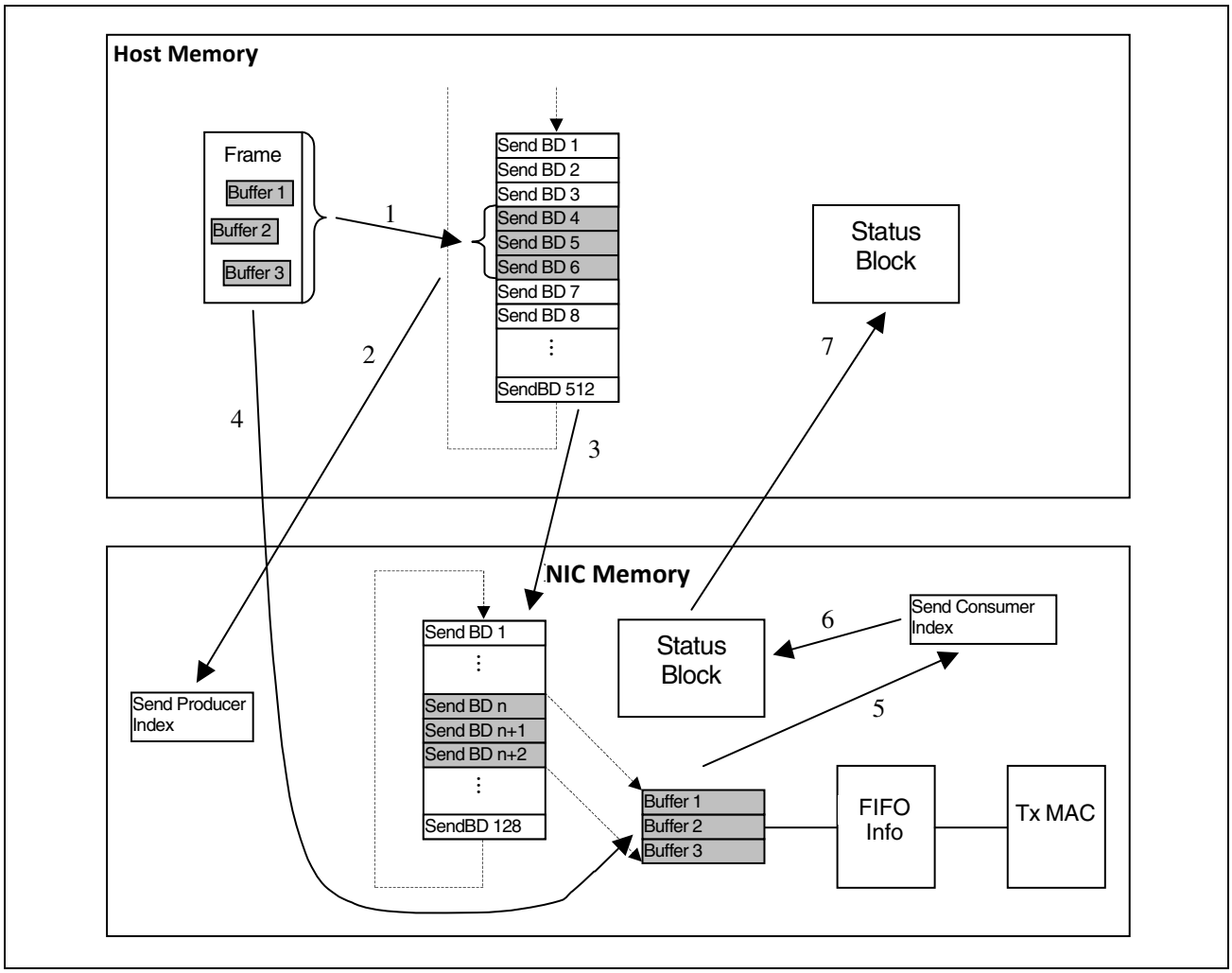
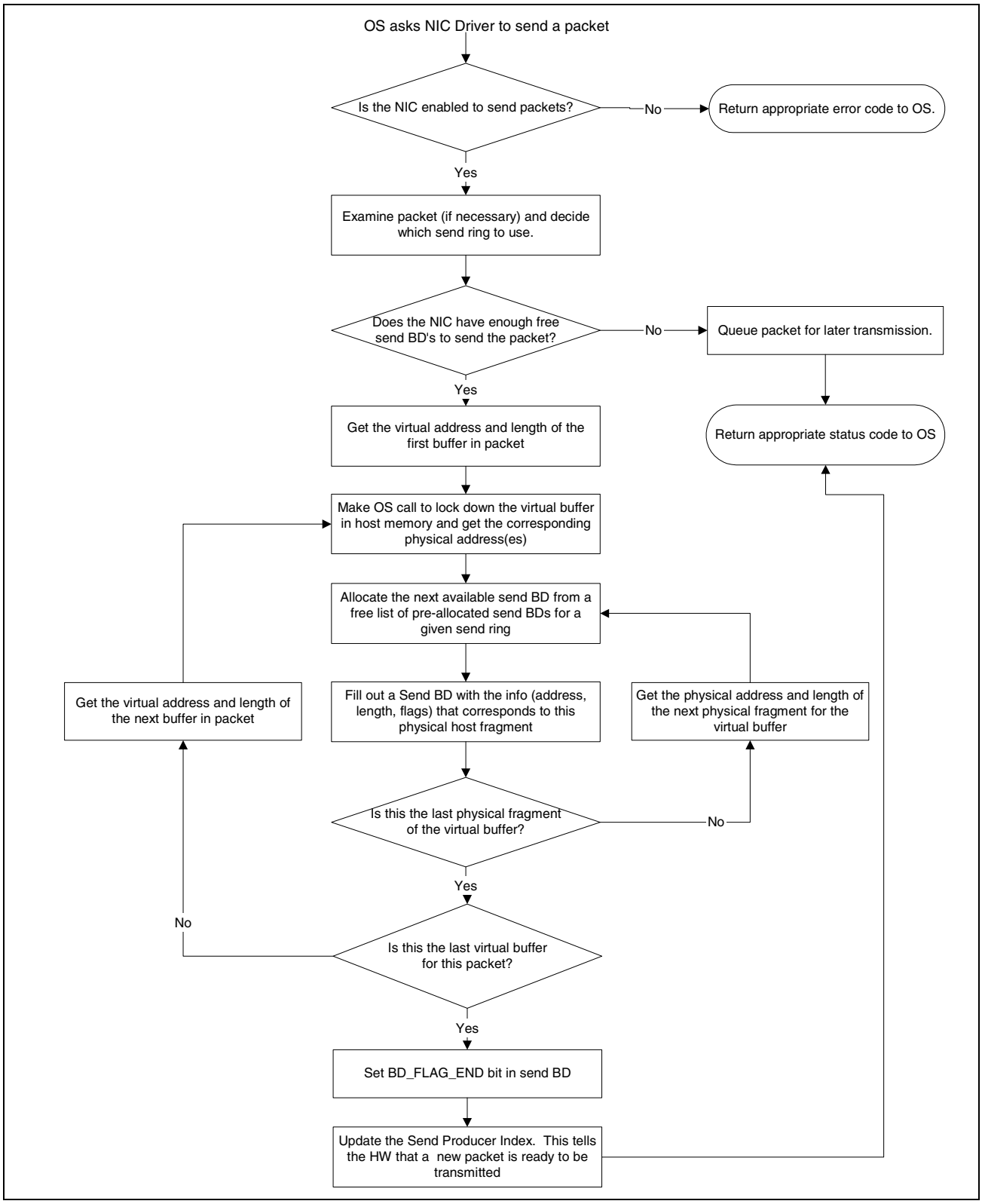


Figure 31: Basic Driver Flow to Send a Packet



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 474) 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 68).

- 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 Table 358 on page 334). 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 219).

The BCM5718 family of Ethernet controllers 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 36 illustrates the MAC register format.

The BCM5718 family 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 586).

Table 36: 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 37). 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 37: 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 Promiscous_Mode bit (bit 8) of the Receive_MAC_Mode register (offset 0x468). The Promiscous_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 136 and "MAC Address Setup/Configuration" on page 136), 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 96 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. Call the device reset procedure. (see "Device Reset Procedure" on page 147)
2. Enable/Disable any required bug fixes. Refer to the applicable Errata document for information on any errata that must be worked around by enabling/disabling the control bits of chip bug fixes if any are applicable.
3. Optionally, enable Tagged Status mode by setting the Enable_Tagged_Status_Mode bit of the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68)" on page 283).

Note: For additional information on Tagged Status mode see "Section 11: "Interrupt Processing," on page 230".

4. Clear the driver status block memory region by writing zeros to the host memory region where the status block will be direct memory accessed (DMA) (see "Status Block" on page 62).
5. Configure the DMA Write Water Mark in the DMA Read/Write Control register (see "DMA Read/Write Control Register (Offset: 0x6C)" on page 285), as follows.
   - 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).
   - if the Max Payload Size is 256 bytes or more, set the DMA write water mark bits (bits19-21) of DMA Read/Write Control register to 111b (for a water mark of 256 bytes).
6. Set 0x6c[1:0] = 00b to allow 64 byte cache alignment for DMA writes on 5719.
7. Optionally, set DMA byte swapping by setting 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 472. 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.
8. Configure the host-based send ring by setting the Host_Send_BDs bit in the General Mode Control register (see "Mode Control Register (offset: 0x6800)" on page 472).
9. Now that the 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 472).
10. 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)"). If the Send_No_PsuedoHeader_Checksum bit is set, the host software must seed the correct pseudo header checksum value in TCP/UDP checksum field. Similarly, if the Receive_No_PsuedoHeader_Checksum bit is set, the device driver must calculate the pseudo header checksum and add it to the TCP/UDP checksum field of the received packet.

11. Configure MAC Mbuf memory pool watermarks ("MAC RX MBUF Low Watermark Register (offset: 0x4414)" on page 432, and "Read DMA MBUF High Watermark Register (offset: 0x4418)" on page 432). Broadcom has run hardware simulations on the Mbuf usage and strongly recommends the settings shown in Table 38. These settings/values establish proper operation for 10/100/1000 speeds.

Table 38: 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 must 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 will have 80 bytes of packet data while each of the subsequent Mbufs [except the last Mbuf] will have 120 bytes for packet data. The last Mbuf in the chain will have the rest of the packet data which can be up to 120 bytes.	

12. Configure flow control behavior when the Rx Mbuf low watermark level has been reached (seeTable 39).

Table 39: Recommended BCM57XX Ethernet controller Low Watermark Maximum Receive Frames Settings

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

13. Enable the buffer manager by setting the Enable and Attn_Enable bits in the Buffer Manager Mode register (see "Buffer Manager Mode Register (offset: 0x4400)" on page 430). The buffer manager handles the internal allocation of memory resources for send and receive traffic.

14. 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 in order to replenish used receive descriptors. The recommended configuration value for the standard receive BD Ring replenish threshold is 0x19 (see "Standard Receive BD Producer Ring Replenish Threshold Register (offset: 0x2C18)" on page 375)

15. Initialize the standard receive Buffer Ring. Host software must write the Ring Control Block structure (see "Ring Control Block" on page 108) to the standard receive BD Ring RCB register (see "Standard Receive BD Ring RCB Registers" on page 368"). Host software must initialize the host physical memory address based on allocation routines specific to the OS.

16. Initialize the Max_len/Flags Receive Ring RCB register (0x2458).

Note: Beginning with the BCM57785 and BCM5718 families of controllers, field 15:2 no longer specifies the maximum expected size of a receive frame (it was formerly "Reserved").

Note: For the BCM5718 family, do not initialize the Receive Producer Ring NIC Address register (offset: 0x245C).

17. Initialize Receive BD Standard Producer Ring Index (0x26C) to zero.
18. Initialize the Standard Ring Replenish Watermark register (offset: 0x2d00). The recommended value is 0x20. If supporting Jumbo frames, also initialize 0x2d04 (the recommended value is 0x10). See "Jumbo Frames" on page 117 for more information about supporting Jumbo frames.
19. Initialize the Send Ring Producer Index registers by clearing (that is, set to zero) the Send BD Ring Host Producer index (see "Send BD Ring Host Producer Index Register (offset: 0x300–0x307)" on page 309).
20. Disable unused Receive Return Rings. Host software must write the RCB_FLAG_RING_DISABLED bit to the flags field of the ring control blocks of all unused Receive Return Rings.
21. Initialize Receive Return Rings. The Receive Return Ring RCBs are located in the Miscellaneous Memory region from 0x200 to 0x2FF. Host software must 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 or 1024, 2048, or 4096 depending on which ring's RCB is being initialized.
22. Initialize the Receive Producer Ring mailbox registers. The driver must 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 Register (offset: 0x5868)" on page 464).

Note: The host software must ensure 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 "LSO Read DMA Status Register (offset: 0x4804)" on page 437).

The Standard RX Producer threshold value should be set very low. Some operating systems may run short of memory resources and the number of BDs that are made available will decrease proportionally.

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

23. Configure the MAC unicast address. See "MAC Address Setup/Configuration" on page 136 for a full description of unicast MAC address initialization.
24. Configure random backoff seed for transmit. See the Ethernet Transmit Random Backoff register (see "Ethernet Transmit Random Backoff Register (offset: 0x438)" on page 317). 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

25. 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 317) must be configured before RX traffic is accepted. Host software must account for the following variables when calculating the MTU:
    - VLAN TAG
    - CRC
    - Jumbo Frames Enabled
26. Configure the Inter-Packet Gap (IPG) for transmit by setting the Transmit MAC Lengths register (see "Transmit MAC Lengths Register (offset: 0x464)" on page 323). The register contains three bit fields: IPG_CRS_Length, IPG_Length, and Slot_Time_Length. The value 0x2620 should be written into this register.

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

27. Configure the default RX return ring for non-matched 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 the RX packet is placed. The bit field is located in the Receive Rules Configuration register (see "Receive Rules Configuration Register (offset: 0x500)" on page 328).
28. . Configure the number of Receive Lists. The Receive List Placement Configuration register (see "Receive List Placement Configuration Register (offset: 0x2010)" on page 360) 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
29. 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 361).
30. Enable RX statistics by asserting the Statistics_Enable bit in the Receive List Placement Control register (see "Receive List Placement Statistics Control Register (offset: 0x2014)" on page 361).
31. Enable the Send Data Initiator mask by writing 0xFFFFFF (24 bits) to the Send Data Initiator Enable Mask register (see "Send Data Initiator Statistics Mask Register (offset: 0xC0C)" on page 348").
32. Enable TX statistics by asserting the Statistics_Enable and Faster_Statistics_Update bits in the Send Data Initiator Control register (0x0C08)
33. Disable the host coalescing engine by writing 0x0000 to the Host Coalescing Mode register (see "Host Coalescing Mode Register (offset: 0x3C00)" on page 415). Software needs to disable the host coalescing engine before configuring its parameters.
34. 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 415) and poll for 0x0000.
35. Configure the host coalescing tick count. The Receive Coalescing Ticks and Send Coalescing Ticks registers specify the number of clock ticks elapsed before an interrupt is driven (see "Receive Coalescing Ticks Register (offset: 0x3C08)" on page 416" and "Send Coalescing Ticks Register (offset: 0x3C0C)" on page 417). The clock begins ticking after RX/TX activity. Broadcom recommends the settings shown inTable 40.

Table 40: Recommended BCM57XX Ethernet Controller Host Coalescing Tick Counter Settings

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

36. Configure the host coalescing BD count. The Receive Max Coalesced BD and Send Max Coalesced BD registers specify the number of frames processed before an interrupt is driven (see "Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 418 and "Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 420). Broadcom recommends the settings shown in Table 41.

Table 41: Recommended BCM57XX Ethernet Controller Host Coalescing Frame Counter Settings

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

37. Configure the max-coalesced frames during interrupt counter. While host software processes interrupts, this value is used. Broadcom recommends the settings shown in Table 42.

Table 42: 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

38. Initialize host status block address. Host software must write a physical address to the Status Block Host Address register, which is the location where the MAC must DMA status data (see "Status Block Host Address Register (offset: 0x3C38)" on page 423). This register accepts a 64-bit value in register 0x3C38 (high order 32 bits) and 0x3C3C (low order 32 bits).

39. Enable the host coalescing engine (0x3C00 bit 1).

40. Enable the receive BD completion functional block by setting the Enable and Attn_Enable bits in the Receive BD Completion Mode register (see "Receive BD Completion Mode Register (offset: 0x3000)" on page 377").

41. Enable the receive list placement functional block by setting the Enable bit in the Receive List Placement Mode register (see "Receive List Placement Mode Register (offset: 0x2000)" on page 359.

42. Enable DMA engines by setting the Enable_FHDE, Enable_RDE, and Enable_TDE bits in the Ethernet Mac Mode register (see "EMAC Mode Register (offset: 0x400)" on page 310).

43. Enable and clear statistics by setting 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 310).

44. Delay 40 microseconds.

45. Configure the General Miscellaneous Local Control register (see "Miscellaneous Local Control Register (offset: 0x6808)" on page 475). 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.

46. Delay 100 microseconds.

47. Configure the Write DMA Mode register (see "Write DMA Mode Register (offset: 0x4C00)" on page 452). The following bits are 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

48. Set bit-29 of the Write DMA Mode register (see "Write DMA Mode Register (offset: 0x4C00)" on page 452) to enable the host coalescence block fix that configures the device to send out status block update before the interrupt message.

49. Delay 40 microseconds.

50. Set register 0x4900[2] = 1 to prevent DMA overruns for BD read DMA engine. This enables a hardware function which limits all BD fetches to 256 bytes or less.

51. Configure the Read DMA Mode register (see "LSO Read DMA Mode Register (offset: 0x4800)" on page 435). 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

52. Delay 40 microseconds.

53. Set 0x4800[24] = 0 to Allows multiple outstanding read requests from the non-LSO read DMA engine.

54. Set 0x4800[17:16] = 11b to Allows 4KB burst length reads for Jumbo/LSO network frames.

55. Set 0x4910[19:18] = 11b to allow 4KB burst length reads for non-LSO (i.e. standard) network frames.

56. Enable the receive data completion functional block by setting the Enable and Attn_Enable bits in the Receive Data Completion Mode register (see "Receive Data Completion Mode Register (offset: 0x2800)" on page 373).

57. Enable the send data completion functional block by setting the Enable bit in the Send Data Completion Mode register (see "Send Data Completion Mode Register (offset: 0x1000)" on page 352).

58. Enable the send BD completion functional block by setting the Enable and Attn_Enable bits in the Send BD Completion Mode register (see "Send BD Completion Control Registers" on page 358).

59. Enable the receive data and BD initiator functional block by setting 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 364).

60. 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 347).

61. Enable the send BD initiator functional block by setting the Enable and Attn_Enable bits in the Send BD Initiator Mode register (see "Send BD Initiator Mode Register (offset: 0x1800)" on page 355).

62. Enable the send BD selector functional block by setting the Enable and Attn_Enable bits in the Send BD Selector Mode register (see "Send BD Ring Selector Mode Register (offset: 0x1400)" on page 353).

63. Enable the transmit MAC by setting 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 320). Optionally, the software may set the Enable_Flow_Control to enable 802.3x flow control.

64. Delay 100 microseconds.

65. Enable the receive MAC. Set the Enable bit in the Receive MAC Mode register (see "Receive MAC Mode Register (offset: 0x468)" on page 323). Optionally, the software may set the following bits:

- Enable_Flow_Control bit–Enable 802.3x flow control
- Accept_oversized bit–Ignore RX MTU up to 64K maximum size
- Promiscuous_Mode bit–Accept all packets regardless of destination address
- No_CRC_Check bit–RX MAC will not check Ethernet CRC
- Various Hash Enable bits–if using RSS mode (Receive Side Scaling)

66. Delay 10 microseconds.

67. Setup the LED Control Register (0x40C). The Broadcom driver uses a value of 0x800 when initializing this register.

68. Activate link and enable MAC functional blocks by setting the Link_Status bit in the MI Status register (see "MII Status Register (offset: 0x450)" on page 319) to generate a link attention.

69. Optionally, disable auto-polling on the management interface (see "MII Mode Register (offset: 0x454)" on page 319).

70. Set Low Watermark Maximum Receive Frame register (offset: 0x504) to a value of 1 for the BCM5717 and BCM 5718 family of controllers.

71. Configure D0 power state in PMSCR (see "Power Management" on page 188). Note that 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.

72. Setup the physical layer and restart auto-negotiation. For details on PHY auto-negotiation, refer to the applicable PHY data sheet.

73. Setup multicast filters. See "Packet Filtering" on page 136 for details on multicast filter setup.

74. Enable the Host Interrupt:

a. Set the Clear_Interrupt bit in the Miscellaneous Host Control register (offset: 0x68) to clear the interrupt and Clear the Mask_Interrupt bit in the Miscellaneous Host Control register (offset: 0x68) to unmask the interrupt.

b. Set the interrupt mail box register (offset: 0x200) to 0.

Device Reset Procedure

1. Write the T3_MAGIC_NUMBER (0x4B657654 = KevT) to the device memory at offset 0xB50 to notify the bootcode that the reset is a warm reset (driver initiated core_clocks reset).

2. Save PCI command register 0x4 before chip reset (GRC_MISC_CFG core clock reset will clear the memory enable bit in PCI register 0x4, so we save relevant registers here)

3. Clear the Fast Boot Program Counter register (Offset 0x6894) and enable the Memory Arbiter (see "Memory Arbiter Mode Register (offset: 0x4000)" on page 428).

4. Initialize the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68)" on page 283).

a. Set the Enable_Endian_Word_Swap bit in the Miscellaneous Host Control register, when the host processor architecture is little-endian. Set the Enable_Endian_Word_Swap bit and the Enable_Endian_Byte_Swap bit in the Miscellaneous Host Control register, when the host processor architecture is big-endian.

b. Enable the indirect register pairs by setting the Enable_Indirect_Access bit in the Miscellaneous Host Control register (see "Indirect Mode" on page 172).

c. 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.

5. Reset the core clocks by setting the CORE_Clock_Blocks-Reset bit in the General Control Miscellaneous Configuration register and setting the Disable_GRC_Reset_on_PCI-E_Block bit (bit-29) to 1 ("Miscellaneous Configuration Register (offset: 0x6804)" on page 474).

6. Wait for the core-clock reset to complete. The core clock reset disables indirect mode and flat/standard modes ¡X software cannot poll the core-clock reset bit to deassert, since the local memory interface is disabled by the reset. Delay a minimum of 1 millisecond before continuing the initialization sequence.

7. Enable MAC memory space decode and bus mastering by setting the Bus_Master and Memory_Space bits in the PCI Configuration Space Command register (see "Status and Command Register (offset: 0x04)" on page 272).

8. Enable the MAC memory arbiter by setting the Enable bit in the Memory Arbiter Mode register (see "Memory Arbiter Mode Register (offset: 0x4000)" on page 428).

9. Initialize the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68)" on page 283).

a. Set the Enable_Endian_Word_Swap bit in the Miscellaneous Host Control register, when the host processor architecture is little-endian. Set the Enable_Endian_Word_Swap bit and the Enable_Endian_Byte_Swap bit in the Miscellaneous Host Control register, when the host processor architecture is big-endian.

b. Enable the indirect register pairs by setting the Enable_Indirect_Access bit in the Miscellaneous Host Control register (see "Indirect Mode" on page 172).

c. 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.

10. Set Word_Swap_Data, Byte_Swap_Data, and win the General Mode Control register when the host processor architecture is little endian. Optionally, set Byte_Swap_Non_Frame_Data in the General Mode Control register when the host processor architecture is big endian, (see "Mode Control Register (offset: 0x6800)" on page 472).

11. Poll for bootcode completion. The device driver must poll the general communication memory at 0xB50 for the one is complement of the T3_MAGIC_NUMBER (that is, 0xB49A89AB). The bootcode should complete initialization within 1000 ms for Flash devices and 10000 ms for SEEPROM devices.

Device Closing Procedure

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

1. Disable Host Interrupt.

a. Set the Mask_Interrupt bit in the Miscellaneous Host Control register (offset: 0x68) to disable interrupt

b. Set the interrupt mail box register (offset: 0x200)

2. Tell Firmware the driver are shutting down and doing prereset.

Note: Do this only if ASF enabled.

a. Write the 0x2 to the device memory (offset: 0xB78) to PAUSE the firmware

b. Set the RX-CPU Event register (offset: 0x6810) bit14 (SW Event 7) 7 bit and wait for SW bit14 (SW Event 7) to be clear

c. Write MAGIC Number 0x4B657654 to the device memory (offset: 0xB50)

d. Write 0x2 to the device memory (offset 0xc04)

3. Disable all the receiver blocks.

a. Clear the Enable bit in the Receive MAC Mode register (offset: 0x468)

b. Clear the Enable bit in the Receive BD Initiator Mode register (offset: 0x2C00)

c. Clear the Enable bit in the Receive List Placement Mode register (offset: 0x2000)

d. Clear the Enable bit in the Receive BD Initiator Mode register (offset: 0x2400)

e. Clear the Enable bit in the Receive Data Completion Mode Register (offset: 0x2800)

f. Clear the Enable bit in the Receive BD Completion Mode Register (offset: 0x3000)

4. Disable all the transmit blocks.

a. Clear the Enable bit in the Send BD Selector Mode register (offset: 0x1400)

b. Clear the Enable bit in the Send BD Initiator Mode Register (offset: 0x1800)

c. Clear the Enable bit in the Send Data Initiator Mode Register (offset: 0xC00)

d. Clear the Enable bit in the Read DMA Mode Register (offset: 0x4800)
e. Clear the Enable bit in the Send Data Completion Mode Register (offset: 0x1000)
f. Clear the Enable bit in the Send BD Initiator Mode Register (offset: 0x1C00)

5. Shut down all of the memory managers and related state machines.
   a. Clear the Enable bit in the Host Coalescing Mode Register (offset: 0x3C00)
   b. Clear the Enable bit in the Write DMA Mode Register (offset: 0x4C00)
   c. Reset the FTQ by setting 0xffffffff to the FTQ Reset Register (offset: 0x5C00)
   d. Set FTQ Reset Register 0x5C00 to 0
6. Reset the controller. Call Device Reset Procedure (see "Device Reset Procedure" on page 147)
7. Instruct the firmware do the following post reset:

Note: Do this only if ASF enabled.

   a. write 0x2 to the device memory (offset 0xc04)
   b. write 0x80000002 to the device memory (offset 0xc04)
8. Free the RX/TX Ring list and buffers.
9. Set PCI power state to D3hot.

Energy Efficient Ethernet™

The BCM5718 family of controllers 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 BCM5718 family of controllers.

```
/*
 * 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: IEEE1588

IEEE1588 Time Sync Introduction

IEEE1588 and IEEE802.1AS are two protocols designed to perform time synchronization over Ethernet networks. IEEE1588 Precision Time Protocol (PTP) provides a UDP packet based time synchronization mechanism, while IEEE802.1AS extends PTP to operate directly over the L2 or Ethernet layer. Time precision offered by PTP is in the sub-microsecond range and, in fact, can get down to a Nanosecond.

NetXtreme Time Sync Assist

NetXtreme architecture provides a set of hardware (HW) features to assist IEEE1588 and IEEE802.1AS traffic over Ethernet. Both 1588 and 802.1AS are complex protocols from the perspective of statefulness and are best implemented by a software module resident in the host computer. However, since these protocols demand a sub-microsecond, in some cases nanosecond grade precision, long latencies encountered by host software do not permit these protocols to be implemented in software (Host CPU) in their entirety. The goal of NetXtreme architecture is to provide the cheapest and simplest set of HW hooks so that a service grade PTP profile may be enabled in the host server computer.

The assists that NetXtreme provides are:

- A 64-bit Counter clocked by the 125Mhz DLL clock to serve as the Precision Clock - Thus 8ns is the maximum precision offered. This clock is known as the EAV Reference Clock.
- A 64-bit Transmit Time Stamp Register - Software shall mark certain types of PTP message Packets in turn for TX hardware to capture launch time of that packet.
- A 64-bit Receive Time Stamp Register - RX hardware shall crack and identify certain types of PTP packets as configured by host SW. The reception time of such a packet shall be recorded in this register. Host SW shall retain the control to choose which PTP message Types get time stamped.

This feature, Time Sync Assist, may be enabled by setting a Mode bit in the chip.

Coexistence

The Time Sync Assist feature coexists with the following NetXtreme features:

- VLAN Tagging - in-band or out-of-band
- SNAP/LLC Framing - transmit and receive.
- Transmit LSO - note that PTP packets cannot be TCP segments.
- Transmit IP, UDP, TCP checksum offload - IP/UDP Transmit Checksum offload is applicable to UDP PTP packets, provided the packets are appropriately formatted by the software protocol engine, that is, 2B UDP padding etc must be taken care by software.

PTP Link Delay Measurement

At the completion of the Delay Request/Response exchange, the delay requester uses four timestamps (t1, t2, t3, t4) to compute the link delay. The link delay is computed as the average of the two one-way delays using the following formula:

T delay = [(t2 - t1) + (t4 - t3)] / 4

Assume that both nodes contain a NetXtreme Time-Synch Capable NIC. Table 43 describes the roles the PTP software component and the Time-Synch capable NIC hardware play during the above exchange.

Table 43: PTP Link Delay Measure Roles

	Delay Requester		***Delay Responder***	
	Host Software	***NIC Hardware***	***NIC Hardware***	***Host Software***
1	TX PTP Delay Request packet- mark capture →	Capture TX stamp (t1) →		
2			→ Receive and identify PTP delay request packet, then capture RX Time Stamp (t2)	→ Receive packet, Read RX Time Stamp Reg.
3a			Capture TX Time Stamp (t3) ←	TX PTP Delay Response packet with embedded (t2) value — mark capture ←
4	← Receive Packet, Read RX Time Stamp Reg	Receive and identify PTP Delay Response packet, then capture RX Time Stamp (t4) ←		
3b	Receive packet		← (pass through)	TX PTP Delay Response Follow-up packet with embedded (t3) value
5	Collect t1, t2, t3, and t4 to compute T delay.			

PTP Time Synchronization Messaging

The slave device then uses the link delay (Tdelay) and the Sync Message timestamps (t1, t2) to calculate the time offset it needs to compensate its local clock by using the following equation:

Tslave-offset = t2 - t1 - Tdelay

Assume that both nodes contain a NetXtreme Time-Synch Capable NIC. Table 44 describes the roles the PTP software component and Time-Synch capable NIC hardware play during above exchange.

Table 44: PTP Time Synchronization Messaging Roles

	PTP Master Node		*PTP Slave Node*	
	Host Software	***NIC Hardware***	***NIC Hardware***	***Host Software***
1	TX PTP Sync packet—mark capture →	Capture TX stamp (t1) →		
2			→ Receive and identify PTP sync packet, then capture RX Time Stamp (t2)	→ Receive packet, Read RX Time Stamp Reg.
3	TX PTP Sync Follow-up packet with embedded (t1) value →		→ (pass through)	→ Receive PTP Follow-up packet
4				Collect t1, t2, and Tdelay to compute the Slave Offset value

Hardware Description

The following items describe the included hardware:

Clock Hardware:

- EAV Reference Clock - this clock is neither the Master-clock nor the Slave-clock, but a specific clock. All time related hardware services are provided with respect to this clock.
- Reference Time Capture off an external GPIO Trigger
- A Time Watchdog which drives a GPIO output pin upon meeting a programmable time value
- A divided EAV Reference Clock output

Transmit Time Sync hardware:

- A 64-bit TX Time Stamp Register.
- A bit defined in Send BD for indication of TX Time capture

Receive Time Sync hardware:

- Programmable Receive Frame Cracker.
- A 64-bit RX Time Stamp Register + A 16-bit RX PTP Sequence ID Register
- An RX Time Stamp Lock Timer

Each of above items is described separately in the following sections.

EAV Reference Clock/Counter

The EAV Reference Clock (Count) is a 64-bit free-running clock-tick counter with a 1ns time base. All Real-Time sensitive services needed by EAV are in reference to this counter. In the 5719/5720 chip, this counter is clocked by a 125Mhz free running DLL. Therefore, the precision of this counter is limited to 8ns for this generation of NIC. This implies that the LSB 3-bits of EAV Reference Count shall always be "000" in this chip. This counter will emerge with a 0 value from Power on Reset and always count upwards. After reaching the full count, the counter will roll-over to 0 and count up again. These two counters are the only static attributes of this clock. The rest are programmable.

Time referencing, Time-stamping, Time-sampling etc. all real-time interfacing of the chip hardware with the AV Device Driver shall be performed in reference to this counter.

The following are the instrumentations available for the EAV Reference Counter:

1. Reset Control: The Driver has the option to configure the EAV Reference Counter to be reset in response to the following events:
   a. GRC Reset (MAC-core soft reset) and PCIE FLR (if supported)
   b. PCIe Link Reset
   c. Network Link transition from UP→DOWN
   d. Network Link transition from DOWN→UP
2. Explicit Driver Controls:
   a. Driver may reset the counter.
   b. Driver may stop the counter.
   c. Driver may re-start the counter after being stopped.
   d. Driver may over-write the counter value by PIO write.
3. The driver may read the value of the free-running counter anytime. For that purpose, a 32-bit Register pair is provided.
4. The driver may take a snap-shot of this counter.

EAV Reference Corrector

The 64-bit of EAV Ref Count described in the previous section is the integer porting of the counter. This is the part visible to the system for time stamping purposes. There is also a 24-bit correction field associated with the EAV Reference Count. This field is used by the host system to periodically add or subtract a minimum resolution value from the EAV Reference Count, in case of 5719/5720. That value happens to be 8 nanoseconds. This feature is mainly used to compensate for the local crystal/PLL drifts. The add/subtract operation is performed entirely in hardware. Software is responsible only for programming the EAV Reference Corrector field appropriately.

The Corrector features a Correction-Value and a Correction-Sense sub-field. Host software (in this case the software PTP module) must compute the correction amount and then load the Correction-Value/Sense fields accordingly. The programmed value could be anything from 0x000001 through 0xFFFFFF. A value 0x0 is NOT permitted. The Correction-Value is added to a 24-bit accumulator every EAV Reference Clock tick (accumulator is re-initiated to 0 at POR reset and every time the Corrector register is updated). Therefore, the accumulator's value will increase with every EAV Ref clock cycle until it overflows. In the EAV Reference Clock tick in which the accumulator overflows, the hardware reloads the accumulator with the last programmed Correction-Value. On the same clock cycle, based on the programmed Correction-Sense, the hardware either adds or subtracts 8 ns from EAV Reference Count. Therefore, the host software controls the sense, value, and period of an automatic hardware based correction. Although software may re-program the EAV Reference Corrector frequently, be cautious that a large negative correction-value may result in out of order time-stamps.

Time Watchdogs

A Time Watchdog is a 64-bit register which may be programmed by host software to a specific time value in the future. When the value of the EAV Reference Count equates the value of the Time Watchdog, the1588_GPIO[n] output pin is toggled. There are two such Time Watchdogs which work independently.

Divided EAV Reference Clock Output

An additional PLL channel output has been routed to an external pin. This output clock is edge synchronous to the EAV Reference clock, although there could be a < 5ns routing delay present. A register is available in the CPMU block in which the clock-divisor value may be programmed by host software. The available output frequency range is 125MHz through 4.8 MHz. See the Flash Clock Policy Register (0x366C).

Transmit Time Stamping Service

The software is able to mark certain types of PTP packets (Delay Request, Synch etc) and the NIC hardware captures the time when such a packet's first byte, or SOF, is launched for transmission. Ideally, the time capture is the instant of the actual SOF launch on the wire, but due to practical limitations, the capture will take place at the time the SOF appears on GMII interface among EMAC and GPHY. The propagation delay through the GPHY is virtually fixed, and hence, could easily be accounted for by system software.The captured time is in reference to the EAV Reference Count. There is only a single 64-bit register, the TX Time Stamp Register, for this purpose. The precision of this stamp is 1ns. System software may read the value of this register anytime to get the time-stamp of last sent PTP packet. Reference the following table of send ring SBD for IEEE1588.

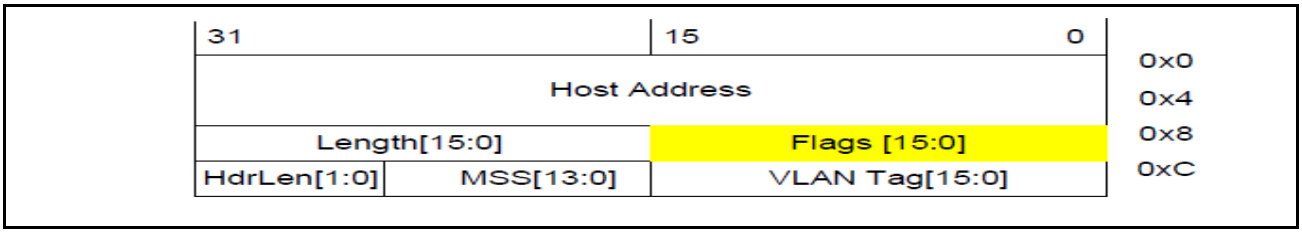


Table 45: Send Ring SBD Flags

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 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 be set 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 Ether+IP+TCP Headers to be replicated in each segment arising out of a Large TCP Segment (LSO).
5	Capture Time Stamp	If this bit is 1, this frame's launch time shall be captured in the TX Time-Stamp Register.
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.

Receive Time Stamp and Sequence ID Registers

The time of reception of chosen PTP frames shall be recorded by hardware in the RX Time Stamp Register. The place of time capture is the GMII interface, the point of capture is arrival of the SOF octet and the reference of time is the EAV Reference Count. Thus, the time of appearance of the SOF octet of every RX frame on the internal GMII interface needs to be recorded by hardware and held it in a temporary register or a temporary FIFO. After reception of a complete frame, upon examining the RX Frame cracker results, hardware shall decide if that particular Frame had the Host SW desirable PTP message type and only then it will transfer its acquired time stamp into the RX Time Stamp Register. Along with the RX Time Stamp, hardware shall also capture the respective PTP packets sequenceID field. Hardware shall also mark a status field in a packet's Receive BD indicating if the packet was a PTP packet and also if RX Time Stamp was captured for the packet.

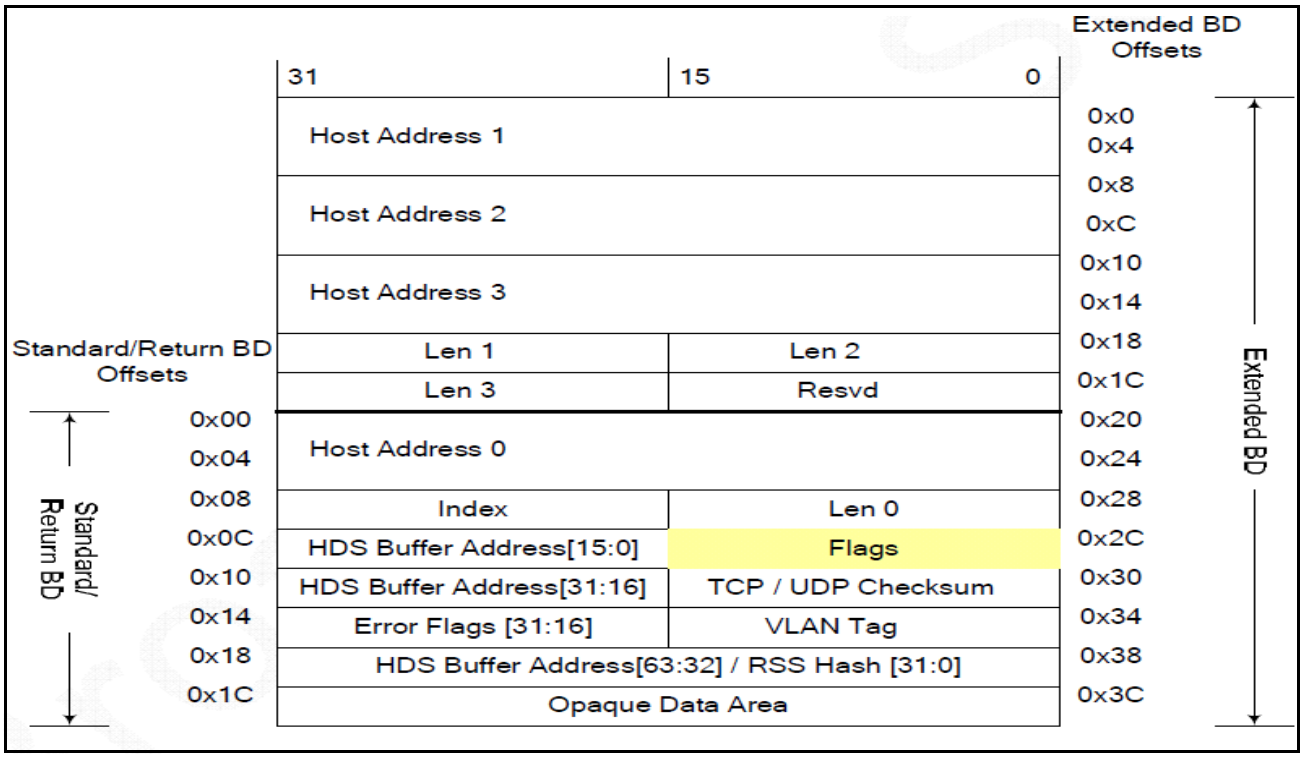


Table 46: Receive Return Ring RBD Flags

Bit #	*Flag Name*	*Flag Description*
1:0	PTP Packet Type	00: PTP Delay Request Packet 01: PTP Delay Response Packet 10: PTP Sync Packet 11: All other type of PTP Packets
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.

Table 46: Receive Return Ring RBD Flags (Cont.)

Bit #	*Flag Name*	*Flag Description*
4	PTP Status[0]	Bit{[9][4]} makes up this 2-bit field 00: Not a PTP packet 01: PTP v1 (UDP) packet Time Stamped 10: PTP v2 (L2) packet Time Stamped 11: PTP packet, but not time stamped
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 NMIC firmware.
6	BD_FLAG_VLAN_TAG	The frame associated with this buffer descriptor has an 802.1Q VLAN tag associated with it.
7	RSS_HASH_TYPE	Hash type of the receive packet. It indicates which hash_type was used on the receive packet if multiple hash type are defined.
8		
9	PTP Status[1]	See bit [4]
10	BD_FLAG_FRAME_HAS_ERROR	An error was detected by the NIC. The detected error type is set in the Error_Flag word of the receive buffer descriptor.
11	Reserved	
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.

Time Sync Registers

GRC MODE REG [0x6800]

Name	Bits	Access	Default Value	Description
Legacy	31:20			
Time Sync Mode Enable	19	RW	0	Write 1 to this bit to enable Time Sync Mode.
Legacy	20:0			

EAV REF COUNT CAPTURE LSB REG [Offset 0x6900]

The MSB and LSB registers are interface to the actual EAV Ref Counter hardware. The Counter value may be read via this pair anytime and even be overwritten by this pair anytime. While reading the pair, the hardware Counter does not stop, only its value is latched in this pair.

The only two legal sequences of accessing this pair is Read-LSB followed by Read-MSB and Write-LSB followed by Write-MSB.

Name	Bits	Access	Default Value	Description
EAV Reference Count [lower half]	31:3	RW	UUUU	LSB of the EAV Reference Count – Reading this LSB latches a Count in this pair until the time the corresponding MSB is read. Writing to this LSB latches the value in this pair and the subsequent write to the MSB transfers the 64-bit value to EAV Ref Counter and counting immediately resumes from there.
Reserved	2:0	RO	000	[2:0] shall always be 000.

EAV REF COUNT CAPTURE MSB REG [Offset 0x6904]

Name	Bits	Access	Default Value	Description
EAV Reference Count [Upper half]	31:0	RW	UUUU	MSB of the EAV Reference Count – See the pairing LSB register. Reading this register returns the MSB of the 64- bit EAV Ref count previously latched by performing an LSB read. Writing to this MSB transfers the 64-bit value, this plus previously latched LSB, to EAV Ref Counter and counting immediately resumes from there. Back to back writes to this MSB has no effect.

EAV REF CLOCK CONTROL REG [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 APEGPIO register accordingly.

Note: HW behavior shall be indeterminate in case of conflicting or duplicate assignment of GPIO pins to the same resource. A platform design MUST allocate its dedicated TimeSync_GPIO pin first before using any pin from APE_GPIO shared pool (we are talking PCB/Hardware design here).

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
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	0x0	
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 de-asserting edge of PCIe Reset
Reserved	7:3	RO	0x0	

Name	Bits	Access	Default Value	Description
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[0] REG [Offset 0X6910]

This LSB and 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 snapshotted by TimeSync/APE_GPIO [2:0] shall always be 000

EAV REF-COUNT SNAP-SHOT MSB[0] REG [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 snapshotted by TimeSync/APE_GPIO

EAV REF CORRECTOR REG [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	If 0, the correction is an addition If 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 – upon which the EAV REF COUNT is corrected by a 8ns unit and the acc is reloaded. A 0x0 value is not permissible.

TX TIME STAMP LSB REG [Offset 0x05C0]

This LSB and MSB pair captures time-stamp of transmit packets when marked to do so. The only legal sequence of accessing this pair is Read-LSB followed by Read-MSB. Once the LSB is read, the pair attains a frozen state, and is unfrozen immediately after a subsequent MSB read – if another TX packet passes to the wire between these two reads, it's time stamp capture request is ignored by hardware.

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.

TX TIME STAMP MSB REG [Offset 0X05C4]

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.

RX TIME STAMP LSB REG [Offset 0X06B0]

Name	Bits	Access	Default Value	Description
RX Time Stamp [lower half]	31:0	RO	U	LSB of the RX Time Stamp.

RX TIME STAMP MSB REG [Offset 0x06B4]

This MSB and LSB pair captures the time-stamp (63-bits) of received PTP packets when qualified to do so. HW overwrite of the Time Stamp value is controlled by an interlock policy – See "RX PTP CONTROL REG [Offset 0X06C8]" on page 164. The MSB and LSB registers may be accessed in the order of LSB first and MSB second – else the Valid / Interlock bit would not serve its purpose.

Name	Bits	Access	Default Value	Description
RX Time Stamp Valid / Interlock	31	RO	U	This bit is set by hardware in conjunction with posting a new value in the Time Stamp MSB and LSB fields. When this bit is 1 and SW executes a read to the RX TIME STAMP MSB Register, hardware shall clear this bit – the reset behavior of this bit is influenced by RX PTP CONTROL Register's [RX TIME STAMP INTERLOCK POLICY] field.
RX Time Stamp [Upper half]	30:0	RO	U	MSB of the RX Time Stamp.

RX PTP SEQUENCE ID REG [Offset 0X06B8]

Name	Bits	Access	Default Value	Description
Reserved	31:16	RO	0x0000	

Name	Bits	Access	Default Value	Description
PTP Sequence ID	15:0	RO	0x0000	This field reflects the Sequence ID extracted from the PTP packet which was last Time Stamped by RX hardware ***Note:*** Hardware shall blindly capture 2 octets from offset 30 of each time-stamped PTP packet – not all message types carry SequenceId, so use this value with discretion.

RX LOCK TIMER LSB REG [Offset 0x06C0]

Name	Bits	Access	Default Value	Description
Lock Time LSB Value	31:0	RW	0x0000	See RX Lock Timer MSB Register.

RX LOCK TIMER MSB REG [Offset 0x06C4]

See "RX PTP CONTROL REG [Offset 0X06C8]" on page 164

Name	Bits	Access	Default Value	Description
Reserved	31:16	RO	0x0	
Lock Time MSB Value	15:0	RW	0x00	This, concatenated with Lock Time LSB value, constitutes a 48-bit Lock Timer value. Precision of the value is 1ns – which equates to the precision of the EAV Reference Count.

RX PTP CONTROL REG [Offset 0X06C8]

Name	Bits	Access	Default Value	Description
Reserved	31:28	RO	0x00	
RX Time Stamp Interlock Policy	27:26	RW	01	This field determines how the successive RX Time Stamps are retained in the RX TIME STAMP Register, given that there is a chance of over-write. 00 => HW freely over-writes RX Time Stamp values. 01 => Interlock Mode – HW does not over-write unless SW reads a posted RX Time Stamp value. 10 => Lock Timer Mode - HW does not over-write unless SW reads a posted RX Time Stamp value OR the Lock Timer has expired. 11 => Reserved ***Note:*** Changing the state of this field while receiving PTP traffic may result in abrupt HW behavior. This field is best configured statically.

Name	*Bits*	*Access*	*Default Value*	*Description*
PTP RX Time Stamping Enable bit-map	25:23	RW	00	Write a 1 to enable – all permutations allowed. [25] RX Time Stamp PTP V1 - L4 packets [24] RX Time Stamp PTP V2 - L4 packets [23] RX Time Stamp PTP V2 - L2 packets See field [15:0]
Reserved	22:16	RO	0x00	
Stamp-able PTP Message Type bit-map	15:0	RW	0x0000	This field, in conjunction with field [25:23] qualifies a particular PTP message type packet to be time stamped by hardware. Write a 1 in a bit position to enable time stamping of the corresponding message type and write a 0 in a bit position to disable time stamping of the corresponding message type: [00] => Sync Event [01] => Delay_Req [02] => Pdelay_Req [03] => Pdelay_Resp [04 – 07] => N/A [08] => Follow_Up [09] => Delay_Resp [10] => Pdelay_Resp_Follow_Up [11] => Announce [12] => Signaling [13] => Management [14 – 15] => N/A ***Note:*** Not all message Types are available in PTP V1.

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] Register.

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].

Name	Bits	Access	Default Value	Description
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[1] Register

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-COUNT SNAP-SHOT LSB[1] REG [Offset 0X6930]

This LSB and 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 snapshotted 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 snapshotted by TimeSync/APE_GPIO.

Section 9: 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 BCM5718 is PCIe v1.1 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.

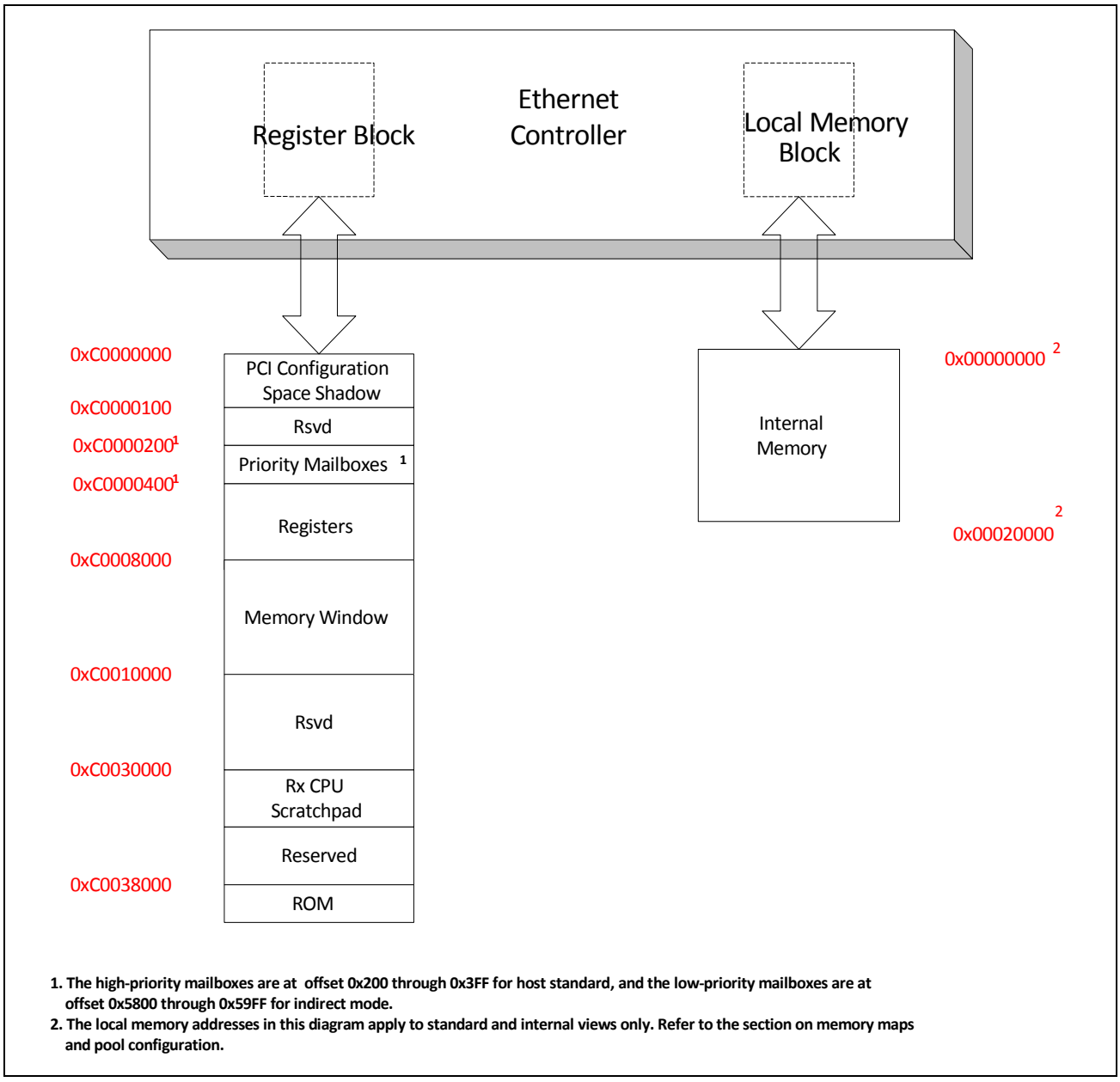
Example: Intel x86 architectures use two I/O mapped I/O addresses 0xCF8 and 0xCFC for host-based access to PCI configuration space. Should a host device driver access these I/O addresses on every device read/write, CPU overhead would grow greatly. Generally, host device drivers should not use PCI configuration 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 32).

Figure 32: Local Contexts



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

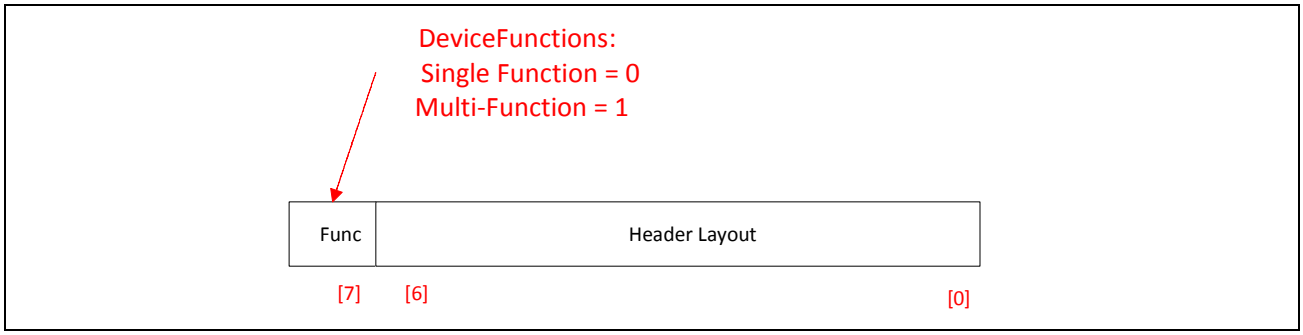
PCI Configuration Space Registers

The Ethernet controller configuration space can be broken into two regions: Header and Device Specific. Table 47 on page 172 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 33: 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 47.

Table 47: Device Specific Registers

Register	*Cross Reference*
Miscellaneous Host Control	"Miscellaneous Host Control Register (offset: 0x68)" on page 283.
PCI State	"PCI State Register (offset: 0x70)" on page 286.
Register Base Address	"Register Base Register (offset: 0x78)" on page 287.
Memory Base Address	"Memory Base Register (offset: 0x7C)" on page 287.
Register Data	"Register Data Register (offset: 0x80)" on page 287.
Memory Window Data	"Memory Data Register (offset: 0x84)" on page 288.
UNDI Receive BD Standard Producer Ring Producer Index Mailbox	"UNDI Receive BD Standard Producer Ring Producer Index Mailbox Register (offset: 0x98–0x9C)" on page 288.
UNDI Receive Return Ring Consumer Index Mailbox	"UNDI Receive Return Ring Consumer Index Register (offset: 0x88–0x8C)" on page 288.
UNDI Send BD Producer Index Mailbox	"UNDI Send BD Producer Index Mailbox Register (offset: 0x90–0x94)" on page 288.

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.

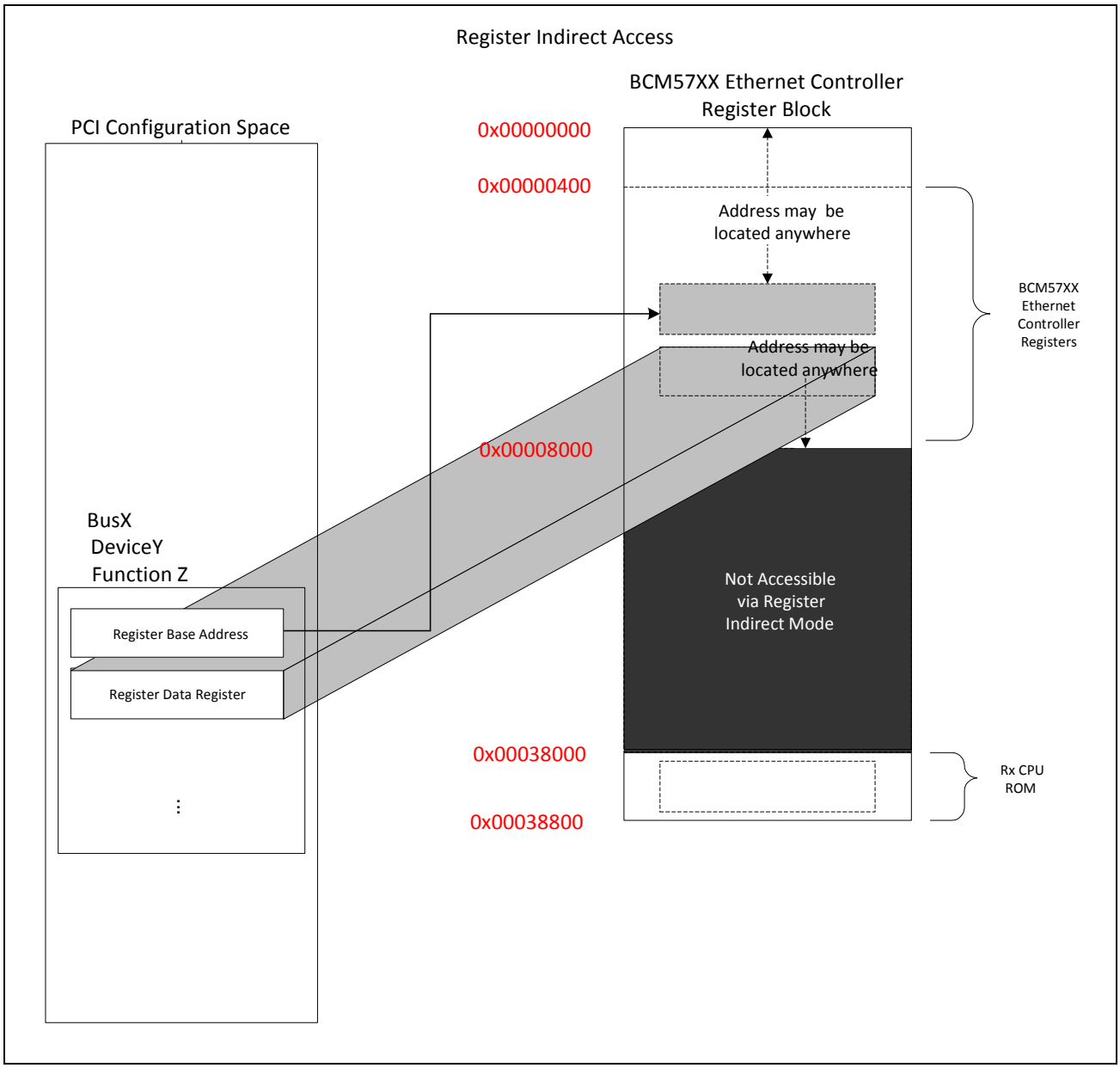
Note: Host software must assert the Indirect_Mode_Access bit in the Miscellaneous Host Control register (see "Miscellaneous Host Control Register (offset: 0x68)" on page 283) to enable indirect mode.

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 34 on page 174).

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 34: Register Indirect Access

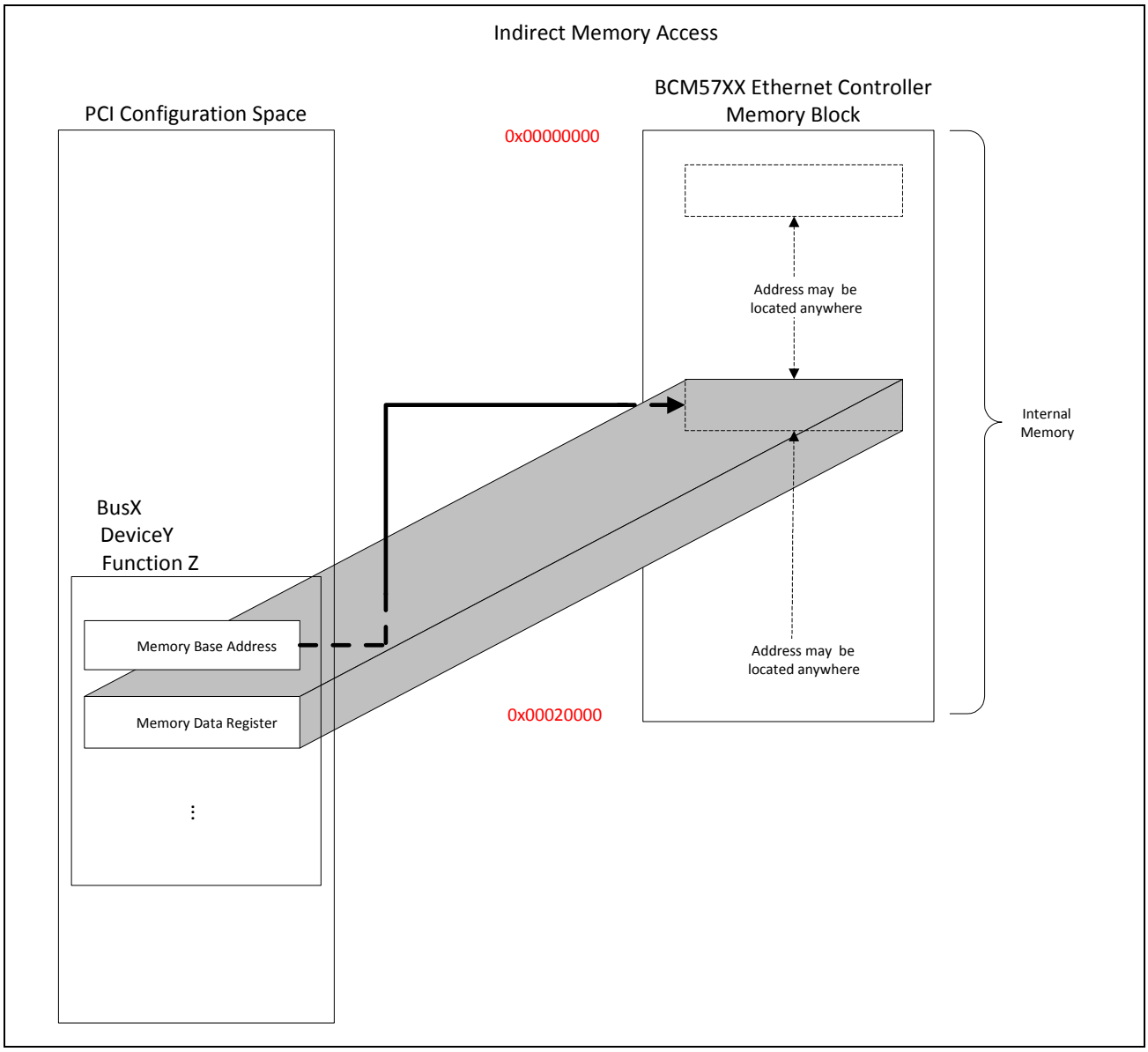


Indirect Memory Access

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 35 on page 176).

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 35: Indirect Memory Access



UNDI Mailbox Access

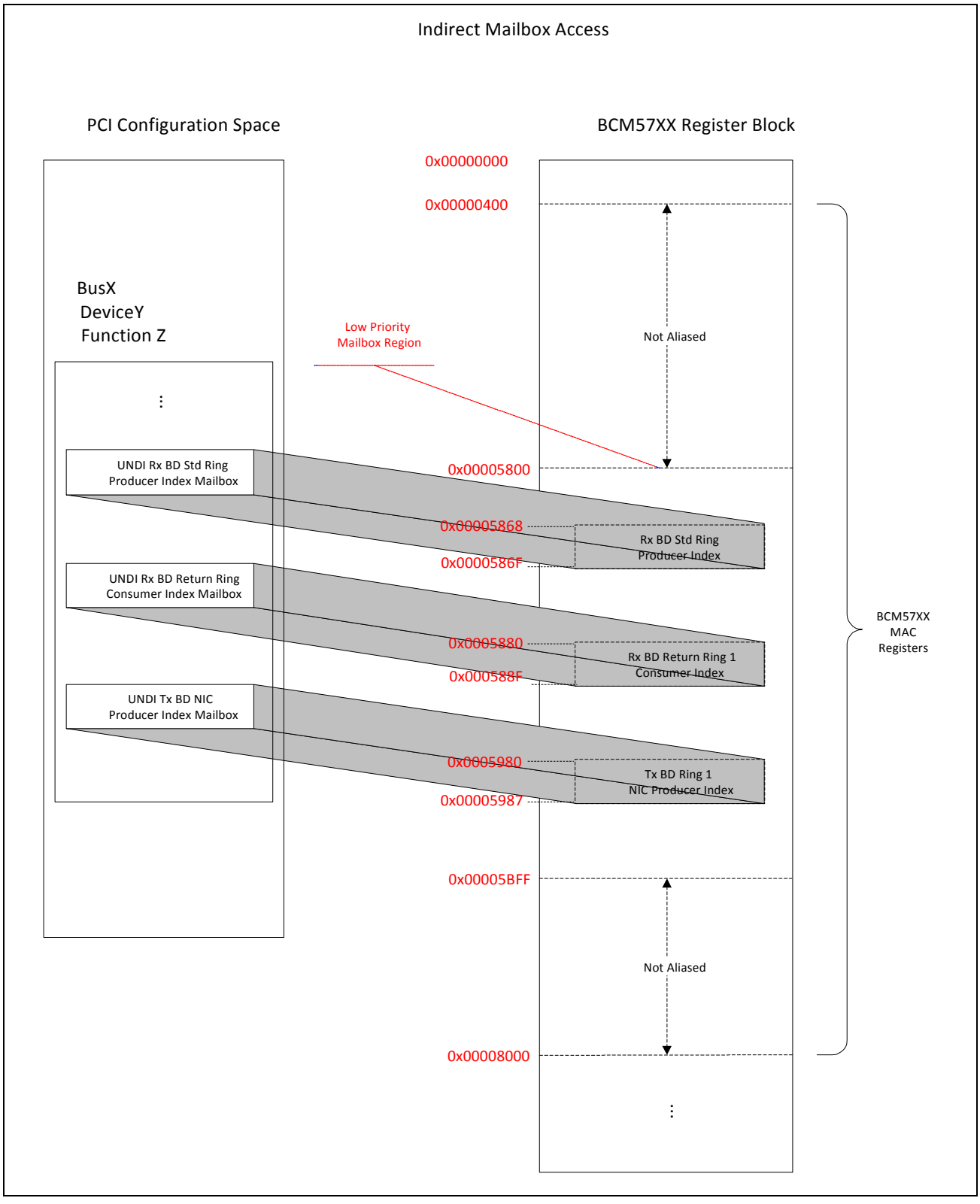
The UNDI mailboxes are shadows of Ethernet controller mailbox registers. All mailboxes reside in the Ethernet controller register block, not memory block. Unlike register and memory indirect access, the UNDI Mailboxes shadows are mapped 1:1 to a Ethernet controller register; these shadow registers do 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 Register (offset: 0x5898-0x589F)" on page 465), 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 464). 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 36 on page 178).

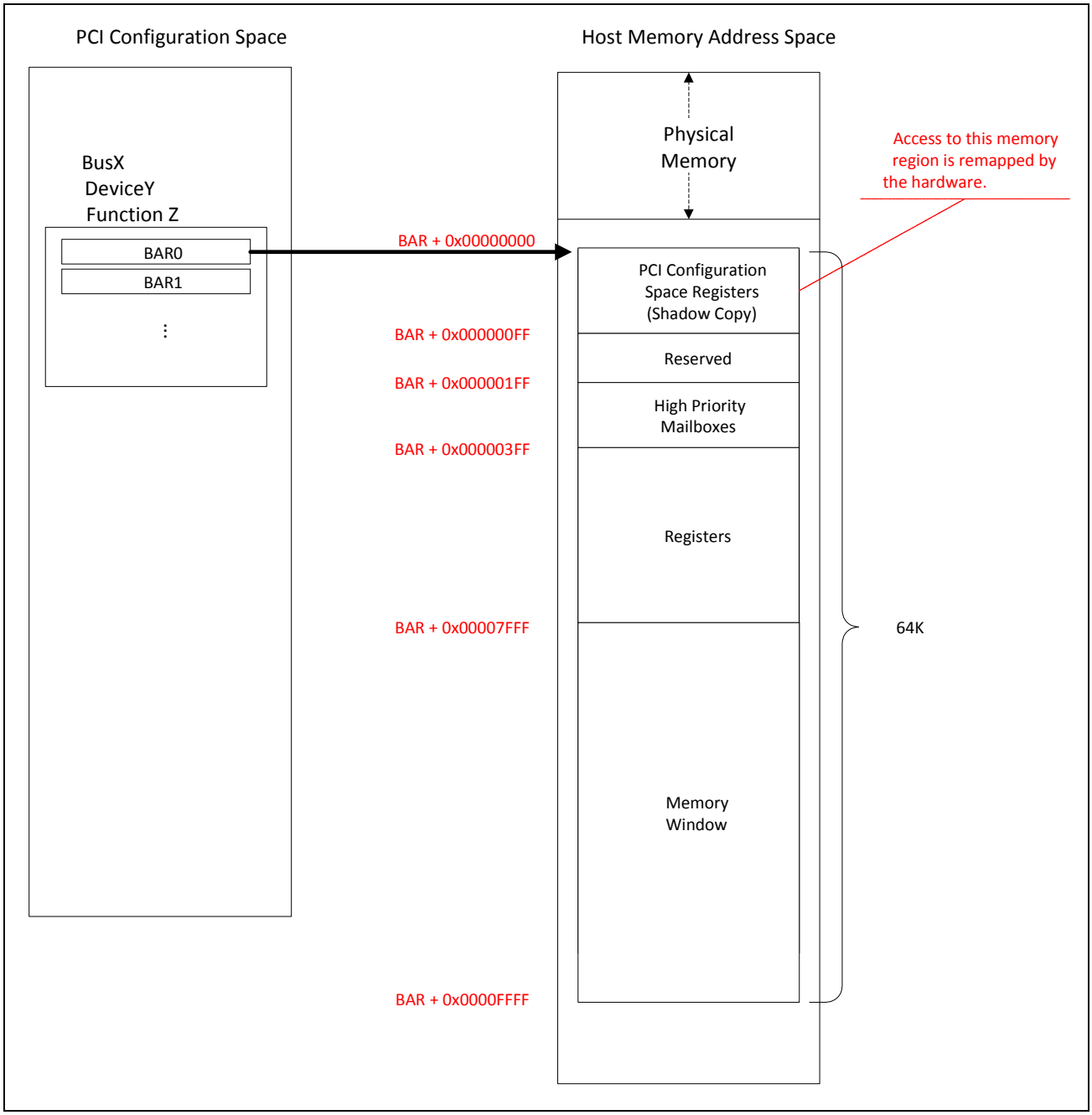
Figure 36: Low-Priority Mailbox Access for Indirect Mode



Standard Mode

Standard mode is the most useful memory mapped I/O view provided by the Ethernet controller (see Figure 37). 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 37: Standard Memory Mapped I/O Mode



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

Table 48: 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 13: "Ethernet Controller Register Definitions," on page 270 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 38). 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 38: Memory Window Base Address Register

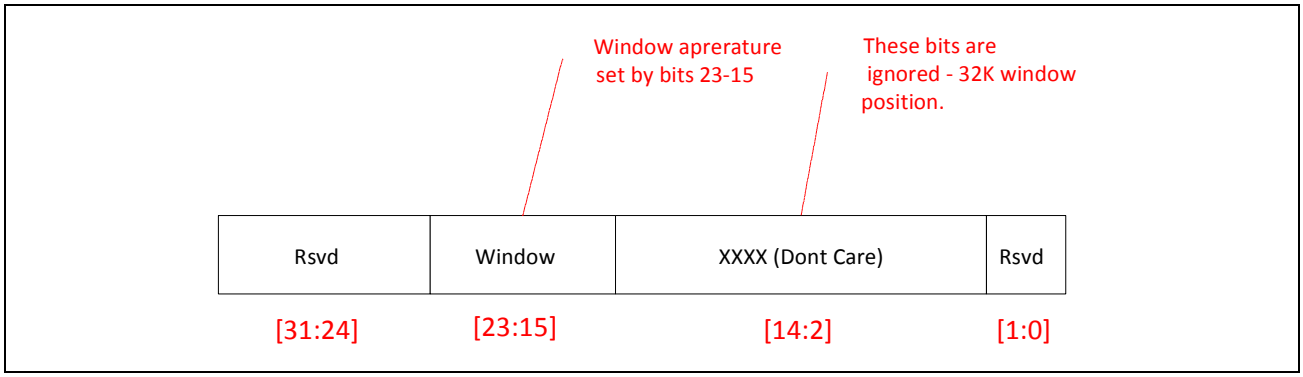
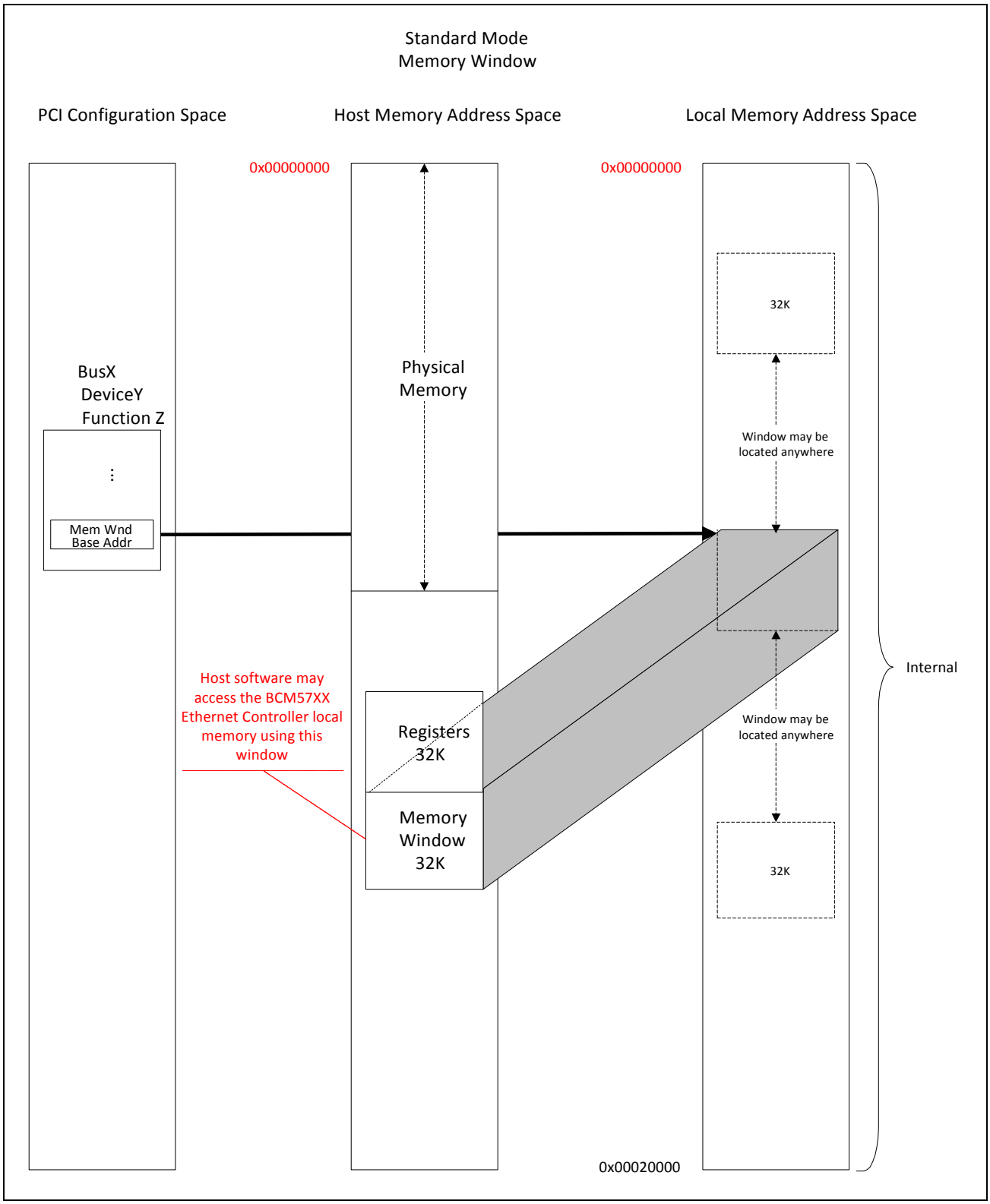


Figure 39 on page 181 shows how the 32K window can float in the Ethernet controller's local memory. The window aligns on 32K boundaries.

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 39 on page 181 shows the relationship between the Memory_Window_Base_Address register and the Memory Window.

Figure 39: Standard Mode Memory Window



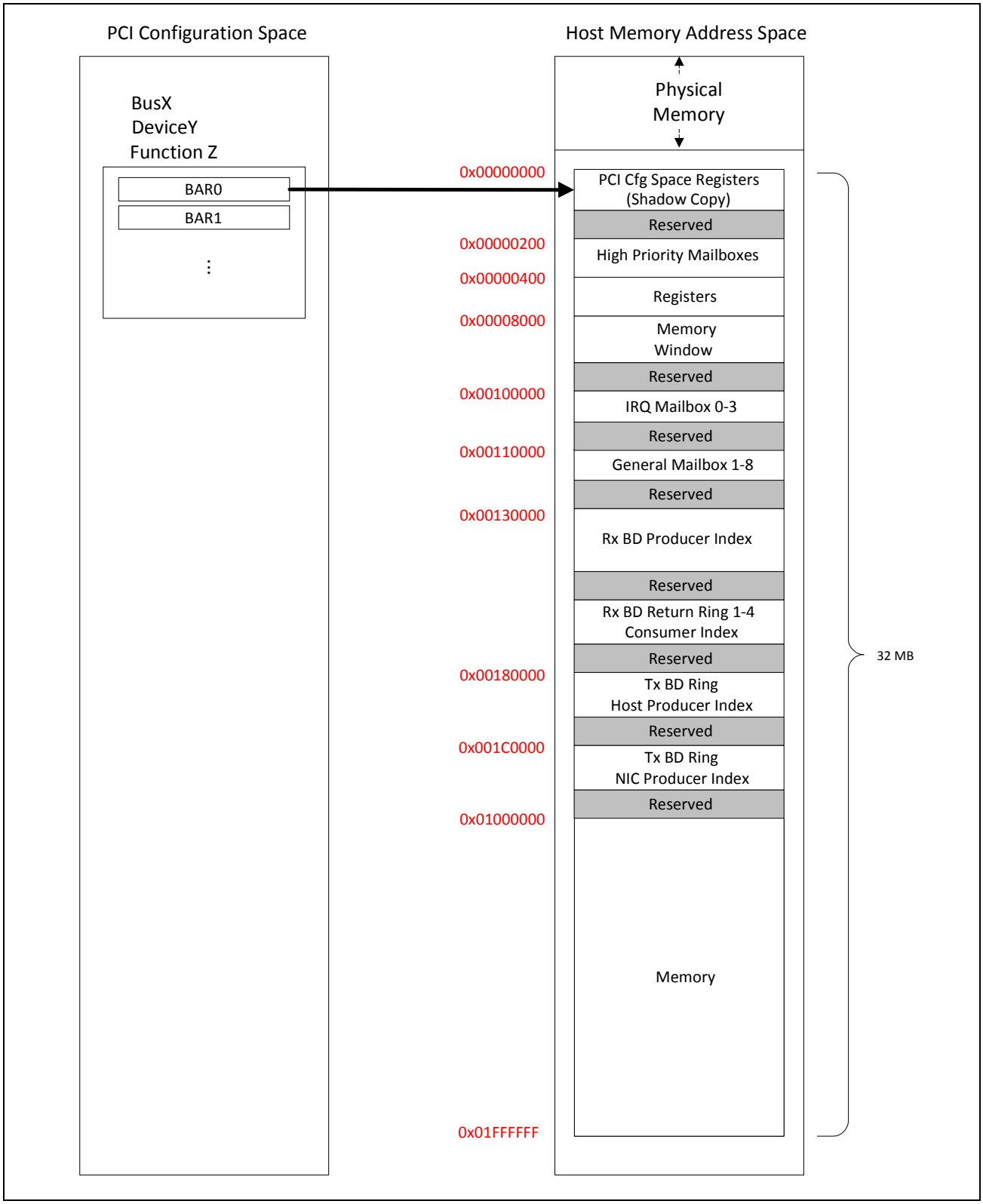
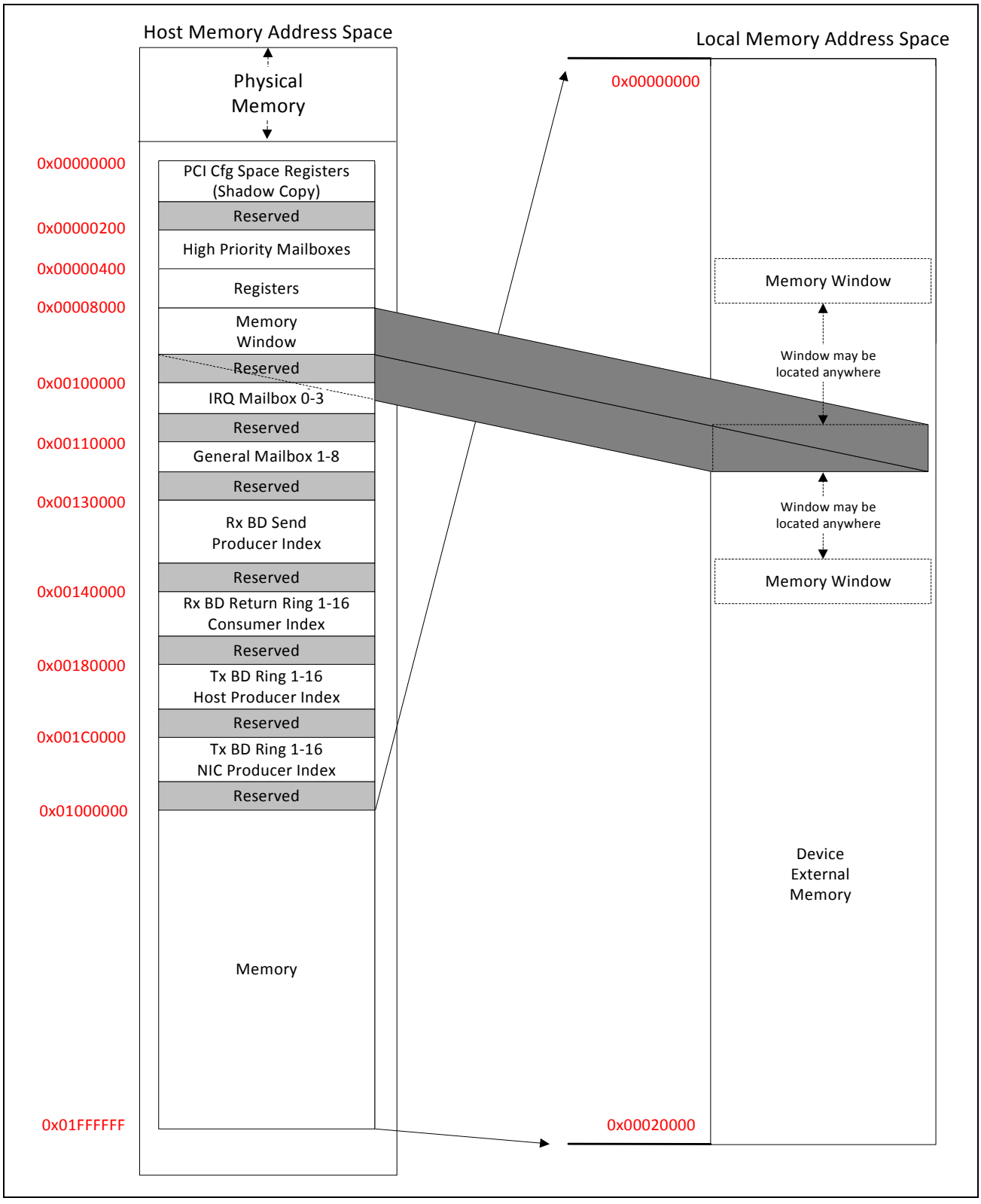

PCI Configuration Space
Host Memory Address Space
BusX
DeviceY
Function Z
BAR0
BAR1
Physical Memory
0x00000000
PCI Cfg Space Registers (Shadow Copy)
Reserved
0x00000200
High Priority Mailboxes
0x00000400
Registers
0x00008000
Memory Window
Reserved
0x00100000
IRQ Mailbox 0-3
Reserved
0x00110000
General Mailbox 1-8
Reserved
0x00130000
Rx BD Producer Index
Reserved
Rx BD Return Ring 1-4 Consumer Index
Reserved
0x00180000
Tx BD Ring Host Producer Index
Reserved
0x001C0000
Tx BD Ring NIC Producer Index
Reserved
0x01000000
Memory
0x01FFFFFF
32 MB

Figure 40: Techniques for Accessing Ethernet Controller Local Memory



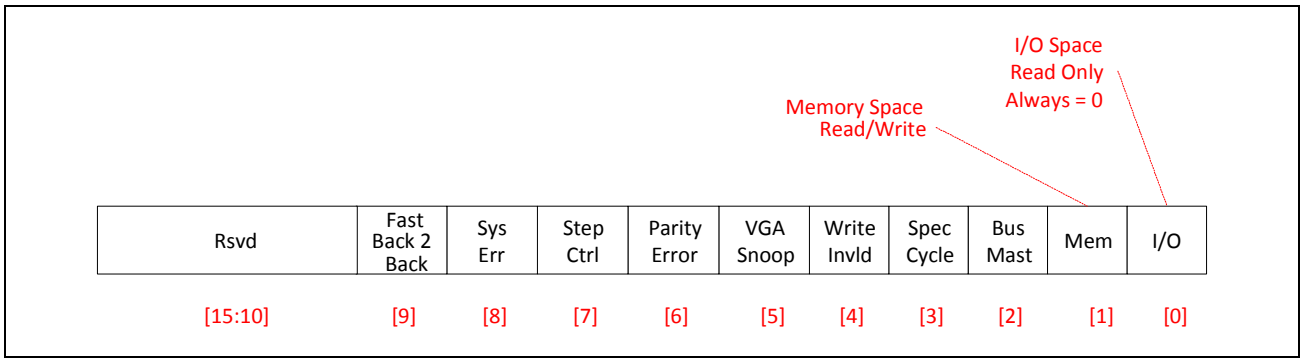
Memory Mapped I/O Registers

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 41). 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 41: PCI Command Register



PCI State Register

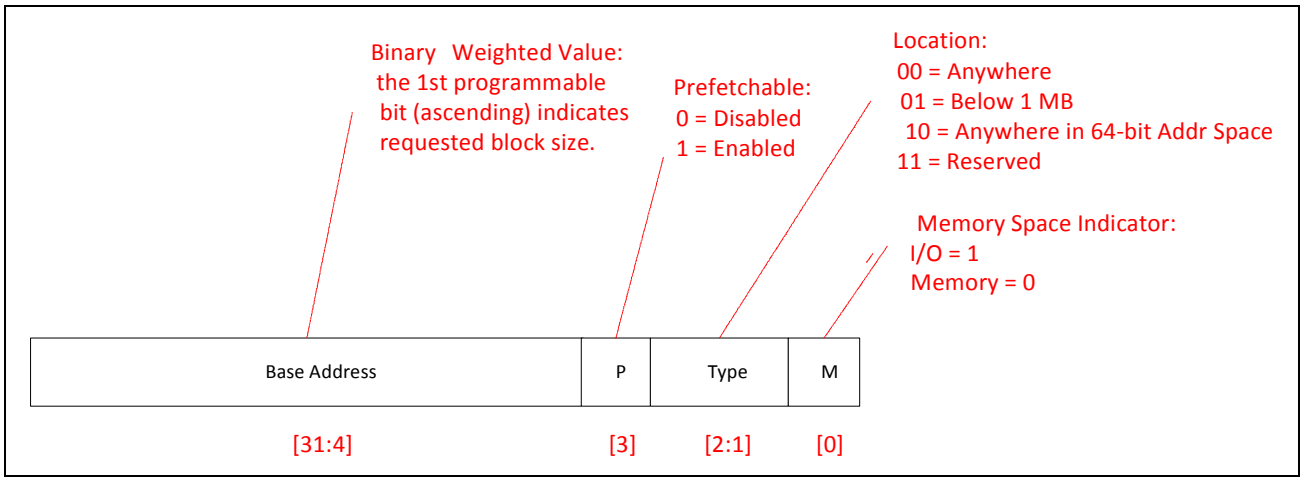
The PCI_State register is 32-bits wide. Operating mode is set with the Flat_View bit in the PCI_State register. When the Flat_View bit is asserted, the Ethernet controller decodes a 32M of block host memory. When the Flat_View bit is de-asserted, the Ethernet controller decodes a 64K block of host memory.

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 42).

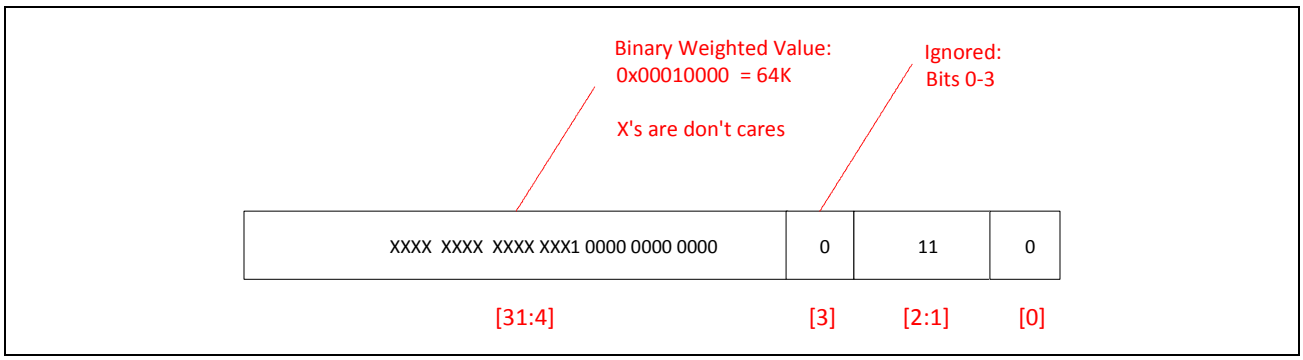
- Bits 4–31 in the PCI_Base_Address register are selectively programmable based on the amount of host memory requested. The PnP BIOS/OS will use an algorithm to test the BAR bits and determine the amount of physical memory requested.
- The Memory_Space_Indicator bit designates whether the BAR is memory 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 42: 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 43 shows the BAR register and the bits returned to the OS/BIOS during resource allocation.

Figure 43: PCI Base Address Register Bits Read in Standard Mode

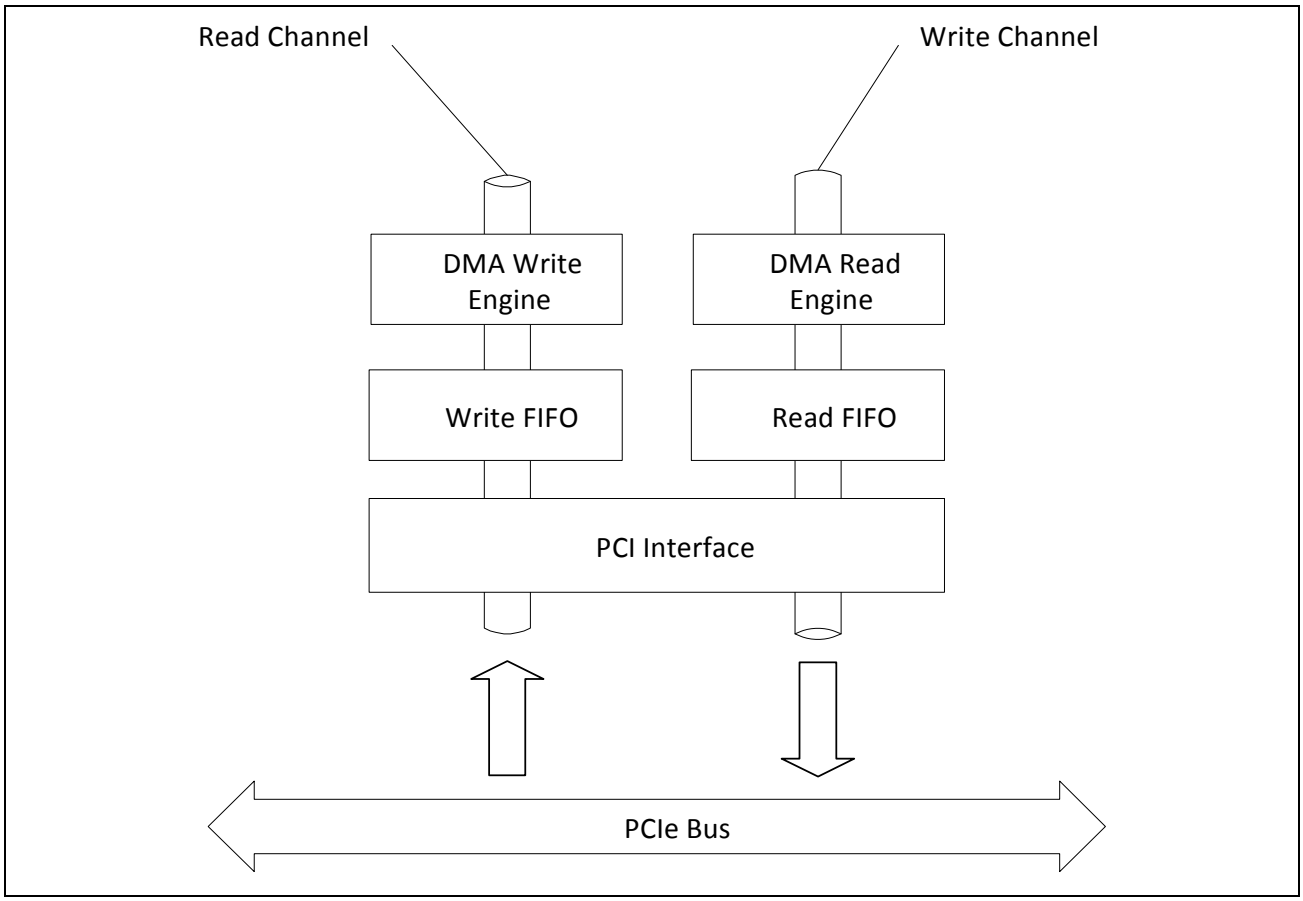


Bus Interface

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 BCM5718 family architecture identifies two channels—a read DMA channel and a write DMA channel. Each channel corresponds to the appropriate DMA engine (see Figure 44). The configuration of the DMA engines and the PCI interface is discussed in this section.

Figure 44: 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 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)" on page 286). 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 nonzero value appropriate for the expansion ROM size selected in NVRAM (see Section 4: "Common Data Structures," on page 70) 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 (PCI v2.1 for BCM5719) 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.

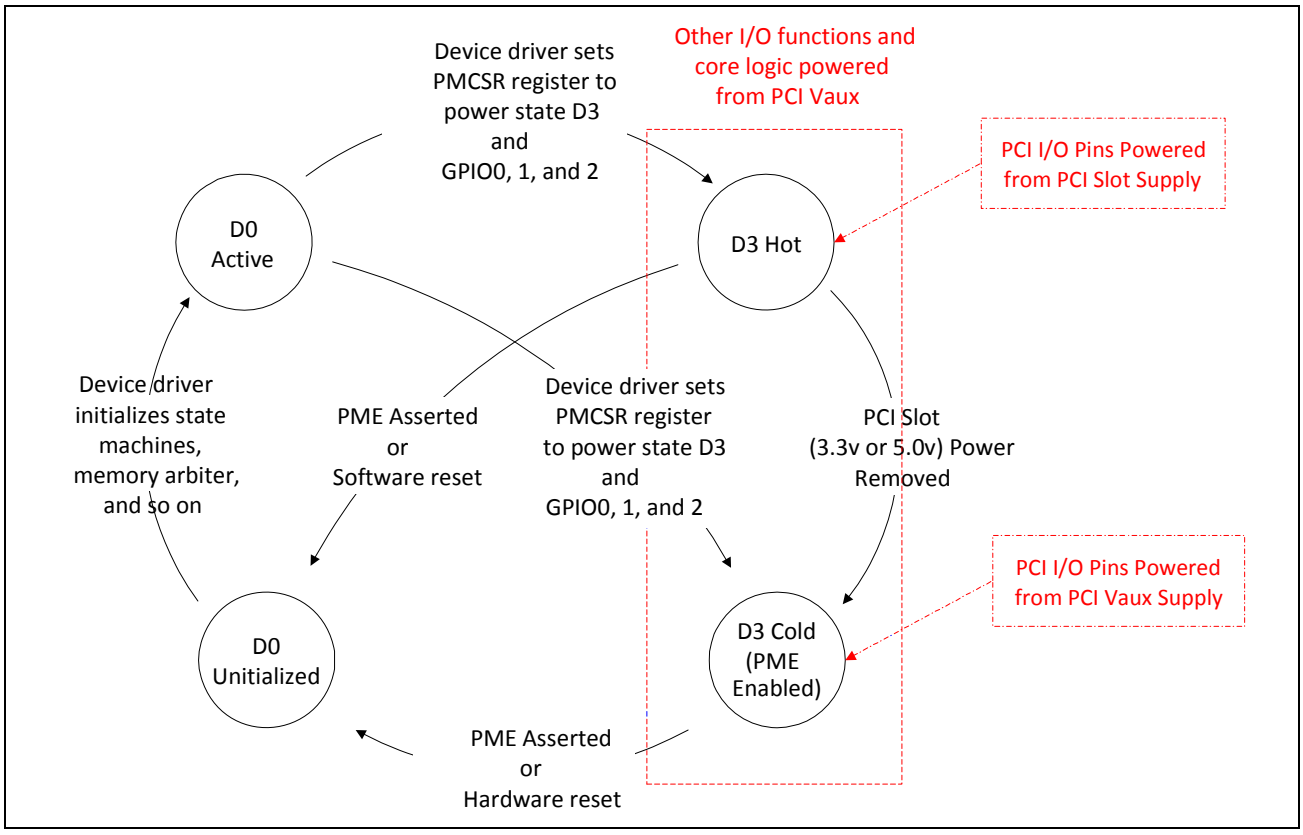
The following functional blocks are integral to MAC power management:

- PMSCR register
- PCI Clock Control register
- Miscellaneous Control register
- WOL
- PCI Vaux Supply
- PCI Slot Power Supply
- GPIO

Operational Characteristics

Figure 45 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 45: Power State Transition Diagram



Device State D0 (Uninitialized)

The D0 state is entered after a PCI reset or device (software) reset. The assertion of $\overline{\text{PME}}$ causes the PCI bridge to drive $\overline{\text{RST}}$. 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 137) to move the MAC into a D0 active state.

When the BCM5718 family NetXtreme devices detect 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 213.

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 49. 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.

Table 49: GPIO Usage for Power Management for Broadcom Drivers[a]

Function	*Description*	*GPIO0*	*GPIO1*	*GPIO2*
VAux	Sequence for switching to VAux	0	1	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 50 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 50: Ethernet Controller Power Pins

Supply Pins	*D0 Normal*	*D3 Hot, D3 Cold*
VDDIO	PCIe Slot Vmain	PCIe Slot Vaux
VDDC	PCIe Slot Vmain	PCIe Slot Vaux

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)" on page 283); 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.

Note: For the 57818 family chip, the clock control register (offset: 0x68) does not need to be configured.

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 188) 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. 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.

Disable Device Through BIOS

The Ethernet controllers can be disabled (that is, placed in Low Power IDDQ mode) 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.

Endian Control (Byte and Word Swapping)

Note: Setting register 0x68 bit 2 (Enable Endian Byte Swap) causes PCI configuration reads to the following registers to become swapped:

- 0x40
- 0x68 - 0x9C
- 0xF4 - 0xFF

This is different behavior from previous NetXtreme controllers. Reference BCM5718 Family errata relating to byte swap control for additional information.

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 51: 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 52: Storage of Big-Endian Data

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

Table 53: 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)" on page 283). 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)" on page 283). 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 46 and Figure 47.

	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

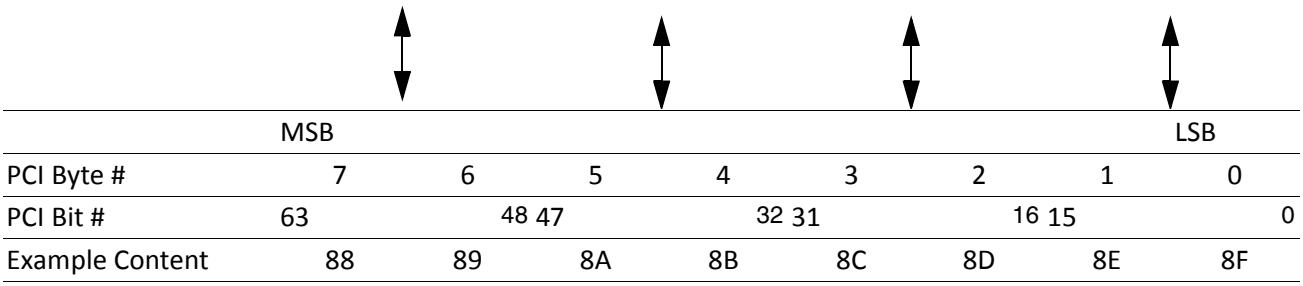


Figure 46: Default Translation (No Swapping) on 64-Bit PCI

Internal Byte Ordering **PCI 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 47: 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 54: 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 47 on page 196. 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 Ordering **PCI Byte Ordering**

	31	16	15	0
0x00	88	89	8A	8B

31	16	15	0	
88	89	8A	8B	0x00

0x04	8C	8D	8E	8F		8C	8D	8E	8F	0x04

Figure 48: 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 49: 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

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 55: 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 56: 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 57: 32-Bit PCI Bus (WSD = 0, BSD = 0)

B3	B2	B1	B0
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 58: 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 59: 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 60: 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 61: 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 62: 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 63: 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 64: 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 65: Send Buffer Descriptor (Big-Endian 32-Bit format)

Byte #	0		1	2	3	
Bit #	31	16	15		0	
	MSB	Host Address			LSB	0x00 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 66: 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 67: 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 68: 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 69: 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 10: 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 BCM5718 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 the ability 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 controller 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 322) 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 313) and the Mode_Control.Interrupt_on_MAC_Attention bit (see "Mode Control Register (offset: 0x6800)" on page 472). 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 83) 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

```
// Setup the value that we are going to write to MI 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 MI communication register
MII_Communication_Register = Value32
// Now read back MI Communication register until the start bit
// has been cleared or we have timed out (>5000 reads)
Loopcount = 5000
While (LoopCount > 0)
Begin
    Value32 = MII_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
// Now return the value that we read (lower 16 bits of reg)
Return (Value32 & 0xffff)
```

Writing a PHY Register

```
// Setup the value that we are going to write to MI 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 MI communication register
MII_Communication_Register = Value32
// Now read back MI Communication register until the start bit
// has been cleared or we have timed out (>5000 reads)
Loopcount = 5000
While (LoopCount > 0)
Begin
    Value32 = MII_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
```

PHY Loopback Configuration

External PHY Loopback

```
10Base-T
- Write 0x0100 to PHY register 00h  // Force 10Base-T, full duplex
- Write 0x1000 to PHY register 1Eh  // Force link  (required for 10Base-T)
- Set MAC register 0x400[3:2] = 01b // Set MII
- Set MAC register 0x400[1]   = 0   // Force full-duplex operation

100Base-TX
- Write 0x2100 to PHY register 00h  // Force 100Base-TX, full duplex
- Write 0x1000 to PHY register 1Eh  // Force link  (required for 100Base-TX)
- Set MAC register 0x400[3:2] = 01b // Set MII
- Set MAC register 0x400[1]   = 0   // Force full-duplex operation

1000Base-T
- Write 0x1B00 to PHY register 09h  // Enable 1000Base-T master mode
- Write 0x8400 to PHY register 18h  // Enable external loopback mode
- Write 0x1000 to PHY register 1Eh  // Force link (optional for 1000Base-T)
- Write 0x0140 to PHY register 00h  // Force 1000Base-T operation
- Set MAC register 0x400[3:2] = 10b // Set GMII
- Set MAC register 0x400[1] = 0     // Force full-duplex operation
```

Internal PHY Loopback

```
---------------------
10Base-T
- Write 0x4100 to PHY register 00h  // Force 10Base-T, full duplex, internal loopback
- Write 0x1000 to PHY register 1Eh  // Force link  (required for 100Base-TX)
```

```
- Set MAC register 0x400[3:2] = 01b // Set MII
- Set MAC register 0x400[1]   = 0   // Force full-duplex operation

100Base-TX
- Write 0x6100 to PHY register 00h  // Force 100Base-TX, full duplex, internal loopback
- Write 0x1000 to PHY register 1Eh  // Force link (required for 100Base-TX)
- Set MAC register 0x400[3:2] = 01b // Set MII
- Set MAC register 0x400[1]   = 0   // Force full-duplex operation

1000Base-T
- Write 0x4140 to PHY register 00h  // Force 1000Base-T, full duplex, internal loopback
- Write 0x1000 to PHY register 1Eh  // Force link (optional for 1000Base-T)
- Set MAC register 0x400[3:2] = 10b // Set GMII
- Set MAC register 0x400[1] = 0     // Force full-duplex operation
```

PHY Configuration Auto-Negotiation (10/100/1000 Speed with Half and Full Duplex Support)

Basic PHY pseudo-code:

- Enable 10/100/1000 PHY loopback mode
- Enable/disable Auto-MDI crossover
- Enable auto-negotiation
- Forced 10/100/1000 link speeds

```
-------------------
PHY Reset Procedure
-------------------
uint16_t val16;

// Initiate PHY reset by setting 0x00[15] = 1
phy_write(0x00, 0x8000);

// Wait up to 100ms for 0x00[15] = 0
for (int i = 0; i < 100; i++) {
  val16 = phy_read(0x00);

  if ((val16 & 0x8000) == 0)
    break;

  // Delay for 1ms
  delay_us(1000);
}

----------------------
PHY Loopback Procedure
----------------------

// Force link down by enabling loopback
phy_write(0x00, (1 << 14));

// Wait up to 15ms for link to drop
for (int i = 0; i < 15000; i++) {
  if ((phy_read(0x01) & (1 << 4) == 0)
    break;
```

```
  // Wait 10us
  delay_us(10);
}

--------------------------------------------------
Configure Auto-MDIX (Cable pair swapping for 10/100)
--------------------------------------------------

// Read Misc Control Register (PHY 18h, Shadow 7)
phy_write(0x18, 0x7007);
val16 = phy_read(0x18)

//Set Force Auto MDIX Mode (Phy 18h, Shadow 7, bit 9)
val16 |= (1 << 9);

// Write Misc control Register (set write enable bit)
phy_write(0x18, val16 | (1 << 15));

// Enable Auto MDIX Crossover (Phy 10h, bit 14)
val16 = phy_read(0x10);
val16 &= ~(1 << 14);
phy_write(0x10, val16);

------------------------------------------------------------------------
Autonegotiation (10/100/1000 speed with half and full duplex support)
------------------------------------------------------------------------

uint16_t gig = 0, anar = 0;

// Reset PHY

// Enable auto-MDIX

// Force loopback

// Select pause and asymmetric pause advertisement for Ethernet
anar = (1 << 11) | (1 << 10) | (00001b << 0);

// Select 10/100 half/full duplex advertisement
anar |= (1 << 8) | (1 << 7) | (1 << 6) | (1 << 5);

// Enable 10/100 autoneg advertisement
phy_write(0x04, anar);

// Select 1000Mb full-duplex operation
gig = (1 << 9);

// Advertise full-duplex operation
phy_write(0x09, gig);

// Enable and restart autonoegotiation
phy_write(0x00, ((1 << 12) | (1 << 9)));


--------------------
10Base-T Half-Duplex
--------------------
```

```
uint16_t bmcr = 0;

// Reset PHY

// Enable auto-MDIX

// Set link speed to 10Mb
bmcr = (0 << 6) | (0 << 13);

// Set half-duplex operation
bmcr |= (0 << 8);

// Force loopback

// Disable 1000Mb autoneg advertisement
phy_write(0x09, 0);

// Disable 10/100 autoneg advertisement
phy_write(0x04, (00001b << 0));

// Write forced link speed
phy_write(0x00, bmcr);
```

```
--------------------
10Base-T Full-Duplex
--------------------

uint16_t bmcr = 0;

// Reset PHY

// Enable auto-MDIX

// Set link speed to 10Mb
bmcr = (0 << 6) | (1 << 13);

// Set full-duplex operation
bmcr |= (1 << 8);

// Force loopback

// Disable 1000Mb autoneg advertisement
phy_write(0x09, 0);

// Disable 10/100 autoneg advertisement
phy_write(0x04, (00001b << 0));

// Write forced link speed
phy_write(0x00, bmcr);
```

```
----------------------
100Base-TX Half-Duplex
----------------------

uint16_t bmcr = 0;
```

```
// Reset PHY

// Enable auto-MDIX

// Set link speed to 10Mb
bmcr = (0 << 6) | (1 << 13);

// Set half-duplex operation
bmcr |= (0 << 8);

// Force loopback

// Disable 1000Mb autoneg advertisement
phy_write(0x09, 0);

// Disable 10/100 autoneg advertisement
phy_write(0x04, (00001b << 0));

// Write forced link speed
phy_write(0x00, bmcr);

----------------------
100Base-TX Full-Duplex
----------------------

uint16_t bmcr = 0;

// Reset PHY

// Enable auto-MDIX

// Set link speed to 10Mb
bmcr = (0 << 6) | (1 << 13);

// Set full-duplex operation
bmcr |= (1 << 8);

// Force loopback

// Disable 1000Mb autoneg advertisement
phy_write(0x09, 0);

// Disable 10/100 autoneg advertisement
phy_write(0x04, (00001b << 0));

// Write forced link speed
phy_write(0x00, bmcr);

----------------------
1000Base-T Half-Duplex
----------------------
- Half duplex operation not supported at 1000Mb per 802.3 specification

----------------------
1000Base-T Full-Duplex
----------------------
- Forced speed not supported, must use autoneg but only advertise 1000Mb speed
```

```
uint16_t bmcr = 0, gig = 0;

// Reset PHY

// Enable auto-MDIX

// Set link speed to 1000Mb
bmcr = (1 << 6) | (0 << 13);

// Select full-duplex operation
gig = (1 << 9);

// Force loopback

// Disable 10/100 autoneg advertisement
phy_write(0x04, (00001b << 0));

// Advertise full-duplex operation
phy_write(0x09, gig);

// Enable and restart autonoegotiation
phy_write(0x00, (bmcr | (1 << 12) | (1 << 9)));
```

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 MI Mode register. The device will poll for the link status bit in the transceiver.
- Command Control. Writing to the MI Communications register directly to either read or write the transceiver registers.

Autopolling mode has the lower priority and it will be stalled any time there is an active operation through the MI 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 MI 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 205 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 205 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

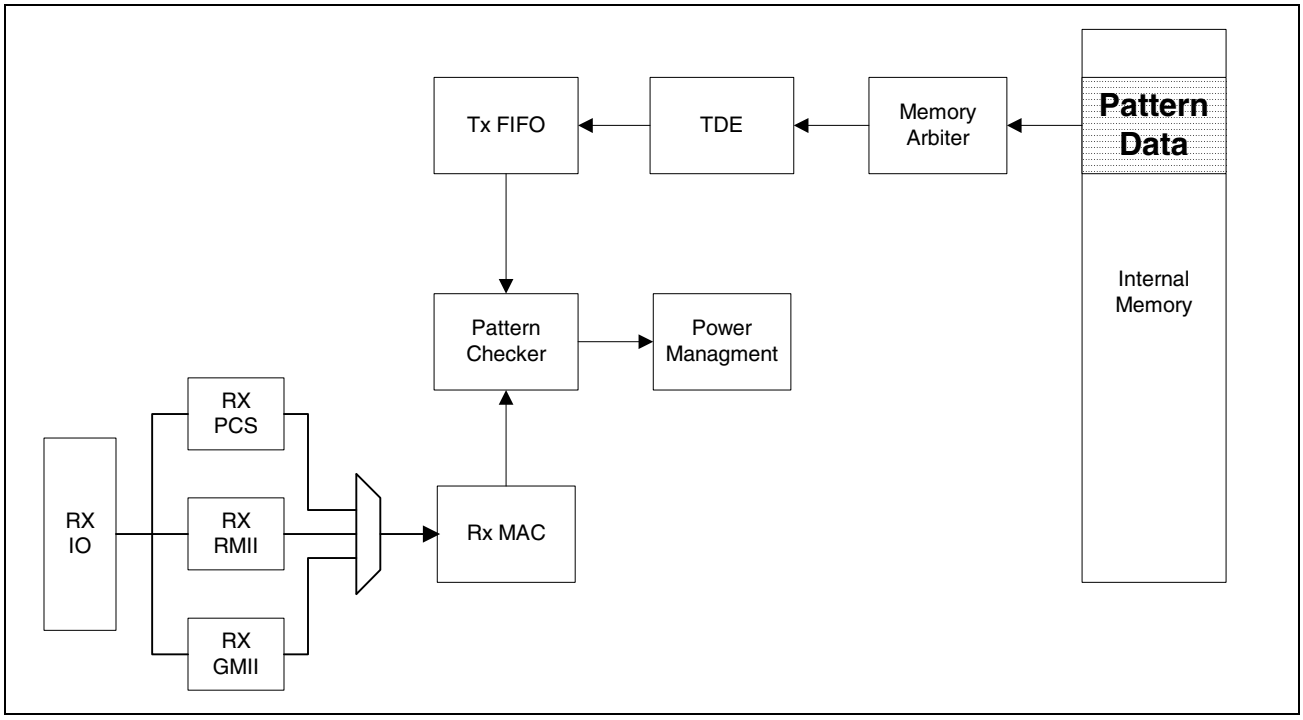
The Ethernet controller is capable of WOL in 10/100 Mbps for copper-based controllers.

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 50). 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 50: 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 70 shows the required memory regions for the WOL pattern.

Table 70: Required Memory Regions for WOL Pattern

Internal Address Range	*Size*	*Name*
0x8000–0x8FFF	8 KB	Miscellaneous Memory Region

For 5718 Family chip, the address range starts from 0x20000 (ASF Disable) / 0x24000 (ASF Enable).

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 317 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 "WOL Pattern Configuration Register (offset: 0x434)" on page 317 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.

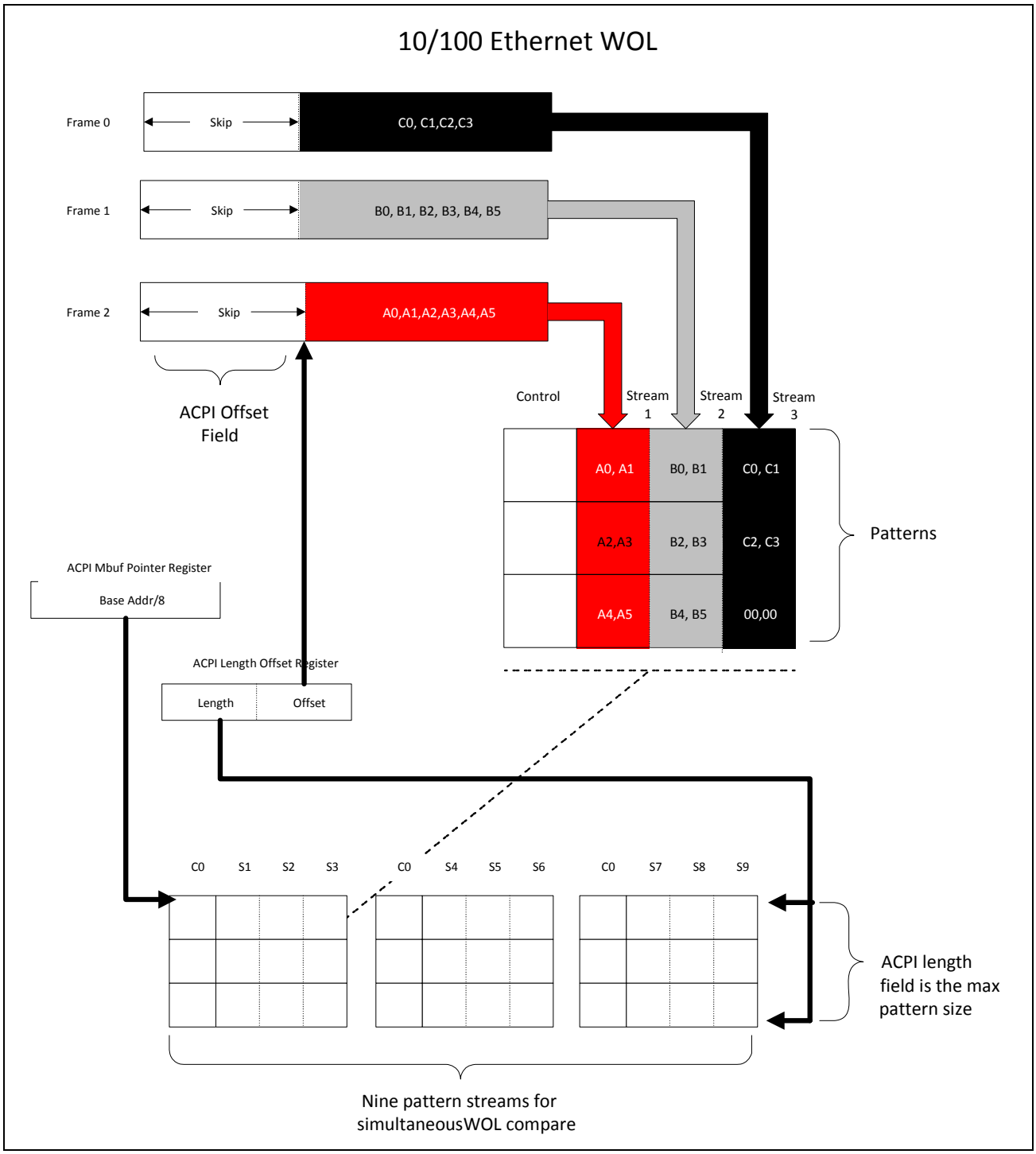
The programmer must use the following equation to calculate the number of clock cycles required to match patterns at 10/100 wire-speed: (Length/2) * 3 MA clocks. The equation breaks down as follows:

  - 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 51 on page 217, three Ethernet frames are compared against the nine available patterns.

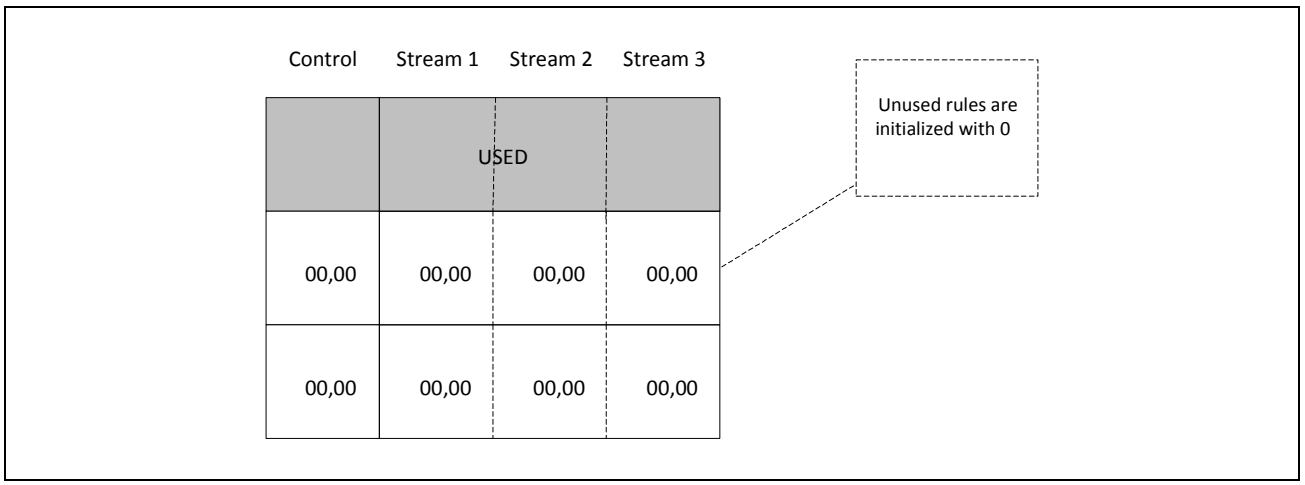
Figure 51: Comparing Ethernet Frames Against Available Patterns (10/100 Ethernet WOL)



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 52).

Figure 52: 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 72). 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 72). 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 71 and Table 72 use Sx notation to denote separate comparison streams. The D0 notation indicates the first 2 bytes in the packet stream are compared.

Table 71: 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 72: 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	RW
60	S0 Low Byte Enable	Enable S0 lower byte for comparison	RW

Table 72: Frame Control Field for 10/100 Mbps Mode (Cont.)

Bits	Field	Description	Access
59	S1 High Byte Enable	Enable S1 higher byte for comparison	RW
58	S1 Low Byte Enable	Enable S1 lower byte for comparison	RW
57	S2 High Byte Enable	Enable S2 higher byte for comparison	RW
56	S2 Low Byte Enable	Enable S2 lower byte for comparison	RW
55:51	Reserved	–	–
50	S0 Done	End of S0 Stream	RW
49	S1 Done	End of S1 Stream	RW
48	S2 Done	End of S2 Stream	RW

Table 73 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 73: 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 74: 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 75). 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 75: 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 76). 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 76: 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 77 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 77: 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 78 lists the WOL mode control registers in the Ethernet controllers.

Table 78: 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 317.
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 317.
	Offset	The number of bytes into the RX stream/frame to begin the pattern comparison.	

Table 78: 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 310.
	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 as per PCI specifications.
	Alternate_Clock_Sour ce	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 475.
	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)" on page 280
	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) (see "Power Management Control/Status Register (offset: 0x4C)" on page 280). 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)" on page 283). 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 310, 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 317. 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 317. 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 317. 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 310 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 310. 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 310. 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)" on page 280). The receive CPU will be stopped, and the clocking circuitry disabled.
16. Set the Enable_Alternate_Clock bit in the PCI Clock_Control register. 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 323" and put it in promiscuous mode by setting the Promiscuous Mode bit of "Receive MAC Mode Register (offset: 0x468)" on page 323.
19. Enable the PME bit in the PCI Power Management Control/Status Register (offset: 0x4C) (see "Power Management Control/Status Register (offset: 0x4C)" on page 280). The Ethernet controller asserts PME to wake up the system. Set the Power_State bits to D3 in the PCI Power Management Control/Status Register (offset: 0x4C).

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 582.

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 320). 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 79: 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 329) 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 80 shows the pause quanta based on the Enable_Long_Pause bit setting.

Table 80: 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 81). 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 81: Keep_Pause Recommended Value

Register.Bit	*Recommended Value*
Receive_MAC_Mode_Control.Keep_Pause	DISABLED

Statistics Block

The statistic block shown in Table 82 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 582).

Table 82: 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 BCM5718 family 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 82 lists the flow control registers in the Ethernet controllers.

Table 83: 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 323.
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 320.
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 329.

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 11: Interrupt Processing

NetXtreme Legacy Interrupt Model

For reference, this section reviews the legacy NetXtreme interrupt model.

When the controller completes a transmit or a receive event, 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 controller, and which receive buffer descriptors (Rx BDs) have been consumed by a newly received packet. Normally, host software checks this status block whenever an interrupt is generated. In addition, host software could also poll the status block to determine whether it had been updated by the hardware since the last time it read the status block (this is called during interrupt processing). The legacy status block format is shown in Table 84.

Whenever the controller updates the status block, it decides whether to assert the interrupt line (INTA#). If MSI were enabled, the controller would DMA the MSI data DWORD instead of asserting a line interrupt (or, in the case of PCIe, instead of sending an assert interrupt message).

The controller has interrupt avoidance mechanisms ("host interrupt coalescing") that allow the host to instruct the controller not to generate an interrupt every time it writes a status block into host memory. In addition, it has mechanisms that allow host software to control when and how often the status block is updated in host memory. Since the status block updates and interrupt generation need not happen one-to-one, the following mechanism is in place to communicate to host software if the status block was updated since it was last read by the host — essentially avoiding race conditions:

1. The controller DMAs the status block into host memory before a line interrupt or MSI is generated.
2. The host interrupt service routine (ISR) reads an "update bit" at the top of the status block and checks whether this bit is set to 1.
3. When set to 1, the update bit of the status block indicates to host that the status block has been refreshed by the controller.
4. The ISR must then write a zero to clear/deassert this bit to dirty the status block, and then the ISR may proceed to read the updated producer/consumer index pointers, etc.
5. If the update bit is not set to 1, the interrupt may be considered as spurious and the ISR may wish to abort.

This mechanism allows host software to determine if the status block has been updated. Due to various platform-dependent asynchronous timing issues, an ISR may occasionally see stale status block data. In this case, 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.

Table 84: NetXtreme Legacy Status Block Format

Offset	*3116*	*150*	
0x00	Status Word		
0x04	[31:8] Reserved 0x0		[7:0]Status Tag

Table 84: NetXtreme Legacy Status Block Format (Cont.)

Offset	*3116*	*150*
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

Legacy Status Word Format:

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change
- Bit [2]: Error/Attention
- Bit [3]: Resvd–always 0
- Bit [4]: Resvd–always 0
- Bit [5]: Resvd–always 0
- Bits [31:6]: Reserved 0x0

ISR Flow

The basic flow of an ISR is as follows:

1. Acknowledge interrupt. Write a nonzero value (i.e., value = 1) to interrupt mailbox 0 (see Interrupt Mailbox 0 Register 0x200) for host standard and flat modes, and other Interrupt Mailbox Registers 0x5800-0x40 for indirect mode, to indicate to the controller that the driver is currently processing the interrupt. This step temporarily disables further device interrupts.
2. Read and save the value of the status tag field of the status block.
3. Claim the interrupt. Determine if action is required. Read the updated bit of the status word. If the update bit is asserted then the controller has updated the status block.
4. Clear the update bit of the status word in the status block. This indicates that the host driver either has or will touch the status block. If a during interrupt event occurs, the host driver can examine the update bit later to determine if a fresh status block has been moved to host memory space since.
5. Check for Rx traffic indications:
   a. Loop through enabled Rx Return Rings (0 to 3).
   b. Check for differences between Rx return ring producer indices (status block) and Rx return ring consumer indices (value written to mailbox on previous call) as this indicates the number of Rx frames to process from the Rx return ring.
   c. Process the Rx packets.
   d. Update the Rx return ring consumer indices in each mailbox.
6. Check for Tx completions:

a. Check for differences between previous consumer index (tracked by host driver) and current consumer index in the status block. These are the Tx BDs which can be made available again to the next send operation.

b. 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 the status block (i.e., flush status blocks cached by intermediate PCI bridges).

8. Check the update bit in the status word of the status block. If the update bit is asserted, then new data has been DMAed to the host.

   a. Repeat steps 5 and 6.

9. Check error bit in status word. The driver may check the state machine/FTQ status registers for various attentions.

10. Reenable interrupts. When the "status tagged" status mode bit of the miscellaneous host control register (see "Miscellaneous Host Control Register (offset: 0x68)" on page 283) is set to 1, 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 finished processing Rx and Tx. Otherwise write 0 to Interrupt Mailbox 0 register. This step also clears existing interrupts.

Legacy Status TAGGING Mode

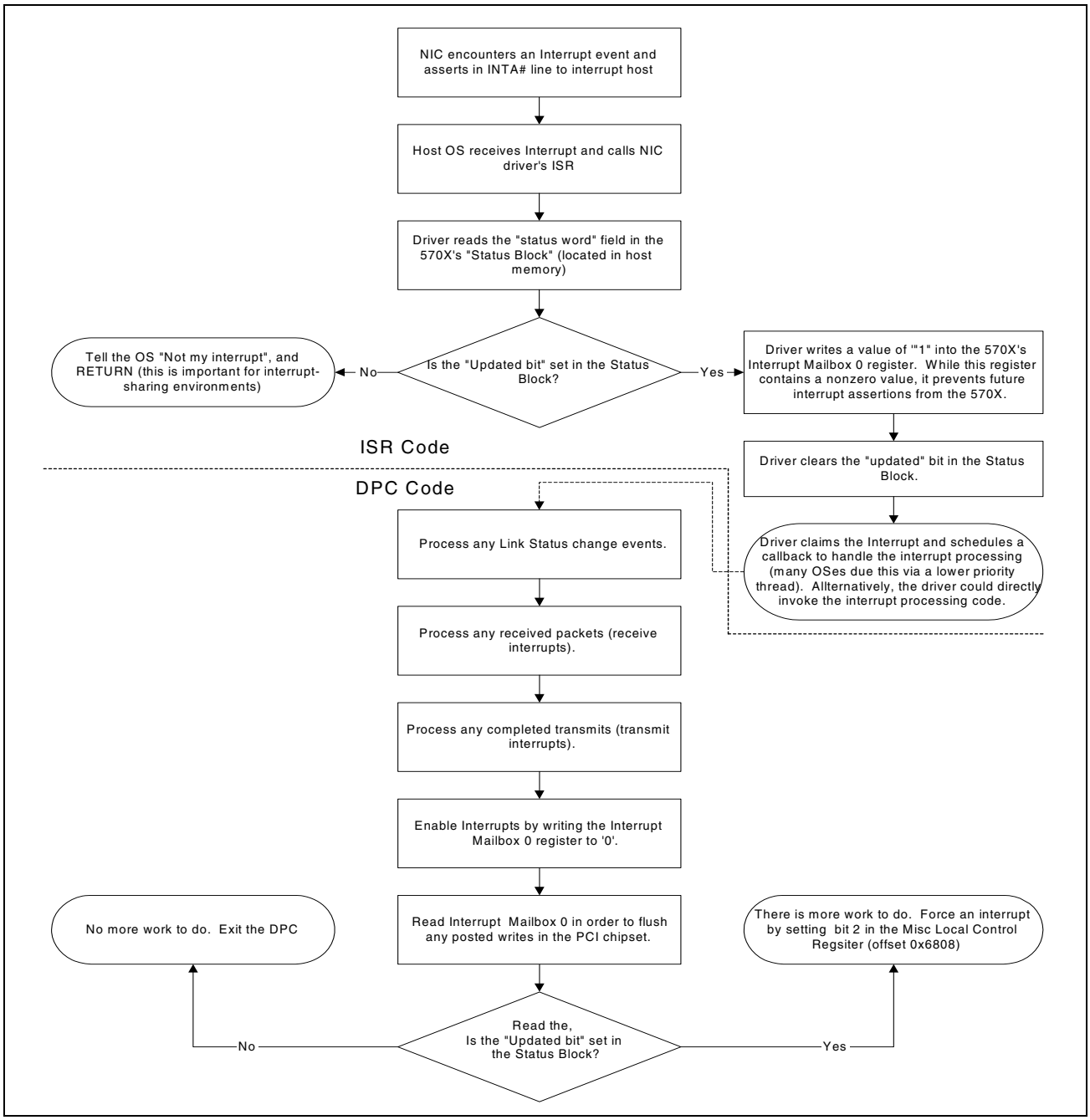
This mode is enabled by setting the "status tagged" mode bit of the miscellaneous host control register (0x68). When enabled, a unique 8-bit tag value is inserted into the status block status tag at location [7:0]. The status tag can be returned to the [31:24] field of the INT mailbox register by the driver. When the mailbox register field [23:0] is written with a zero value, the tag field of the mailbox 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, then the controller triggers another interrupt immediately.

Basic Driver Interrupt Processing Flow

Flowchart for Servicing an Interrupt

The following figure shows the basic driver interrupt service routine flow.

Figure 53: Basic Driver Interrupt Service Routine Flow



Interrupt Procedure

1. Acknowledge interrupt. Write a nonzero value (i.e., value = 1) to the interrupt mailbox 0 (see registers 0x270–0x304) 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 83).
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) for host standard and "Interrupt Mailbox 0 Register (offset: 0x5800)" 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)" on page 283) 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.

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 software 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 software 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 85).

Table 85: 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 INTA bit	"Miscellaneous Host Control Register (offset: 0x68)" on page 283.
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 475.
Interrupt Mailbox 0 register	"Interrupt Mailbox 0 Register (offset: 0x5800)" on page 463
Receive Coalescing Ticks register	."Receive Coalescing Ticks Register (offset: 0x3C08)" on page 416
Send Coalescing Ticks register	."Send Coalescing Ticks Register (offset: 0x3C0C)" on page 417
Receive Max Coalesced BD Count register	."Receive Max Coalesced BD Count Register (offset: 0x3C10)" on page 418
Send Max Coalesced BD Count register	"Send Max Coalesced BD Count Register (offset: 0x3C14)" on page 420.
Mode Control register. • Interrupt on Flow Attention (Bit 28) causes a host interrupt when an enabled flow attention occurs • Interrupt on DMA Attention (Bit 27) causes a host interrupt when an enabled DMA attention occurs • Interrupt on MAC Attention (Bit 26) causes a host interrupt when an enabled MAC attention occurs • Interrupt on RX RISC Attention (Bit 25) causes a host interrupt when an enabled RX-RISC attention occurs	"Mode Control Register (offset: 0x6800)" on page 472

MSI

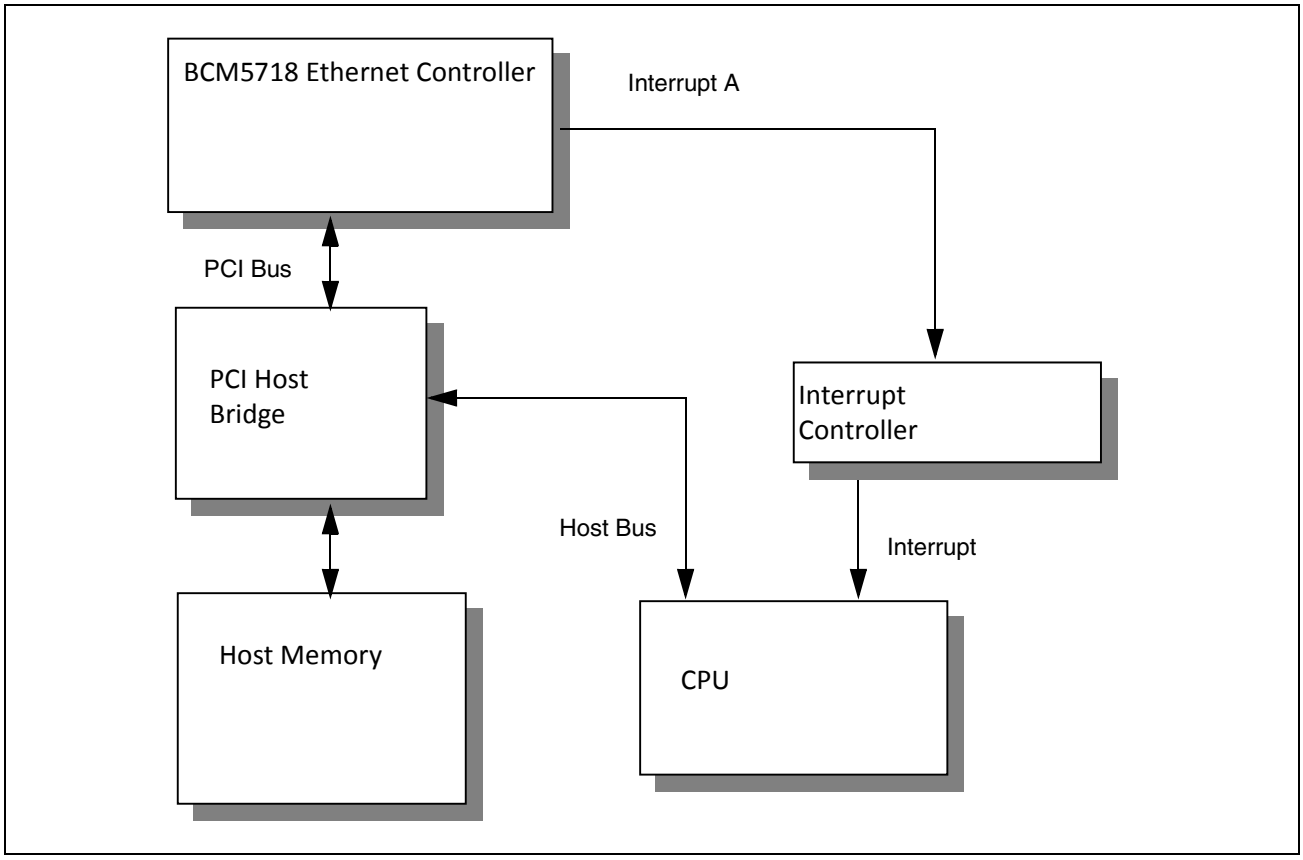
PCI Specification 2.2 defines a new mechanism for a device to request services by its device driver. It is called Message Signaled Interrupt (MSI). MSI will eventually deprecate the traditional interrupt mechanism. In MSI, device DMAs a specified DWORD data to a specified host address if it needs to 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 54.

Figure 54: 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:

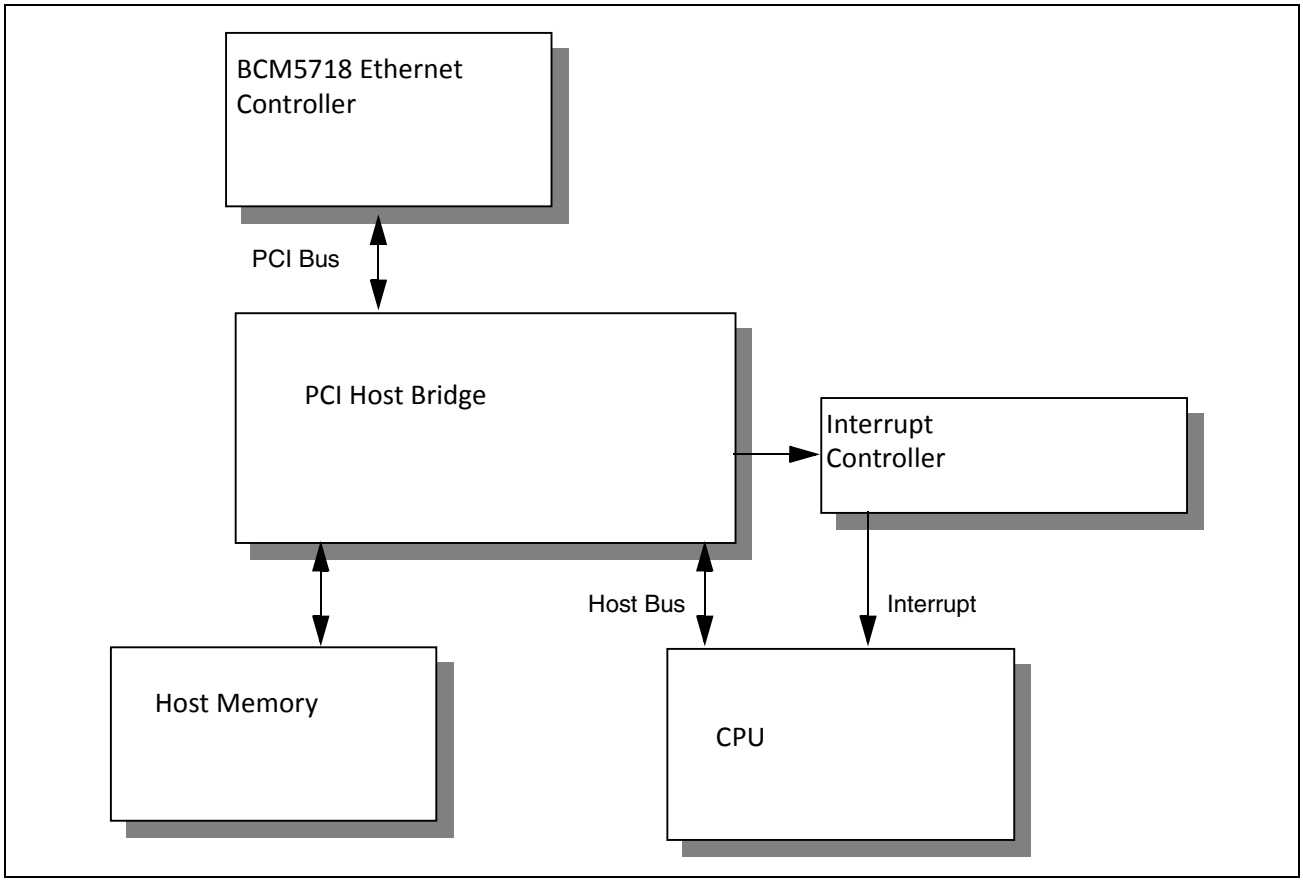
- DMAs data of received packets to the host.
- DMAs receive buffer descriptors to Receive Return Ring in the host memory.
- DMAs status block to the host memory.
- Generates an interrupt to request its device driver for processing.

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 55.

Figure 55: 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.

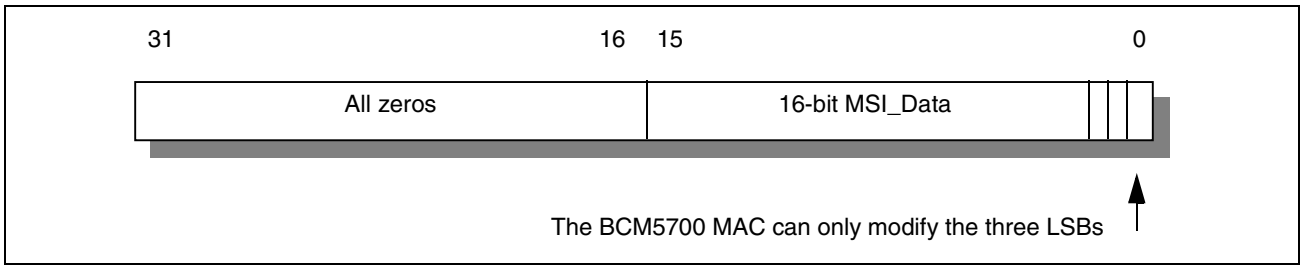
MSI Address

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 56.

Figure 56: 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

The BCM5718 family introduces PCIe™-compliant message signal interrupts (MSI-X), supporting a maximum of 17 vectors (BCM5718).

This is accomplished without disturbing the legacy interrupt system model. The introduction of interrupt vectoring provides per-receive-queue indication and per-transmit-queue completion to the device driver. The maximum number of MSI-X vectors attainable is indirectly determined by certain controller modes of operation.

The existence of multiple receive queues (Rx return rings) in the BCM5718 family stems either from the Receive Side Scaling (RSS) feature or from the I/O Virtualization (IOV) feature.

Note: RSS and IOV are mutually exclusive — only one of these features may be enabled in the BCM5718 family at any time.

When both RSS and IOV are disabled, all of the received packets are posted in the default receive queue (Rx Return Ring 0). This default Rx return ring is sometimes referred to as "the Receive Return Ring". The number of transmit queues is also limited to one in this use case.

In RSS enabled mode, receive traffic classification (also known as sorting) is done purely based on the RSS hash lookup table. Four RSS receive queues and a single transmit queue exist in this mode for the controller. Therefore, up to 5 MSI-X vectors are offered in RSS mode.

In IOV mode, receive traffic classification is performed by VRQ filters. Sorted Rx traffic is routed among 17 Virtual Receive queues (VRQs). The transmit side offers up to 16 transmit queues. Therefore, while in IOV mode, there is an opportunity to map packet indication and completions into 17 MSI-X vectors.

The MSI-X specification allows a device to advertise the availability of a chosen number of vectors in its PCIe capabilities list; however, it does not specify a mechanism for an operating system (OS) to negotiate the number of vectors down to the number it would actually want to use. So, one simple way for an OS to allocate fewer vectors than a device has asked for is to fill out all of the vector table entries advertised by a device. In such an event, the device would first need to identify the situation, and then reassign or regroup the internal interrupt sources into the limited number of table entries allocated by the OS. But, the device hardware would not possess all of the information necessary to accomplish either task. This means that device driver involvement is required. In the BCM5718 family, it has been assumed that, in most cases, an OS would allocate the requested number of vectors. A deviation from that would be considered an exception; in that case, because support for MSI-X has already been advertised to the BIOS, there would be no way to fall back to INTx mode. This situation could be handled by forcing everything into MSI-X Vector#0 and not using other vectors, disregarding however many of the originally requested vectors the OS actually allocated.

Thus, the BCM5718 family offers two vector modes within the MSI-X mode:

- Single Vector mode
- Multivector mode

Each of these two modes in turn offers the following submodes:

- Single Vector mode (Restrict to Vector#0)

  - Single Vector RSS mode
  - Single Vector IOV mode
- Multivector mode
  - Multivector RSS mode (5 Vectors Requested)
  - Multivector IOV mode (17 Vectors Requested)

The idea is that during boot up, if either RSS or IOV mode is to be enabled, the driver would program the controller into the appropriate Multivector submode first. Subsequently, if the OS does not allocate the requested number of MSI-X vectors (five for RSS and 17 for IOV), then the driver must reprogram the controller into 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.

Single Vector mode or Multivector mode may be chosen by the driver by programming register bit 0x6000[7]. Such submodes are derived by the controller from the appropriate RSS, IOV and Multiple Send queue mode settings. All permissible combinations are shown in the table below.

Note: IOV mode enables 16 VRQs but does not necessarily enable multiple Tx queues, which may be enabled independent of IOV-mode selection. Hence the send completion, when Single Send queue is chosen, will always be indicated on Vector#0 while Multiqueue Send Completions will be paired with Rx queue indications in the respective vectors.

Table 86: MSI-X Vector Mode Selection

MSI-X Vector Mode		***Multivector Enable 0x6000[7]***	***RSS Mode Enable 0x468[23]***	***IOV Mode Enable 0x6800[8]***	***Multi TXQ Enable 0x1800[5]***
Single Vector — RSS Mode	Single TXQ	0x0	0x1	0x0	0x0
	Multiple TXQ	0x0	0x1	0x0	0x1
Single Vector — IOV Mode	Single TXQ	0x0	0x0	0x1	0x0
	Multiple TXQ	0x0	0x0	0x1	0x1
Multivector — RSS Mode	Single TXQ	0x1	0x1	0x0	0x0
	Multiple TXQ	0x1	0x1	0x0	0x1
Multivector — IOV Mode	Single TXQ	0x1	0x0	0x1	0x0
	Multiple TXQ	0x1	0x0	0x1	0x1

Vector allocation in each mode is shown below:

Single Vector Mode:

- RSS mode: Vector#0–Aggregate of the following:
- Rx Return Ring Indication (Active only when RSS is Disabled)
- Send Completion
- Link Status Change
- Error/Attention

  - Rx Return Ring3 through Ring0 Indications
- IOV mode: Vector#0 — Aggregate of the following:
  - All 16 Send queue Completions
  - All 17 VRQ Indications
  - VRQ Active Bit-Map
  - Link Status Change
  - Error/Attention

Multivector Mode:

- RSS (5 Vector) mode: Vector allocation would be as follows:
  - Vector#0–Aggregate of the following:
    - Send Completion (Only in Single TXQ mode)
    - Link Status Change
    - Error/Attention
  - Vector#1–RSS Return Ring 0 Indication/TXQ 1 Completion (in Multi TXQ mode)
  - Vector#2–RSS Return Ring 1 Indication/TXQ 2 Completion (in Multi TXQ mode)
  - Vector#3–RSS Return Ring 2 Indication/TXQ 3 Completion (in Multi TXQ mode)
  - Vector#4–RSS Return Ring 3 Indication/TXQ 4 Completion (in Multi TXQ mode)
- IOV (17 Vector) mode: Vector allocation would be as follows
  - Vector#0 — Aggregate of the following:
    - VRQ0 Indication (Default VRQ)
    - Send Completion (Only in Single TXQ mode)
    - Link Status Change
    - Error/Attention
    - VRQ Active Bit-Map
  - Vector#1 — Aggregate of the following:
    - VRQ1 Indication
    - Transmit queue 1 Completion (in Multi TXQ mode)
    - Link Status Change
  - Vector#2–Aggregate of the following
    - VRQ2 Indication
    - Transmit queue 2 Completion (in Multi TXQ mode)
    - Link Status Change
    .
    .
    .
  - Vector#16–Aggregate of the following
    - VRQ16 Indication
    - Transmit queue 16 Completion (in Multi TXQ mode)
    - Link Status Change

In the case of Multivector selection, it is not mandatory to enable all four RSS queues or enable all VRQs/send queues. So, in the case of Multivector, RSS vector#1 through vector#4 are useful only in conjunction with the respective number of RSS queues or CPU# enable. For example, if only three receive queues are receiving traffic, and the Multivector mode is selected by the driver, then even though vectors 0, 1, 2, and 3 remain active, vector#4 never triggers. Hence, in Multivector RSS mode the receive queues are hard-mapped 1:1 with MSI-X vectors 1 through 4. Therefore it is not permissible to regroup receive queues into vectors in an arbitrary manner. This policy also applies to Multivector IOV mode.

Consequently, Multivector mode must only be selected in conjunction with the enabling of either RSS or IOV.

Note: There is no known use case for enabling Multiple Tx queues (TSS) without enabling either RSS or IOV. Therefore, doing so is not permitted.

In Single Vector mode, all receive traffic is indicated via Vector#0. In RSS mode, all four receive queue indications, and all four Send Completions are thus grouped together in Vector#0. While in IOV mode, all 17 Rx queues and all 16 Tx queues are grouped into Vector#0. Also, this mode must be used when RSS and IOV are both disabled (but MSI-X is enabled, although it is not anticipated that doing this has any real-world usefulness). There could be instances when the OS enables RSS but allocates only one MSI-X vector to our device. The Broadcom driver would resort to choosing Single Vector mode in such a scenario.

Note: When MSI-X is disabled, INTx or MSI continue to function as in legacy device implementations. The data structures used in INTx and MSI mode are identical to those in Single Vector MSI-X mode.

The BCM5718 family offers two product SKUs. The MSI-X table size requested by each differs:

- BCM5717: This controller advertises MSI-X table size = 5.
- BCM5718: This controller advertises an MSI-X table size = 17 by default; however, an NVRAM Configuration option is be provided to limit the Table Size to 5 if desired.

MSI-X Plumbing

The basic interrupt/ISR model for NetXtreme controllers remains largely unchanged even with the introduction of vectored interrupts in the form of MSI-X. However, there are four basic design changes:

- Replication of status blocks, mailboxes, etc. This essentially scales the legacy model from one vector to five or 17 vectors.
- Implement MSI-X capability structures in PCIe configuration space.
- Implement PCI-mandated MSI-X data structures within each MAC core, including new base address register (BAR) decoder, MSI-X table, pending bit array (PBA) structure, etc.
- Add per Rx queue host coalescing attributes.

Each item is discussed in detail in the following sections.

Replication of Status Blocks and INT Mailboxes

As shown in Table 87 on page 245, all four Rx Return Consumer Indices are indicated in the legacy Status Block. When MSI-X Multivector RSS mode is enabled, it leads to logically five interrupt vectors. Each of these vectors is bound to its own status block; thus, there are five different status blocks, numbered *Status-Block0* through *Status-Block4*. Each of these five vectors is also bound to its own INT Mail-Box registers.

Each Status Block has a fixed location in the Host Memory. Hence, there are five Status Block Host Address Registers (64-bit). Table 87 shows all five Status-Block Host Address Registers and INT Mail-Box Registers.

Similarly, in case of MSI-X Multivector IOV mode, there are 17 sets of Status Blocks and MailBox registers.

Table 87: MIS-X Status-Block and Mail Box Addresses

		IOV Mode		RSS Mode		
Status Block Number	***Status Block Host Address Register (64-bit)***	***INT Mail Box Register Address***	***Indication Items***	***INT Mail Box Register Address***	***Indication Items***	***Comments***
Legacy	0x3C3C, 0x3C38	0x200 (*)	ALL	0x200(*)	ALL	Legacy Status Block. Used by INTx or MSI. Also used in MSI-X Single-Vector RSS mode or IOV mode

Table 87: MIS-X Status-Block and Mail Box Addresses (Cont.)

Status Block Number	Status Block Host Address Register (64-bit)	IOV Mode		RSS Mode		Comments
		INT Mail Box Register Address	Indication Items	INT Mail Box Register Address	Indication Items	
0	0x3C3C, 0x3C38	0x200	VRQ 0 RR Prod Index Error/Attention Single SBD Cons Index Link Status RBD 0 Cons Indexes VRQ Active Bit-Map	0x200 (*)	Link-Status change Error/Attention Single SBD Cons Index StdRBD Cons Indx JmbRBD Cons Indx	Used only in MSI-X Multivector RSS mode or Multivector IOV mode for Vector#0 -- Vector#2.
1	0x3D00, 0x3D04	0x208	VRQ1 RR Prod Index SBD1 Cons Index RBD 1 Cons Index	0x208 (*)	Rx Return Ring 0 Prod Index SBD1 Cons Index	
2	0x3D08, 0x3D0C	0x210	VRQ2 RR Prod Index SBD2 Cons Index RBD 2 Cons Index	0x210 (*)	Rx Return Ring 1 Prod Index SBD2 Cons Index	
3	0x3D10, 0x3D14	0x218	VRQ3 RR Prod Index SBD3 Cons Index RBD 3 Cons Index	0x218 (*)	Rx Return Ring 2 Prod Index SBD3 Cons Index	Used only in MSI-X Multivector RSS mode or Multivector IOV mode for Vector#3 -- Vector#4.
4	0x3D18, 0x3D1C	0x220	VRQ4 RR Prod Index SBD4 Cons Index 'RBD 4 Cons Index	0x220 (*)	Rx Return Ring 3 Prod Index SBD4 Cons Index	
5	0x3D20, 0x3D24	0x228	VRQ5 RR Prod Index SBD5 Cons Index RBD 5 Cons Index	N/A	N/A	Used in MSI-X Multivector IOV mode for Vector#5 -- Vector#16
6	0x3D28, 0x3D2C	0x22C	VRQ6 RR Prod Index SBD6 Cons Index RBD 6 Cons Index	N/A	N/A	
7	0x3D30, 0x3D34	0x230	VRQ7 RR Prod Index SBD7 Cons Index RBD 7 Cons Indexes	N/A	N/A	

Table 87: MIS-X Status-Block and Mail Box Addresses (Cont.)

Status Block Number	Status Block Host Address Register (64-bit)	IOV Mode		RSS Mode		Comments
		INT Mail Box Register Address	Indication Items	INT Mail Box Register Address	Indication Items	
……. …….	…….. ……..	…….. ……..	……. …….	N/A	N/A	–
16	0x3D78, 0x3D7C	0x254	VRQ16 RR Prod Index SBD16 Cons Index RBD 16 Cons Index	N/A	N/A	–

Each replicated Status Block has its own format; however, each of the new status block formats is extrapolated from the legacy status block format.

Note: Although High Priority INT Mail Boxes are DWORD (32-bit) registers, the original four were placed on QWORD (64-bit) boundaries for legacy PCI 64-bit target access purposes. Broadcom's PCIe does not allow 64-bit target accesses. Thus, the new Mailboxes are being placed 32-bits apart; however, we are keeping the original four addresses intact.

Single-Vector RSS Mode Status Block Format

Table 88 shows the Status-Block format used by Vector#0 in the Single-Vector RSS mode. Note that first 24B of this structure is identical to the legacy Status Block.

Table 88: Status Block Format (MSI-X Single-Vector RSS Mode)

Offset	3116	150
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 [1] Consumer Index (Acts as Single Send queue Cons Index)	Receive Jumbo Producer Ring Consumer Index
0x14	Send BD 2 Consumer Index (IF Multi SendQ)	Receive Return Ring 0 Producer Index
0x18	Send BD 3 Consumer Index (IF Multi SendQ)	Send BD 4 Consumer Index (IF Multi SendQ)

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

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change

- Bit [2]: Error/Attention
- Bits [31:4]: Reserved 0x0

Single-Vector IOV Mode Status Block Format

In Single-Vector IOV mode, all events are reported in the same Status Block used by Vector#0.

Table 89: Status Block format (MSI-X Single-Vector IOV Mode)

Offset	*3116*	*150*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0]Status Tag
0x08	Single Send BD Consumer Index (IF Single Send queue ELSE 0x0)	Receive Return Ring 0 Producer Index
0x0C	Send BD 1 Consumer Index (IF Multi SendQ)	Receive Return Ring 1 Producer Index
0x10	Send BD 2 Consumer Index (IF Multi SendQ)	Receive Return Ring 2 Producer Index
0x14	Send BD 3 Consumer Index (IF Multi SendQ)	Receive Return Ring 3 Producer Index
0x18	Send BD 4 Consumer Index (IF Multi SendQ)	Receive Return Ring 4 Producer Index
0x1C	Send BD 5 Consumer Index (IF Multi SendQ)	Receive Return Ring 5 Producer Index
0x20	Send BD 6 Consumer Index (IF Multi SendQ)	Receive Return Ring 6 Producer Index
0x24	Send BD 7 Consumer Index (IF Multi SendQ)	Receive Return Ring 7 Producer Index
……	……	……
0x48	Send BD 16 Consumer Index (IF Multi SendQ)	Receive Return Ring 16 Producer Index
0x4C	Standard RBD Ring 0 Consumer Index	Jumbo RBD Ring 0 Consumer Index
0x50	Standard RBD Ring 1 Consumer Index	Jumbo RBD Ring 1 Consumer Index
0x54	Standard RBD Ring 2 Consumer Index	Jumbo RBD Ring 2 Consumer Index
0x58	Standard RBD Ring 3 Consumer Index	Jumbo RBD Ring 3 Consumer Index
……..	……	……
0x8C	Standard RBD Ring 16 Consumer Index	Jumbo RBD Ring 16 Consumer Index

Status-Block Status Word Format (Single-Vector IOV):

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change
- Bit [2]: Error/Attention
- Bits[5:3]: Resvd–always 0x0
- Bit [6]: Change in VRQ Active Bit Map
- Bits [14:7]: Resvd–always 0x0
- Bit[15]: VRQ Active Bit-Map[16]
- Bits [31:16]: VRQ Active Bit-Map[15:0]

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 Table 90.

Table 90: Status Block [0] Format (MSI-X Multivector RSS Mode]

offset	*3116*	*150*
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	Single Send BD Consumer Index (IF Single Send queue ELSE 0x0)	Reserved 0x0
0x14	Reserved 0x0	Receive Jumbo Producer Ring Consumer Index

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

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change
- Bit [2]: Error/Attention
- Bit [3]: Resvd–always 0
- Bit [4]: Resvd–always 0
- Bit [5]: Resvd–always 0
- Bits [31:6]: Reserved 0x0

Table 91: Status Block [1 N ≤ 4] Formats (MSI-X Multivector RSS Mode)

offset	*3116*	*150*
0x00	Status Word {Valid for all Status Blocks}	
0x04	[31:8] Reserved 0x0	[7:0] Status Tag[n] {Independent for each Status Blocks}
0x08	Reserved 0x0	Receive Return Ring 1 Producer Index {Valid only for Status Block2 else Rsvd 0x0}
0x0C	Receive Return Ring 2 Producer Index {Valid only for Status Block3 else Rsvd 0x0}	Receive Return Ring 3 Producer Index {Valid only for Status Block4 else Rsvd 0x0}
0x10	Send BD [1 ≤ N ≤ 4] Consumer Index (IF Multi Send queue Enabled, ELSE 0x0)	Receive Return Ring 0 Producer Index {Valid only for Status Block1 else Rsvd 0x0}
0x14	Reserved 0x0	Reserved 0x0

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

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

Multivector IOV Mode Status Block Format

Two slightly varying Status Block formats are used in this mode: one format for Status Block#0 and the other for Status Block#1 through Status Block#16, as shown in Table 92.

Table 92: Status Block [0] Format (MSI-X Multivector IOV Mode)

offset	*3116*	*150*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0] Status Tag
0x08	RBD0 Standard Producer Ring Consumer Index	Reserved 0x0
0x0C	Reserved 0x0	Reserved 0x0
0x10	Single Send BD Consumer Index (IF Single Send queue ELSE 0x0)	Receive Return Ring 0 Producer Index
0x14	Reserved 0x0	RBD0 Jumbo Producer Ring Consumer Index

Status-Block [0] Status Word Format (Multivector IOV):

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change
- Bit [2]: Error/Attention
- Bits[5:3]: Resvd–always 0x0
- Bit [6]: Change in VRQ Active Bit Map
- Bits [14:7]: Resvd–always 0x0
- Bit[15]: VRQ Active Bit-Map[16]
- Bits [31:16]: VRQ Active Bit-Map[15:0]

Table 93: Status Block [1 ≥ N ≤ 16] Format (MSI-X Multivector IOV Mode)

offset	*3116*	*150*
0x00	Status Word	
0x04	[31:8] Reserved 0x0	[7:0] Status Tag
0x08	RBD[N] Standard Producer Ring Consumer Index	Reserved 0x0
0x0C	Reserved 0x0	Reserved 0x0
0x10	Send BD [N] Consumer Index (IF Multiple Send queue ELSE 0x0)	Receive Return Ring [N] Producer Index
0x14	Reserved 0x0	RBD [N] Jumbo Producer Ring Consumer Index

Status-Block [1–16] Status Word Format (Multivector IOV):

- Bit [0]: Update-Bit
- Bit [1]: Link Status Change
- Bits [31:1]: Resvd–always 0x0

MSI-X Capability Structure

MSI-X requires that its device-resident data structures or registers be addresses over BARs, as opposed to MSI, which declares its device structure addresses directly in related configuration registers. The are certain restrictions related to the BARs:

- If a BAR is shared with MSI-X for other device purposes, the MSI-X region must be isolated within a 4 KB naturally aligned region. Or, the MSI-X structures may be placed in their own captive BARs.
- MSI-X has been placed in it own set of BARs: BAR4 and BAR5 (64-bit).
- Each PCIe Function of the BCM5718 family, that is Function0 and Function1, advertises availability of their BAR4 and BAR5 as MSI-X BARs. This is done via the MSI-X Capability structure.

The MSI-X Capability structure is implemented inside the EP-RC core. It points to two structures that must be implemented inside a device: the MSI-X Table and a Pending Bit Array (PBA). There is also a Message Control Register. The Capability structure is shown in Table 94.

Table 94: MSI-X Capability Structure

3116	***158***	***73***	***20***
Message Control Reg	Next Pointer	Capability ID	–
MSI-X Table Offset			Table BIR
PBA Offset			PBA BIR

The BIR bits point to the BAR registers that a device function uses to base the respective data structures. In the BCM5718 family, both BIRs have a hard-wired value of 0x4, which implies BAR4 and BAR5.

Note: BAR4 and BAR5 must support 8/16- and 32-bit accesses from the host. However, unaligned 16/32-bit access support are not required.

Note: PCI 2.3 specification notes 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

Full specification of MSI-X is available in PCI Specification rev. 2.3; it is not repeated here. The MSI-X Table hosted by BCM5718 family is described here:

- This structure is placed at offset 0x0 pointed by BAR4 and BAR5. The content of the MSI-X Table structure is shown in Table 95.
- Depending on whether IOV mode is selected, the PCIe core advertises either a 17-entry MSI-X table or a 5-entry MSI-X table. The advertisement choice is made only once following POR and cannot be changed afterwards (see the Table Size field). Selection of Single-Vector/Multivector mode does not affect Table Size.
- This table comprises multiple 4 DWORD-long entries and each entry corresponds to one MSI-X vector. Thus, in the BCM5718 family, there is a maximum of 17 such entries.

- Each entry consists of four fields:
  - The Message Address High and Message Address Low fields points 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 BCM5718 family wants to send an interrupt message corresponding to this vector, it writes the value provided by this field into the address pointed by Message Address field(s).
  - Host software is allowed to replicate the same physical host address into multiple entries; that amounts to interrupt vector aliasing. The BCM5718 family can handle such aliasing.
  - Only one bit, bit[0] of the Vector Control field, is implemented by 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 is 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 BCM5718 family is only 4 DWORDs or 128 bits wide, out of which bits [16:0] are useful, while bits [127:17] are reserved for future use. Each PBA bit index corresponds to the respective MSI-X vector# — that is why only 17 bits are implemented in the BCM5718 family. The PBA is placed at the offset 0x120 relative to the addresses pointed by BAR4 and BAR5. See Table 95.
- There is room for expansion of up to 18 MSI-X vectors in future NetXtreme controllers.
- In the Message Control Register, only three fields 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 MAC Core. (Though PCIe allows host software to enable MSI and MSI-X concurrently, albeit erratically, MSI is preempted 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, i.e., 0 through 16 in the BCM5718 family must be masked. If any interrupt vector event occurs while Function Mask is 1, the corresponding Pending bit in PBA must be set. When this bit is 0, the per-vector Mask bits found in each Table Entry determine whether a vector is masked or not.
  - Table Size: This field may declare a value of either 5 or 17. This is accomplished by Boot Code programming the appropriate Private Register of the PCIe core.

Table 95 depicts the implemented address regions of BAR4 and BAR5 in the BCM5718 family. Only two structures are present: the MSI-X Table and the Pending Bit Array in the offsets shown in the table. Any host accesses to the non-implemented addresses in these BARs must be gracefully handled by the controller; i.e., Reads return 0x0 and Writes have no effect whatsoever on the MAC.

Table 95: MSI-X Table and PBA Structures in BCM5718 Family

MSI-X Table Entry#	*DW3 Content (32-bit)*	*DW2 Content (32-bit)*	*DW1 Content (32-bit)*	*DW0 Content (32-bit)*	*BAR4 and BAR5 Offset*
N/A	Reserved Bits [127:17]		(PBA)	Pending Bits[16:0]	0x120
Reserved Entry	…….	…….	…….	…….	0x110
16	Vec#16 Control	Msg#16 Data	Msg#16 Addr H	Msg#16 Addr L	0x100
15	Vec#15 Control	Msg#15 Data	Msg#15 Addr H	Msg#15 Addr L	0xF0
…… ……	…… ……	…… ……	…… ……	…… ……	…… ……

Table 95: MSI-X Table and PBA Structures in BCM5718 Family (Cont.)

MSI-X Table Entry#	*DW3 Content (32-bit)*	*DW2 Content (32-bit)*	*DW1 Content (32-bit)*	*DW0 Content (32-bit)*	*BAR4 and BAR5 Offset*
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 Cognizant Host Coalescing

After sending a complete Transmit packet to the wire, the controller updates the Send BD Ring consumer index internally. Similarly, after moving each full Received packet to the host memory, the controller updates its internal copy of the Rx Return Ring producer index. The controller notifies the host software or the network stack of such consumer and producer index updates by DMAing the Status Block and then sending an interrupt to the Host CPU.

Host software does not necessarily need to get indication for every packet received from the network, nor does it need to get completion for every packet sent to the wire. Heavy interrupt processing would degrade host CPU performance. However, the stack must communicate with application processes in a timely manner to keep up with network bandwidth demanded by the application. Thus, host software must have control in regulating — however coarsely — when and how often it gets interrupts from the controller. The controller, in turn, accumulates update events. When the accumulation reaches a threshold value, as configured by the host software, the controller sends an interrupt. This scheme is known as interrupt coalescing or host coalescing.

Legacy NetXtreme architecture already offers a set of Host Coalescing (HC) Parameters.

Legacy Host Coalescing Parameters

This section lists the legacy set of host coalescing parameters that NetXtreme already offered prior to BCM5718 family.

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 timer is reset to the value of this register and starts counting down after every status block update (regardless of the reason for the status block update). The timer is reset only after status block updates, and is not reset after any given packet is received. When the timer reaches 0, it is considered to be in the expired state. When 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 occur for receive events only if the Receive Max Coalesced BD value is reached. Status block updates for other reasons (e.g., transmit events) 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 generally increases 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 must be very low, and the host is not close to saturation, it is recommended that this register be set to 1.

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 timer is reset to the value of this register and starts counting down after every status block update (regardless of the reason for the status block update). The timer is reset only after status block updates, and is not reset after a transmit event completes. When the timer reaches 0, it is considered to be in the expired state. When the counter is in the expired state, a status block update occurs 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. 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 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) 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 generally increases 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.

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 updates the status block due to a receive event. Whenever the device completes the reception of a packet, it fills in a receive return ring BD, and then increments an internal receive coalesce BD counter. When this internal counter reaches the value in this register, a status block update occurs. This counter is 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 occur for receive packets only via the Receive Coalescing Ticks mechanism. Status block updates for other reasons (e.g., transmit events) 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.

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 updates 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 occurs. This counter is 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 occur for receive packets only via the Send Coalescing Ticks mechanism. Status block updates for other reasons (e.g., receive events) also include any updates to the send indices. For simplicity, if a host wants to get a status block update for every transmitted packet, the host driver could set this register to a value of 1. However, 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 is used to ensure that status block updates due to transmitting packets are not delayed for an infinite amount of time.

Receive Max Coalesced BD Count During Interrupt Register (Offset 0x3c18)

This register has the same attribute as 0x3C10 except that this parameter is active only during the During Interrupt state, which is the state during which the ISR has acknowledged an interrupt by writing a nonzero value to the MailBox register and thus the interrupt is in a masked state. If this parameter triggers (while in During Interrupt state), the controller will DMA the latest Status Block to the host memory, but the interrupt will remain deasserted.

Send Max Coalesced BD Count During Interrupt Register (Offset 0x3c1c)

This register has the same attribute as 0x3C14 except that this parameter is active only during the During Interrupt state, which is the state during which the ISR has acknowledged an interrupt by writing a nonzero value to the MailBox register and thus the interrupt is in a masked state. If this parameter triggers (while in During Interrupt state), the controller DMAs the latest Status Block to the host memory, but the interrupt remains deasserted.

BCM5718 Family Host Coalescing Parameter Sets

The legacy HC Parameter registers do not offer granularity in terms of individual Rx Return queues or individual Tx queues. Instead, in the legacy implementation, all Transmit/Return Rings are grouped together for metering.

The BCM5718 family offers HC parameters or control on a per-Rx-and-Tx-queue basis when and only when MSI-X Multivector mode is chosen. To that end, 16 more sets of Host Coalescing Parameter registers are added. Each such HC Parameter Set comprises of the following registers:

- Receive [n] Coalescing Ticks Register
- Send [n] Coalescing Ticks Register
- Receive [n] Max Coalesced BD Count Register
- Send [n] Max Coalesced BD Count Register

- Receive [n] Max Coalesced BD Count During Interrupt Register
- Send [n] Max Coalesced BD Count During Interrupt Register

Where, n ranges from 1 through 16. The legacy HC Parameter registers are now called HC Parameter Set [0].

Each of these new sets have the same behavior or attribute as defined in "Receive Coalescing Ticks Register (Offset: 0x3c08)" on page 254 and "Receive Max Coalesced Bd Count Register (Offset: 0x3c10)" on page 255, except one aspect: when MSI-X Multivector mode is enabled, each of these sets associate with its respective Status-Block[n]. However, when either MSI-X is disabled or MSI-X Single-Vector mode is enabled, none of these additional sets would exist. Note that, when MSI-X Single-Vector mode is enabled, even though 5 (or 17) vectors are advertised, only Vector#0 remains active.

Note: To further clarify, in Legacy INTx mode, MSI mode, or MSI-X Single Vector mode, all Transmit and Receive queues are metered collectively by HC Parameter Set [0]. Parameter Sets [1–16] do not exist. Only in Multivector MSI-X mode do HC Parameter Sets [1–16] come into existence.

Table 96 summarizes the existence of HC Parameter Sets, and their association to Status-Blocks. An "----" indicates that this register does not exist in this particular mode.

Table 96: MSI-X Host Coalescing Parameters

Valid HC Parameter Register Set	*Invokes Status Block#*	*IOV-Mode + Multiple TXQ (Netqueue/ VMQ+TSS)*		*IOV-Mode + Single TXQ (VMQ)*		*RSS-Mode + Multiple TXQ (TSS)*		*RSS-Mode + Single TXQ (Legacy)*	
		HC Parameter Registers Address	*Indication Items*	*HC Parameter Registers Address*	*Indication Items*	*HC Parameter Registers Address*	*Indication Items*	*HC Parameter Registers Address*	*Indication Item*
RCTR[0]	0	0x3C08		0x3C08		-----		-----	
SCTR[0]		------	VRQ 0	0x3C0C	VRQ 0	-----		0x3C0C	
RMCBCR[0]		0x3C10		0x3C10	TX	-----		-----	TX
SMCBCR[0]		------	LinkStat	0x3C14	LinkStat	-----	LinkStat	0x3C14	LinkStat
RMCBCDIR[0]		0x3C18	Errors	0x3C20	Errors	-----	Errors	-----	Errors
SMCBCDIR[0]		------	VRQ Map	0x3C24	VRQ Map	-----		0x3C24	
RCTR[1]	1	0x3D80		0x3D80		0x3D80		0x3D80	
SCTR[1]		0x3D84	VRQ 1	-----	VRQ 1	0x3D84	RSS 0	-----	RSS 0
RMCBCR[1]		0x3D88	TXQ 1	0x3D88		0x3D88	TXQ 1	0x3D88	
SMCBCR[1]		0x3D8C		-----		0x3D8C		-----	
RMCBCDIR[1]		0x3D90		0x3D90		0x3D90		0x3D90	
SMCBCDIR[1]		0x3D94		-----		0x3D94		-----	
RCTR[2]	2	0x3D98		0x3D98		0x3D98		0x3D98	
SCTR[2]		0x3D9C	VRQ 2	----	VRQ 2	0x3D9C	RSS 1	----	RSS 1
RMCBCR[2]		0x3DA0	TXQ 2	0x3DA0		0x3DA0	TXQ 2	0x3DA0	
SMCBCR[2]		0x3DA4		-----		0x3DA4		-----	
RMCBCDIR[2]		0x3DA8		0x3DA8		0x3DA8		0x3DA8	
SMCBCDIR[2]		0x3DAC		-----		0x3DAC		-----	

Table 96: MSI-X Host Coalescing Parameters (Cont.)

Valid HC Parameter Register Set	*Invokes Status Block#*	*IOV-Mode + Multiple TXQ (Netqueue/ VMQ+TSS)*		*IOV-Mode + Single TXQ (VMQ)*		*RSS-Mode + Multiple TXQ (TSS)*		*RSS-Mode + Single TXQ (Legacy)*	
		HC Parameter Registers Address	*Indication Items*	*HC Parameter Registers Address*	*Indication Items*	*HC Parameter Registers Address*	*Indication Items*	*HC Parameter Registers Address*	*Indication Item*
RCTR[3]	3	0x3DB0		0x3DB0		0x3DB0		0x3DB0	
SCTR[3]		0x3DB4	VRQ 3	-----	VRQ 3	0x3DB4	RSS 2	-----	RSS 2
RMCBCR[3]		0x3DB8	TXQ 3	0x3DB8		0x3DB8	TXQ 3	0x3DB8	
SMCBCR[3]		0x3DBC		-----		0x3DBC		-----	
RMCBCDIR[3]		0x3DC0		0x3DC0		0x3DC0		0x3DC0	
SMCBCDIR[3]		0x3DC4		-----		0x3DC4		-----	
RCTR[4]	4	0x3DC8		0X3DC8		0x3DC8		0X3DC8	
SCTR[4]		0x3DCC	VRQ 4	-----	VRQ 4	0x3DCC	RSS 3	-----	RSS 3
RMCBCR[4]		0x3DD0	TXQ 4	0x3DD0		0x3DD0	TXQ 4	0x3DD0	
SMCBCR[4]		0x3DD4		-----		0x3DD4		-----	
RMCBCDIR[4]		0x3DD8		0x3DD8		0x3DD8		0x3DD8	
SMCBCDIR[4]		0x3DDC		-----		0x3DDC		-----	
RCTR[5]	5	0x3DE0		0x3DE0					
SCTR[5]		0x3DE4	VRQ 5	-----	VRQ 5	N/A	N/A	N/A	N/A
RMCBCR[5]		0x3DE8	TXQ 5	0x3DE8					
SMCBCR[5]		0x3DEC		-----					
RMCBCDIR[5]		0x3DF0		0x3DE0					
SMCBCDIR[5]		0x3DF4		-----					
RCTR[6]	6	0x3EF8		0x3DE8					
SCTR[6]		0x3EFC	VRQ 6	-------	VRQ 6	N/A	N/A	N/A	N/A
RMCBCR[6]		0x3E00	TXQ 6	0x3DF0					
SMCBCR[6]		0x3E04		-------					
RMCBCDIR[6]		0x3E08		0x3DF8					
SMCBCDIR[6]		0x3E0C		-------					
…….	…….	…….	…….	…….	…….				
…….	…….	…….	…….	…….	…….	N/A	N/A	N/A	N/A
RCTR[16]	16	0x3EE8		0x3EE8					
SCTR[16]		0x3EEC	VRQ 16	-------	VRQ 16	N/A	N/A	N/A	N/A
RMCBCR[16]		0x3EF0	TXQ 16	0x3EF0					
SMCBCR[16]		0x3EF4		-------					
RMCBCDIR[16]		0x3EF8		0x3EF8					
SMCBCDIR[16]		0x3EFC		-------					

Abbreviations:

RCTR[n]	RECEIVE COALESCING TICKS REGISTER[n]
SCTR[n]	SEND COALESCING TICKS REGISTER[n]
RMCBCR[n]	RECEIVE MAX COALESCED BD COUNT REGISTER[n]
SMCBCR[n]	SEND MAX COALESCED BD COUNT REGISTER[n]
RMCBCDIR[n]	RECEIVE MAX COALESCED BD COUNT DURING INTERRUPT REGISTER[n]

SMCBCDIR[n]	SEND MAX COALESCED BD COUNT DURING INTERRUPT REGISTER[n]
0x55AA	New Registers in BCM5718 family (legend)

MSI-X One Shot Mode

The BCM5718 family introduces a new method of MSI-X acknowledgement known as the One Shot mode. When this mode is set, an ISR is asked to skip the interrupt acknowledgement step in which it would otherwise write a nonzero value to the respective INT MailBox. This is sensible because MSI-X vector messages are equivalent to edge-triggered non-shared interrupt events; therefore, there is no need for ISR to explicitly acknowledge the event.

One Shot mode is 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 controller hardware stores a nonzero 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 BCM5718 family, this bit retains the same functionality and associates with Status-Block0 and Vector#0 when MSI-X is enabled. Moreover, 16 additional Coalesce Now bits replicate the same function associated to vector numbers 1 through 16. 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 and Multivector mode Enabled)
- 0x3C00[14]: Coalesce vector#2 Now (When MSI-X Enabled and Multivector mode Enabled)
- 0x3C00[15]: Coalesce vector#3 Now (When MSI-X Enabled and Multivector mode Enabled)
- 0x3C00[16]: Coalesce vector#4 Now (When MSI-X Enabled and Multivector mode Enabled)
- 0x3C00[17]: Coalesce vector#5 Now (When MSI-X Enabled and Multivector mode Enabled)
- ……
- 0x3C00[28]: Coalesce vector#16 Now (When MSI-X Enabled and Multivector mode Enabled)

Misc Coalescing Controls

There are a few Host Coalescing controls in the 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, and some do not apply equally. Such controls are listed here 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 is 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. This bit, 0x68[7], 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 register 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 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 may be set to force a status block update without a corresponding interrupt. This feature is associated to Send BDs only; hence, it applies to Vector#0 in MSI-X mode when Multiple Send Queues are not enabled.

Register Transfer Level (RTL) Note: The HC RTL honors the Coal_Now Flag coming from both Send or Receive flow-through queues (FTQs). Rx FTQ never requests it, as the 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 applies equally to vector#0 through vector#16 in Multivector mode.

End of Receive Stream Interrupt

A new kind of forced interrupt is being introduced in the BCM5718 family. 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 the host memory, a hardware FSM checks if the Rx-MBUF is empty (discounting the effects of pre-allocation). If it is empty, hardware starts counting down a count value. While the countdown 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.

This feature may be enabled or disabled by a register bit. The countdown preload value is also programmable. When enabled in conjunction with Multivector MSI-X mode, there is a programmable option to fire either Vector#0 only or all Vectors.

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
Coalesce Now MSI-X Vector# [16–1]	28:13	WC	0x0	Individual Coalesce Now bits associated with MSI-X Vector# 16 through 1. These bits are self-clearing.
As defined in Legacy	12:0			

End Stream Debounce Register (Offset 0x3cd4)

Name	*Bits*	*Access*	*Default Value*	*DESCRIPTION*
As defined in Legacy	31			
Reserved	30:16	RO	0	–

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 de-bounces effects of IPG or short gaps among packets within a burst. The counter counts in Core-Clocks.

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)" on page 283). 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)" on page 283). 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)" on page 283). 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 Mode Register (offset: 0x3C00)" on page 415). 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 Mode Register (offset: 0x3C00)" on page 415). 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 12: IO Virtualization (IOV)

IO Virtualization or IOV is a solution to transparently share a physical IO device among the child/guest OS images in a virtualized environment. The NetXtreme (BCM5718/19/20) offers a cost efficient set of IO Virtualization features. The following list provides the feature highlights:

- Receive Side:
  - RX Traffic sorting over multiple RX queues
  - Hardware assisted RX Packet replication in Driver
  - Universal VLAN stripping (Not per queue based)
  - 24x Perfect Match Addresses Filters
  - One 128-bit MultiCast Addresses Hash Filter
  - 32-Element/31-Set Programmable Protocol Filter (VLAN, TCP etc)
  - 16 Receive Queues + 1 Default Queue + 1 Drop Queue
  - 17 Standard Receive Producer Rings
  - 17 Jumbo Receive Producer Rings
  - 17 Return Rings
  - Per Queue synchronization with Driver
  - RX Packet Header Data Split and copy - for VMQ

Note: No RSS support in IOV Mode.

- Transmit Side:
  - 16 Transmit Queues for use by NetQueue or NDIS-TSS
  - Per Frame Round Robin and Weighted RR scheduling in Hardware
- General:
  - IOV Mode is a static configuration
  - 18 MSI-X vectors - 1 per TX/RX Queue Pair
  - Per Queue nominal Statistics in Hardware
  - No RX Bandwidth Limitation or TX Traffic/Rate shaping

Note: IOV shall operate in a distinct and a static chip-mode. This mode is called the IOV Mode and the chip would need to be configured during boot-up. Once configured, the mode is irreversible unless a hard-reset is asserted to the chip. During boot-up, if the IOV-Mode is not chosen, the chip will operate in the Legacy Mode which is akin to the operation of previous NetXtreme devices.

Data Structure and Register Changes for IOV

The following changes have been made to support IOV:

Mail Box Register Changes

The following mailbox register changes have been made:

- 16 more Standard RBD Producer Index Mail Boxes have been added.
- 16 more Jumbo RBD Producer Index Mail Boxes have been added.
- 13 more Receive Return Ring Consumer Index Mail Boxes have been added.
- 15 more Host Send Producer Index Mail Boxes have been added.

Receive Mail Box Register Changes

16 Standard RBD Producers, 16 Jumbo RBD Producers, and 13 Receive Return Ring Consumer mail boxes have been added to the High Priority Mail Box region (see "RX Mail Box Registers for VRQ" on page 461).

Send Mail Box Register Changes

15 Host Send Producer Index registers have been added to the High Priority Mail Box region (see "Send Mail Box Registers" on page 357).

Ring Control Block Changes

The following ring control block changes have been made:

- 16 more Standard Producer RCBs added
- 16 more Jumbo Producer RCBs added
- 13 more Return Ring RCBs added
- 15 more Send RCBs added

Receive Ring Control Blocks (see "Receive Ring Control Blocks" on page 73).

Send Ring Control Blocks (see "Send Ring Control Blocks" on page 73).

VRQ Statistics

- VRQ Receive Statistics
- VRQ Transmit Statistics

EMAC collects basic statistics on an individual generic VRQ basis. Though the accumulation mechanism remains the same, i.e., these registers are Clear-On-Read, these statistic registers are independent of the aggregated EMAC statistics registers for LAN and APE (see "VRQ Statistics" on page 457).

MSI-X Vectors Changes

The NetXtreme I offers two vector modes within the MSI-X mode for IOV:

- Single Vector IOV mode (Restrict to Vector#0)
- Multivector IOV mode (17 Vectors Requested)

For more information, see "MSI-X" on page 241.

Register Changes

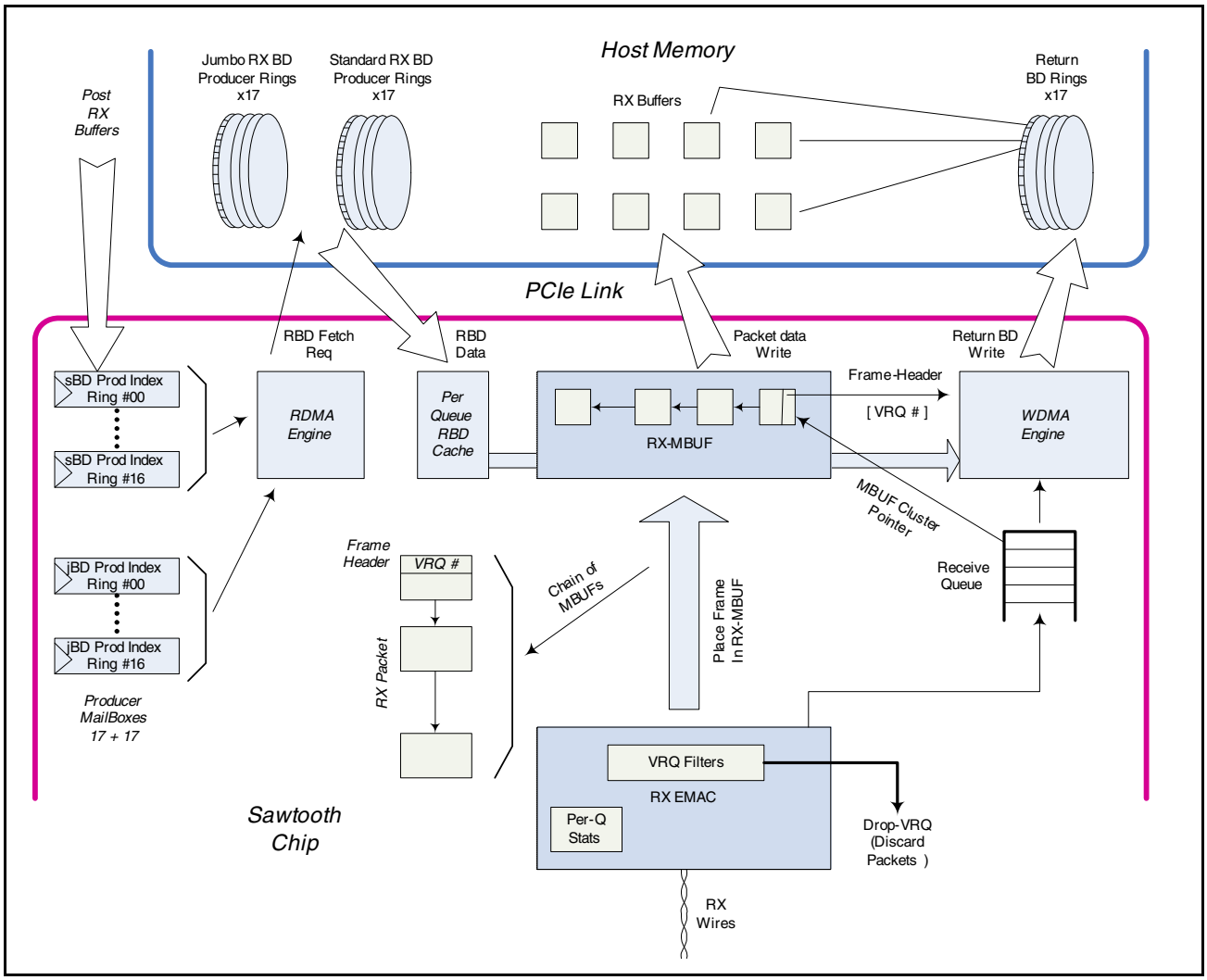
The following register changes have been made:

- GRC-MODE-REG (Offset 0x6800): Additional bits have been added to this existing register (see "Mode Control Register (offset: 0x6800)" on page 472).
- RDI-MODE-REG (Offset 0x2400): Additional bits have been added to this existing register (see "Receive Data and Receive BD Initiator Mode Register (offset: 0x2400)" on page 364).
- Standard Replenish LWM Register (Offset: 0x2D00): This register is meaningful in both Legacy and IOV Modes (see "Standard Replenish LWM Register (offset 0x2D00)" on page 375).
- Jumbo Replenish LWM Register (Offset 0x2D04): This register is meaningful in both Legacy and IOV Modes (see "Jumbo Replenish LWM Register (offset 0x2D04)" on page 376).
- BD Fetch Limit Register (Offset 0x2D08): This register is meaningful only in the IOV-Mode (see "BD Fetch Limit Register (Offset 0x2D08)" on page 377).
- VRQ Status Register (Offset: 0x240C): Additional bits have been added to this existing register (see "VRQ Status Register (offset: 0x240C)" on page 366).
- VRQ Flush Control Register (Offset: 0x2410): Additional bits have been added to this existing register (see "VRQ Flush Control Register (Offset: 0x2410)" on page 366).
- VRQ Flush Timer Register (Offset: 0x2414): The description has been updated on this existing register (see "VRQ Flush Timer Register (offset: 0x2414)" on page 367).
- HC Parameter Set Reset Register (Offset: 0x3C28): This parameter should be placed in this HC block (see "HC Parameter Set Reset Register (Offset: 0x3C28)" on page 423).
- Perfect Match Destination Address Registers: Twenty additional Perfect Match Destination Address Registers are added in RX-EMAC for VRQ Filtering purposes (see "Perfect Match Destination Address Registers" on page 462).
- VRQ Filter Set Registers: The filter block can be programed via a new set of registers (see "VRQ Filter Set Registers" on page 458).
- SEND_BD_INITIATOR_MODE_REG (Offset 0x1800) Additional bits have been added to this existing register (see "Send BD Initiator Mode Register (offset: 0x1800)" on page 355).
- SEND_BD_FETCH_THRESHOLD_REG (Offset: 0x1850) This is a newly added register (see "Send BD Fetch Threshold Register (offset: 0x1850)" on page 357).

IOV - Receive Side

The chosen Receive-Side solution to IOV acceleration is to scale into multiple Receive Indication Queues. These queues are called Virtual Receive Queues or VRQ as defined for NetXtreme internal use. VRQ generally refers to an implementation of both VMQ as well as NetQueue. Each VRQ is provided with dedicated host buffer memory resources. The reason behind this requirement involves memory protection issues with the host platform and OS. Each such queue is also provided with programmable RX traffic filter resources called VRQ Filters. The task of RX-EMAC is to sort the RX traffic designated for multiple VRQs based on VRQ Filter settings. RX-EMAC marks each RX packet with a VRQ number after placing it into the RX-MBUF. Before initiating a RX packet DMA To the host memory, the DMA engine shall inspect each packet's Frame-Header to determine which VRQ it belongs to. It then draws Receive Buffers only from the respective VRQ's host buffer resources and subsequently proceeds to DMA the packet to those buffers. After placing the frame in a VRQ designated host buffer, the controller designates the RX packet selectively to the respective owner (a child/guest OS) of the VRQ and not to any other OS images. Therefore, it implies that each VRQ shall also be associated with an independent receive indication queue, which is nothing but an independent Receive Return Ring (refer to Figure 57 for additional information.

Figure 57: IOV Receive Flow



IOV - Transmit Side

Transmit enhancement for IOV involves adding multiple Host Send Rings or effectively adding multiple transmit queues. At present, Feb 2009, only Netqueue is capable of utilizing multiple TX queues whilst VMQ is not. Although, the Transmit Side Scaling (TSS) feature of NDIS 6.x is capable of using multiple transmit queues. This is the reason this feature, namely 16 Send Rings, could be enabled in BCM5718/19/20 irrespective of the IOV-Mode settings.

BCM5718/19/20 shall implement a limited set of capabilities in this regard:

- A maximum of 16 Send Rings (SBD Rings)
  - Multiple Send Rings could be enumerated only in conjunction to either IOV-Mode or RSS Mode
  - All 16 Send Rings may be enabled in conjunction of IOV Mode
  - Up to 4 Send Rings may be enabled in conjunction of RSS Mode

- Only Host based Rings supported (NIC based Rings de-featured early on)
- Maximum Ring size of 512 each
- 16 High Priority Send Producer Index Mailboxes - one per ring
- 16 Send Ring Control Blocks (RCB)
- 1KB of private on-chip SBD cache per Ring - Total 16KB
- Introduction of a SBD Fetch Threshold
- Send BD Format unchanged in Multi Ring mode.
- No Rate / Traffic Shaping algorithm offered
- Very basic round-robin packet by packet arbitration among 16 Send Rings
- All Send Offload features, namely LSO & Checksum-Offload shall continue to function in all rings without any behavior change.
- Minimal set of per Send Queue EMAC Statistics

The basic Send interface with the Device Driver remains unchanged, only the number of Send Rings are scaled up to 16 from 1. Thus there shall be 16x Send Producer BD Index Mailboxes and 16x Ring Control Block Registers in controller.

SBDs are fetched from the Host memory and are stored in SBD-cache memory internal to the chip. Though such a cache memory could be shared by all 16 Rings, for the sake of simplicity we chose to assign each Send Ring a private partition of the SBD cache - physically it is a single SRAM, but is divided into 16 equal address regions. Each such address range shall serve as a private SBD cache to a Send Ring. The total size of the SBD cache is thus 16 KB.

Section 13: Ethernet Controller Register Definitions

BCM5718 Family Register MAP

The BCM5718 family's internal register map is shown in Table 97.

Table 97: BCM5718 Family Register Map

Block Name	*Block Range*	*Subblock Range*	*Description*
PCIe CFG	0x0000–0x03FF	0x0000–0x03FF	PCIe Configuration Register Shadow
EMAC	0x0400–0x07FF	0x0400–0x06FF 0x0700–0x7FF	EMAC Unused
EMAC STAT	0x0800–0x0BFF	0x0800–0x08FF 0x0900–0x0BFF	EMAC Statistics Unused
SDI	0x0C00–0x0FFF	0x0C00–0x0CF7 0x0CF8–0x0FFF	Send Data Initiator Unused
SDC	0x1000–0x13FF	0x1000–0x100B 0x100C–0x13FF	Send data Completion Unused
SBDS	0x1400–0x17FF	0x1400–0x147F 0x1480–0x17FF	SBDS Registers Unused
SBDI	0x1800–0x1BFF	0x1800–0x1847 0x1848–0x1BFF	Send BD Initiator Unused
SBDC	0x1C00–0x1FFF	0x1C00–0x1C03 0x1C04–0x1FFF	Send BD Completion Unused
RQP	0x2000–0x23FF	0x2000–0x2258 0x2259–0x23FF	Receive List Placement Unused
RDI	0x2400–0x27FF	0x2400–0x24C3 0x24C0–0x27FF	Receive Data Initiator Unused
RDC	0x2800–0x2BFF	0x2800–0x2803 0x2804–0x2BFF	Receive Data Completion Unused
RBDI	0x2C00–0x2FFF	0x2C00–0x2C1B 0x2C1C–0x2FFF	Receive BD Initiator Unused
RBDC	0x3000–0x33FF	0x3000–0x300F 0x3010–0x33FF	Receive BD Completion Unused
CMPU	0x3400–0x37FF	0x3400–0x35FF 0x3600–0x3687	Central Power Management Unit Unused

Table 97: BCM5718 Family Register Map (Cont.)

Block Name	Block Range	Subblock Range	Description
DBU	0x3800–0x3BFF	0x3800–0x3817	Debug Unit (UART)
		0x3900–0x3907	Chip Debug
		0x3908–0x3BFF	Unused
HC	0x3C00–0x3FFF	0x3C00–0x3CC3	Host Coalescing
		0x3CC4–0x3FFF	Unused
MA	0x4000–0x43FF	0x4000–0x400F	Memory Arbiter
		0x4010–0x43FF	Unused
BM	0x4400–0x47FF	0x4400–0x445B	Buffer manager
		0x445C–0x47FF	Unused
DMAR	0x4800–0x4BFF	0x4800–0x4A13	DMA Read
		0x4A14–0x4BFF	Unused
DMAW	0x4C00–0x4FFF	0x4C00–0x4C07	DMA Write
		0x4C08–0x4FFF	Unused
RX-CPU	0x5000–0x53FF	0x5000–0x5037	RX CPU
		0x5038–0x53FF	Unused
Open Block	0x5400–0x57FF	–	–
MB	0x5800–0x5BFF	0x5800–0x5903	Low Priority Mail Box
		0x5903–0x5BFF	Unused
FTQ	0x5C00–0x5FFF	0x5C00–0x5CFF	Flow Through Queue
		0x5D00–0x5FFF	Unused
MSI	0x6000–0x63FF	0x6000–0x6007	Message Signaled Interrupt
		0x6008–0x63FF	Unused
CFG Port	0x6400–0x67FF	0x6400–0x67FF	PCIe Core Private Registers Access to Configuration Space
GRC	0x6800–0x6BFF	0x6800–0x681B	Misc Host Control
		0x6834–0x6843	SEEPROM
		0x6890–0x68A8	Misc Control
		0x68B4–0x68DF	Unused
Rserved	0x6C00–0x6FFF	0x6C00–6C37	Unused
		0x6C38–0x6FFF	Unused
NVM	0x7000–0x73FF	0x7000–0x703B	Non Volatile memory (Flash Controller)
		0x703C–0x73FF	Unused
UART	0x7800–0x7BFF	0x7800–0x781F	Debug UART Modem
		0x7820–0x7BFF	Unused
TL-DL-PL Port	0x7C00–0x7FFF	0x7C00–0x7FFF	PCIe Core Private Register Access to TL, DL & PL
		******	
APE	0x10000–0x18FFF		APE Access
MF	0x19000–0x193FF		Management Filters

PCI Configuration Registers

Device ID and Vendor ID Register (offset: 0x00)

This register is reset by hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Device ID	31:16	RO	–	Default for BCM5717 (LAN Function 0): 0x1655 Default for BCM5718 (LAN Function 0): 0x1656
Vendor ID	15:0	RO	0x14E4	–

Status and Command Register (offset: 0x04)

This register is reset by PCIE Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Detected Parity Error	31	RW2C	0x0	When this bit is set, it indicates that the function has received a poisoned TLP
Signaled System Error	30	RW2C	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	RW2C	0x0	This bit is set when a requester receives a completion with UR completion status
Received Target Abort	28	RW2C	0x0	This bit is set when a requester receives a completion with completer abort completion status.
Signaled Target Abort	27	RW2C	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	RW2C	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.
66 MHz 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.

Name	*Bits*	*Access*	*Default Value*	*Description*
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.
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: 0x08)

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	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 BCM5718 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)

Name	*Bits*	*Access*	*Default Value*	*Description*
BIST	31:24	RO	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 multifunction 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	0x0	This field is implemented by PCIE device as a read/ write field for legacy compatibility purposes.

Base Address Register 1 (offset: 0x10)

Name	Bits	Access	Default Value	Description
Address	31:4	RW	0	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.
Prefetch	3	RO	0x1	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. Path = i_cfg_func.i_cfg_private.
Type	2:1	RO	0x2	These bits indicate that BAR_1 may be programmed to map this adapter to anywhere in the 64-bit address space. Path = i_cfg_func.i_cfg_private.
Space	0	RO	0	This bit indicates that BAR_1 maps a memory space and is always read as 0. Path = i_cfg_func.i_cfg_private.

Base Address Register 2 (offset: 0x14)

Name	Bits	Access	Default Value	Description
Address	31:0	RW	0	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)

Name	Bits	Access	Default Value	Description
Address	31:4	RW	0	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_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.

Name	Bits	Access	Default Value	Description
Prefetch	3	RO	0x1	This bit indicates that the area mapped by BAR_2 may be pre-fetched or cached by the system without side effects. Path = i_cfg_func.i_cfg_private.
Type	2:1	RO	0x2	These bits indicate that BAR_2 may be programmed to map this adapter to anywhere in the 64-bit address space. Path = i_cfg_func.i_cfg_private.
Space	0	RO	0	This bit indicates that BAR_2 maps a memory space and is always read as 0. Path = i_cfg_func.i_cfg_private.

Base Address Register 4 (offset: 0x1c)

Name	Bits	Access	Default Value	Description
Address	31:0	RW	0	These bits set the address upper 32-bit address space. These bits may be combined with the bits in BAR_2 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)

The 32-bit BAR_5 register programs the 3rd base address for the memory space mapped by the card onto the PCI bus. This register can be combined with BAR_4 to make a 64-bit address for supporting Dual Address cycles systems.This register is not needed by Xinan and is expected to be disabled. The register is used by Everest which requires a 2nd BAR. Path = i_cfg_func.i_cfg_public.i_cfg_dec.

Name	Bits	Access	Default Value	Description
Address	31:4	RW	0	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_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.
Prefetch	3	RO	0x1	This bit indicates that the area mapped by BAR_3 may be pre-fetched or cached by the system without side effects. Path = i_cfg_func.i_cfg_private.
Type	2:1	RO	0x2	These bits indicate that BAR_3 may be programmed to map this adapter to anywhere in the 64-bit address space. Path = i_cfg_func.i_cfg_private.
Space	0	RO	0	This bit indicates that BAR_3 maps a memory space and is always read as 0. Path = i_cfg_func.i_cfg_private

Base Address Register 6 (offset: 0x24)

The 32-bit BAR_4 register programs the upper half of the 3nd base address for the memory space mapped by the card onto the PCI bus.

Name	*Bits*	*Access*	*Default Value*	*Description*
Address	31:0	RW	0	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. Path = i_cfg_func.i_cfg_public.i_cfg_dec.

Cardbus CIS Pointer Register (offset: 0x28)

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Cardbus CIS Pointer	31:0	RO	0x0	N/A for PCIE Device

Subsystem ID/Vendor ID Register (offset: 0x2C)

This register is reset by Hard Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Subsystem Device ID	31:16	RO	–	Default for BCM5717 (LAN Function 0): 0x1655 Default for BCM5719 (LAN Function 0): 0x1657 Default for BCM5718 (LAN Function 0): 0x1656 Default for BCM5720 (LAN Function 0): 0x165F
Subsystem Vendor ID	15:0	RO	0x14E4	Identifies board manufacturer

Expansion ROM Base Address Register (offset: 0x30)

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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	RO	0x0	Unused
Capabilities pointer	7:0	RO	0x48	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.

Interrupt Register (offset: 0x3C)

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	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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Indirect Interrupt mail box	63:0	RW	0	Interrupt Mailbox

Power Management Capability Register (offset: 0x48)

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	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)

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 device 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 are 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)

The device driver is prohibited from writing to this register.

Name	*Bits*	*Access*	*Default Value*	*Description*
MSI Control	31:25	RO	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	RW	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	RO	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	RW	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	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)

Name	Bits	Access	Default Value	Description
MSI Lower Address	31:2	RW	Unknown	MSI Lower Address
Reserved	1:0	RO	0	–

MSI Upper Address Register (offset: 0x60)

Name	Bits	Access	Default Value	Description
MSI Upper Address	31:0	RW	Unknown	MSI Upper Address

MSI Data Register (offset: 0x64)

Name	Bits	Access	Default Value	Description
MSI Data	15:0	RW	Unknown	MSI Data

Miscellaneous Host Control Register (offset: 0x68)

Name	Bits	Access	Default Value	Description
ASIC Rev ID	31:28	R	Product ID input	0xF: Indication that BCM5718 family follows new PRODUCT/REV ID mapping
	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
Enable TLP Minor Error Tolerance	15	RW	0	Set this bit to enable TLP minor error tolerance (ATTR/TC/LOCK command)
Log Header Overflow	14	RW	0	Set this bit to enable log header due to overflow

Name	*Bits*	*Access*	*Default Value*	*Description*
Boundary check	13	RW	0	Set this bit to enable crossing 4 KB boundary check
Byte enable Rule Check	12	RW	0	Set this bit to enable the byte enable rule check
Interrupt Check	11	RW	0	Set this bit to enable the interrupt check
RCB Check	10	RW	0	Set this bit to enable RCB check
Enable Tagged Status Mode	9	RW	0	When set, an unique 8-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
Enable Clock Control register read/write capability	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	3	RW	0	Set this bit to enable endian word swapping when accessing through PCIE target interface
Enable Endian Byte Swap	2	RW	0	Set this bit to enable endian byte swapping when accessing through PCIE target interface. ***Note:*** Setting register 0x68 bit 2 (Enable Endian Byte Swap) causes PCI configuration reads to the following registers to become swapped: • 0x40 • 0x68 - 0x9C • 0xF4 - 0xFF This is different behavior from previous NetXtreme controllers. Reference BCM5718 Family errata relating to byte swap control for additional information.
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.
Clear Interrupt	0	WO	0	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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RW	0x0	–
Reserved	28:2	RO	0x00	–
DMA write watermark	21:19	RW	7	Watermark for DMA write. 0: 32B 1: 64B 2: 96B 3:128B 4:160B 5:192B 6:224B 7:256B
Reserved	18:8	RW	0	–
Enable MSIX fix	7	RW	0	Enable msi/msix legacy fix.
DMA read MRRS for slow speed	6:4	RW	0	This setting is for 10/100M Ethernet DMA read MRRS. The pcie_core will accord to this value to be max DMA read length. This configuration 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	3:2	RW	0	–
Disable_64B_cache_alignment	1	RW	0	Disable 64B cache alignment for DMA write to Host memory
Disable_32B_cache_alignment	0	RW	0	Disable 32B cache alignment for DMA write to Host memory

PCI State Register (offset: 0x70)

This register is reset by PCIE Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:20	RO	0x0000	–
Generate reset plus	19	W1 Read 0	0	For func 1 write 1 generates 10 clock wide reset pulse reads always 0 for func 0 reserved
APE Program Space Write Enable	18	RW	0	When this bit is set the APE program space may be written.
APE Shared Memory Write Enable	17	RW	0	When this bit is set the APE shared memory region may be written.
APE Control Register Write Enable	16	RW	0	When this bit is set the APE control registers may be written.
Config Retry	15	RO	0x1 On Hard reset	When asserted, forces all config access to be retried.
Reserved	14:12	RO	0x0	–
Max PCI Target Retry	11:9	RW	0x1	Indicates 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
Flat View	8	RW	0x0	Asserted if the Base Address register presents a 32 MB PCI Address map flat view, otherwise, indicates a 64 KB PCI Address map in standard view
VPD Available	7	RO	0x0	This bit reads as 1 if the VPD region of the NVRAM can be accessed by the host Comes from GRC 6808
PCI Expansion ROM Retry	6	RW	0x0	Force PCI Retry for accesses to Expansion ROM region if enabled
PCI Expansion ROM Desired	5	RW	0x0	Enable PCI ROM base address register to be visible to the PCI host
Reserved	4:0	RO	XXX	–

Reset Counters Initial Values Register (offset: 0x74)

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.
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)

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)

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)

Name	Bits	Access	Default Value	Description
Register Data Register	31:0	RW	X	Register Data at the location pointed by the Register Base Register

Memory Data Register (offset: 0x84)

Name	*Bits*	*Access*	*Default Value*	*Description*
Memory Base Register	31:0	RW	X	Memory value at the location pointed by the Memory Base Register

UNDI Receive Return Ring Consumer Index Register (offset: 0x88–0x8C)

Name	*Bits*	*Access*	*Default Value*	*Description*
UNDI Receive Return C_Idx	63:0	RW	0	UNDI Receive Return Ring Consumer Index Mailbox

UNDI Send BD Producer Index Mailbox Register (offset: 0x90–0x94)

Name	*Bits*	*Access*	*Default Value*	*Description*
UNDI Send BD NIC P_Idx	63:0	RW	0	UNDI Send BD NIC Producer Index Mailbox

UNDI Receive BD Standard Producer Ring Producer Index Mailbox Register (offset: 0x98–0x9C)

Name	*Bits*	*Access*	*Default Value*	*Description*
UNDI Receive BD Standard Ring Producer Index	63:0	RW	0	UNDI Receive BD Std. Ring Producer Index Mailbox

MSI-X Capabilities Registers

MSI-X Capability Header Register (offset: 0xA0)

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. Path = i_cfg_func.i_cfg_public.i_cfg_msi_cap
FUNC_MASK	30	RW	0	If 1, all of the vectors associated with the function are masked regardless of their per vector Mask bit. Path = i_cfg_func.i_cfg_public.i_cfg_msi_cap
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 Path = i_cfg_func.i_cfg_private
MSIX_NEXT_CAP_PTR	15:8	RO	0xac	This value continues the PCI capability chain. It's value specified as 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. Path = i_cfg_func.i_cfg_public.i_cfg_ep_reg
MSIX_CAP_ID	7:0	RO	0x11	Capability ID for MSIX Path = cfg_defs

MSIX_TBL_OFF_BIR – 0xa4

Name	*Bits*	*Access*	*Default Value*	*Description*
TABLE_OFFSET	31:3	RO	0	Path = i_cfg_func.i_cfg_private
TABLE_BIR	2:0	RO	0	Indicates which one of functions BAR is used to map MSI-X table into memory space. Path = i_cfg_func.i_cfg_private

MSIX_PBA_BIR_OFF – 0xa8

Name	*Bits*	*Access*	*Default Value*	*Description*
TABLE_OFFSET	31:3	RO	0	Path = i_cfg_func.i_cfg_private
TABLE_BIR	2:0	RO	0	Indicates which one of functions BAR is used to map MSI-X table into memory space. Path = i_cfg_func.i_cfg_private

PCIe Capabilities Registers

PCIE_CAPABILITY – 0xac

Name	*Bits*	*Access*	*Default Value*	*Description*
unused0	31:30	RO	0	–
MSG_NUM	29:25	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
SLOT_IMPLEMENTED	24	RO	0	Slot Implemented. This register is not supported. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
TYPE	23:20	RO	0	Slot Implemented. This register is not supported. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
VER	19:16	RO	0x2	Capability Version. PCI Express Capability structure version number. These bits are hardwired to 2h. Path= cfg_defs
PCIE_NEXT_CAP_PTR	15:8	RO	0	This registers contains the pointer to the next PCI capability structure. Path= i_cfg_func.i_cfg_public.i_cfg_rd_mux
PCIE_CAP_ID	7:0	RO	0x10	This register contains the PCIExpress Capability ID. Path= i_cfg_func.i_cfg_public.i_cfg_rd_mux

DEVICE_CAPABILITY – 0xb0

Name	*Bits*	*Access*	*Default Value*	*Description*
unused3	31:29	RO	0	–
FLR_CAP_SUPPORTED	28	RO	0	FLR capability is advertized when flr_supported bit in private device_capability register space is set.
CAPTURED_SLOT_PWR_SCALE	27:26	RO	0	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 Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CAPTURED_SLOT_PWR_VAL	25:18	RO	0	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
unused2	17:16	RO	0	–
ROLE_BASED_ERR_RPT	15	RO	0x1	Indicate device is conforming to the ECN, PCI Express Base Specification, Revision 1.1., or subsequent PCI Express Base Specification revisions. Path= i_cfg_func.i_cfg_private
unused1	14:12	RO	0	–
L1_ACCEPTABLE_LATENCY	11:9	RO	0x6	Endpoint L1 Acceptable Latency. These bits are programmable through register space. The bits should be 0 for Root ports. Path= i_cfg_func.i_cfg_private
L0S_ACCEPTABLE_LATENCY	8:6	RO	0x6	Endpoint L0s Acceptable Latency. These bits are programmable through register space. The value should be 0 for root ports. Path= i_cfg_func.i_cfg_private
EXTENDED_TAG_SUPPORT	5	RO	0	Extended Tag Field Support. This bit is programmable through register space. This capability is not currently supported. Path= i_cfg_func.i_cfg_private
unused0	4:3	RO	0	
MAX_PL_SIZE_SUPPORTED	2:0	RO	0x1	Max Payload Size Supported. These bits are programmable from the register space and default value is based on define in version.v file. Path= i_cfg_func.i_cfg_private

DEVICE_STATUS_CONTROL – 0xb4

Name	*Bits*	*Access*	*Default Value*	*Description*
Unused1	31:22	RO	0	–
NP_TRANSACTION_PEND	21	RO	0	This is bit is read back a 1, whenever a non-posted request initiated by PCIE core is pending to be completed. Path= i_tl_top
AUX_PWR_DET	20	RO	0x1	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. Path= input to pcie_vaux_pipe
UNSUP_REQ_DET	19	WC	0	UnSupported Request Detected. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
FATAL_ERR_DET	18	WC	0	Fatal Error Detected. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
NON_FATAL_ERR_DET	17	WC	0	Non-Fatal Error Detected. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
CORR_ERR_DET	16	WC	0	Correctable Error Detected. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
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_REQ_SIZ	14:12	RW	0	Maximum Read Request Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
NO_SNOOP_ENABLE	11	RW	0x1	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
AUX_PWR_PM_ENA	10	RW	0x1	This bit when set enables device to draw aux power independent of PME AUX power Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
Unused0	9	RO	0	–
EXTENDED_TAG_EN	8	RO	0	Extended Tag Field Enable. This capability is not supported. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
MAX_PAYLOAD_SIZE	7:5	RW	0	Max Payload Size. Depending on the spec, internal logic uses either the min or the max of the value of the two functions. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
RELAX_ORDERING_ENABLE	4	RW	0x1	Relax Ordering Enable. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.

Name	Bits	Access	Default Value	Description
U_REQ_REPORT_EN	3	RW	0:pr	Unsupported Request Reporting Enable. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
FATAL_ERR_REPORT_EN	2	RW	0:pr	Fatal Error Reporting Enable. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
NFATAL_ERR_REPORT_EN	1	RW	0:pr	Non-Fatal Error Reporting Enable. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.
CORR_ERR_REPORT_EN	0	RW	0:pr	Correctable Error Reporting Enable. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap.

LINK_CAPABILITY – 0xb8

Name	Bits	Access	Default Value	Description
PORT_NUMBER	31:24	RO	0	PCIE Port Number. These bits are programmable through register. Path= i_cfg_func.i_cfg_private
Unused0	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
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. Path= i_cfg_func.i_cfg_private
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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CLK_PWR_MGMT	18	RO	0x1	Clock Power Management. These bits are programmable through register. The feature itself has to be enabled in version.v Path= i_cfg_func.i_cfg_private

Name	Bits	Access	Default Value	Description
L1_EXIT_LAT	17:15	RO	0x2	L1 Exit Latency. These bits are programmable through register space. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following. **Value / Name / Description** 1 / 1_2 / L1 exit latency of 1 us to 2 us. 2 / 2_4 / L1 exit latency of 2 us to 4 us. 255 / – / end_of_table
L0S_EXIT_LAT	14:12	RO	0x5	L0s Exit Latency. These bits are programmable through register space. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap Depending on whether device is in common clock mode or not, the value reflected by these bits is one of the following. **Value / Name / Description** 4 / 0_1 / L0s exit latency of 512 ns to 1 us. 5 / 1_2 / L0s exit latency of 1 us to 2 us. 255 / – / end_of_table ASPM Support. These bits are programmable through reg space. Path= i_cfg_func.i_cfg_private
ASPM_SUPT	11:10	RO	0x3	**Value / Name / Description** 0 / RES_0 / Reserved 1 / L0S / L0s entry supported 2 / RES_2 / Reserved 3 / L0S_L1 / L0s and L1 supported 255 / – / end_of_table Path= i_cfg_func.i_cfg_private Value used by internal logic is the smaller of the value programmed for each function

Name	Bits	Access	Default Value	Description
MAX_LINK_SPEED	3:0	RO	0x1	

Value	Name	Description
0	64	64 Bytes
1	128	128 Bytes
255	–	end_of_table

LINK_STATUS_CONTROL – 0xbc

Name	Bits	Access	Default Value	Description
Unused3	31:30	RO	0	–
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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
SLOT_CLK_CONFIG	28	RO	0x1	Slot Clock configuration. This bit is read-only by host, but read/write via backdoor CS bus. Path= i_cfg_func.i_cfg_private
LINK_TRAINING	27	RO	0	EP: This bit is N/A and is hardwired to 0. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
Unused2	26	RO	0	–
NEG_LINK_WIDTH	25:20	RO	0	Negotiated Link Width. These bits indicate the negotiated link width of the PCI Express link. Path= i_pl_top.i_pl_ltssm
NEG_LINK_SPEED	19:16	RO	0	Link Speed. These bits indicate the negotiated link speed of the PCI Express link. Path= i_pl_top.i_pl_ltssm
Unused1	15:12	RO	0	–
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 Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

Name	Bits	Access	Default Value	Description
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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
EN_CLK_PW_MGMT	8	RW	0	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 Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
LINK_CR_EXT_SYNC	7	RW	0	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
LINK_CR_COMMON_CLK	6	RW	0	Common Clock Configuration. Value used by logic is resolved to 1 only if both functions (when enabled) have this bit set. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CFG_PSM_RETRAIN_LINK	5	RO	0	Requesting PHY to retrain the link. This bit is only applicable to RC. So for EP it is read only bit. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CFG_PSM_LINK_DISABLE	4	RO	0	Requesting PHY to disable the link. This bit is only applicable to RC. So for EP it is read only bit. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
RCB	3	RW	0	Read Completion Boundary. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

Value	Name	Description
0	64	64 Bytes
1	128	128 Bytes
255	–	end_of_table

Unused0	2	RO	0	–

Name	Bits	Access	Default Value	Description
ASPM_CTRL	1:0	RW	0	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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

SLOT_CAPABILITY – 0xc0

Name	Bits	Access	Default Value	Description
PHYSICAL_SLOT_NUMBER	31:19	RO	0	Not implemented
UNUSED	18:17	RO	0	Not implemented
SLOT_POWER_LIMIT_SCALE	16:15	RO	0	Not implemented
SLOT_POWER_LIMIT_VALUE	14:7	RO	0	Not implemented
UNUSED_2	6:0	RO	0	Not implemented

SLOT_CONTROL_STATUS – 0xc4

Name	Bits	Access	Default Value	Description
SLOT_STATUS	31:23	RO	0	Not implemented
PRESENCE_DETECT	22	RO	0	Not implemented
UNUSED_1	21:16	RO	0	Not implemented
SLOT_CONTROL	15:0	RO	0	Not implemented

ROOT_CAP_CONTROL – 0xc8

This register is not applicable for EP and hardwired to 0.

Name	Bits	Access	Default Value	Description
Unused	31:0	RO	0	–

ROOT_STATUS – 0xcc

This register is not applicable for EP and hardwired to 0.

Name	Bits	Access	Default Value	Description
Unused	31:0	RO	0	–

DEVICE_CAPABILITY_2 – 0xd0

Name	Bits	Access	Default Value	Description
Unused1	31:12	RO	0	–
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.
Unused0	10:5	RO	0	–
CMPL_TIMEOUT_DISABL_ SUPPORTED	4	RO	0x1	Completion Timeout Disable Supported, Programmable through register space Path= i_cfg_func.i_cfg_private
CMPL_TIMEOUT_RANGES_ SUPPORTED	3:0	RO	0xf	Completion Timeout Ranges Supported. Programmable through register space Path= i_cfg_func.i_cfg_private

Value	Name	Description
15	ABCD	Ranges A, B, C, and D
255	–	end_of_table

DEVICE_STATUS_CONTROL2 – 0xd4

Name	Bits	Access	Default Value	Description
DEVICE_STATUS_2	31:16	RO	0	Placeholder for Gen2 Path= i_cfg_func.i_cfg_public.i_cfg_rd_mux
Unused	15:11	RO	0	–
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.
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.
Unused0	7:5	RO	0	–

Name	Bits	Access	Default Value	Description
CMPL_TIMEOUT_DISABLE	4	RW	0	Completion Timeout Disable Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CMPL_TIMEOUT_VALUE	3:0	RW	0	

Value	Name	Description
0	50MS	50ms
1	100US	100us
2	10MS	10ms
3	55MS	55ms
4	210MS	210ms
5	900MS	900ms
6	3_5S	3.5s
	13S	13s
8	64S	64s
255	–	end_of_table

LINK_CAPABILITY_2 – 0xd8

Name	Bits	Access	Default Value	Description
LINK_CAPABILITY_2	31:0	RO	0	Placeholder for Gen2 Path= i_cfg_func.i_cfg_private

LINK_STATUS_CONTROL_2 – 0xdc

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 Path = pl_top
Unused0	15:13	RO	0	–
CFG_COMPLIANCE_DEEMPH	12	RW	0	Compliance De-emphasis. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CFG_COMPLIANCE_SOS	11	RW	0	Compliance SOS. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
CFG_ENTER_MOD_ COMPLIANCE	10	RW	0	Enter Modified Compliance. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

Name	Bits	Access	Default Value	Description
CFG_TX_MARGIN	9:7	RW	0	Controls the value of non de-emphasized voltage level at the TX pins. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap Value used by logic is resolved to the smaller binary value, if two functions have different values.

Value	Name	Description
0	000	800 – 1200 mV for full swing and 400 – 600 mV for half swing.
1	001	Values will be monotonic with non zero Slope
2	010	Values will be 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
255	–	end_of_table

Name	Bits	Access	Default Value	Description
SEL_DEEMPHASIS	6	RW	0	When link is operating at Gen2 rates, this bit selects the level of de-emphasis. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap Value used by logic is resolved to 1 if either function has this bit set.

Value	Name	Description
0	0	–
1	1	–
255	–	–

Name	Bits	Access	Default Value	Description
HW_AUTO_SPEED_DISABLE	5	RO	0	Not Supported and hardwired to 0. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap
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. Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

Name	Bits	Access	Default Value	Description
TARGET_LINK_SPEED	3:0	RW	0x1	Upper limit of link speed Path= i_cfg_func.i_cfg_public.i_cfg_exp_cap

Value	Name	Description
0	2_5	2.5 Gbps
1	5_0	5.0 Gbps
255	–	end_of_table

SLOT_CAPABILITY_2 – 0xe0

Name	Bits	Access	Default Value	Description
SLOT_CAPABILITY_2	31:0	RO	0	Not implemented

SLOT_STATUS_CONTROL_2 – 0xe4

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 ASIC ID (offset: 0xF4)

Name	Bits	Access	Default Value	Description
Product ASIC ID	31:0	RO		For 5717 B0, value is 0x05717100: • 5718 A0, value is 0x05717000 • 5718 B0, value is 0x05717100 • 5719 A0, value is 0x05719000 • 5719 A1, value is 0x05719100 • 5720 AO, value is 0x05720000

Advanced Error Reporting Enhanced Capability Header (offset: 0x100)

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Express Extended Capability ID	15:0	RO	0x0001	Extended Capability ID for the Advanced Error Reporting Capability is 0001h
Capability Version	19:16	RO	0x1	–
Next Capability Offset	31:20	RO	0x13C	Pointer to the Virtual Channel Capability Structure

Uncorrectable Error Status Register (offset: 0x104)

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)

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.
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)

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
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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RO	0	–
Advisory Nonfatal 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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	RO	0	–
Advisory Nonfatal Error Mask	13	RWS	1	Setting this bit masks Advisory Nonfatal errors.
Replay Timer Timeout Mask	12	RWS	0	Setting this bit masks Replay Timer Timeout errors.
Reserved	11:9	RO	0	–
REPLAY_NUM Rollover Mask	8	RWS	0	Setting this bit masks REPLAY_NUM Rollover errors.
Bad DLLP Mask	7	RWS	0	Setting this bit masks Bad DLLP errors.
Bad TLP Mask	6	RWS	0	Setting this bit masks Bad TLP errors.
Reserved	5:1	RO	0	–
Receiver Error Mask	0	RWS	0	Setting this bit masks Receiver errors.

Advanced Error Capabilities and Control Register (offset: 0x118)

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)

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)

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.
Header Byte 7	7:0	ROS	–	The TLP header of the transaction that has incurred a failure.

Header Log Register (offset: 0x124)

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)

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.

Interrupt mail box (High Priority Mailbox) Register (offset: 0x200 - 0x21c)

This mailbox serves two functions. When the host writes it, the interrupt (IntA) is cleared. It is also used by the Host Coalescing engine to determine if the host is in the interrupt handler. If it is non-zero this indicates the host is in the interrupt handler. If it is zero this indicates the host is not in the interrupt handler. The Host Coalescing engine uses this information to determine which set of coalescing parameters it should use.

General mail box (High Priority Mailbox) Register (offset: 0x220 - 0x25c)

Reload Statistics mail box (High Priority Mailbox) Register (offset: 0x260 - 0x264)

High Priority Mailbox Registers

All registers reset are core reset unless specified.

Note: When performing a 32 bit access the higher addresses should be accessed last.

Receive BD Standard Producer Ring Index Register (offset: 0x268-0x26F)

Name	*Bits*	*Access*	*Default Value*	*Description*
Received BD standard Producer Ring Index	7:0	RW	0	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 Jumbo Producer Ring Index Register (offset: 0x270)

Table 98: Receive BD Jumbo Producer Ring Index 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 BD Return Ring 0 Consumer Index 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 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 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 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 Register (offset: 0x300–0x307)

The Send BD Ring Host Producer Index Register contains the index of the next buffer descriptor for a given standard (non-jumbo) 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.

RX Mail Box Registers for VRQ

A set of new High Priority Mail Box Registers have been introduced. The register addresses are as shown in Table 99 below:

Table 99: High Priority Mail Box Registers for VRQ Rings

VRQ #	*RX Jumbo Ring Producer Index*	*RX Standard Ring Producer Index*	*RX Return Ring Consumer Index*	*Comments*	*NIC Diagnostic RX Return Ring Producer Index*
0	0x270	UNDI - 0x98 HP - 0x268	UNDI - 0x88 HP - 0x280	64-bit Registers	0x3C80
1	0x2D4	0x354	0x288 (64-bit Regs)	The new Registers are 32-bit	0x3C84
2	0x2D8	0x358	0x290 (64-bit Regs)		0x3C88
3	0x2DC	0x35C	0x298 (64-bit Regs)		0x3C8C

Table 99: High Priority Mail Box Registers for VRQ Rings (Cont.)

VRQ #	RX Jumbo Ring Producer Index	RX Standard Ring Producer Index	RX Return Ring Consumer Index	Comments	NIC Diagnostic RX Return Ring Producer Index
4	0x2E0	0x360	0x2A0		0x3C90
5	0x2E4	0x364	0x2A4		0x3C94
6	0x2E8	0x368	0x2A8		0x3C98
7	0x2EC	0x36C	0x2AC		0x3C9C
8	0x2F0	0x370	0x2B0		0x3CA0
9	0x2F4	0x374	0x2B4		0x3CA4
10	0x2F8	0x378	0x2B8	The new Registers are 32-bit	0x3CA8
11	0x2FC	0x37C	0x2BC		0x3CAC
12	0x340	0x380	0x2C0		0x3CB0
13	0x344	0x384	0x2C4		0x3CB4
14	0x348	0x388	0x2C8		0x3CB8
15	0x34C	0x38C	0x2CC		0x3CBC
16	0x350	0x390	0x2D0		0x3CC4

Additional Notes:

- 0x268 - 0x298, 0x88, 0x98 are legacy registers
- NIC Diagnostic Standard RBD Consumer Index [1 - 16] == 0x3F40 - 0x3F7C
- NIC Diagnostic Jumbo RBD Consumer Index [1 - 16] == 0x3F80 - 0x3FFC

Ethernet MAC (EMAC) Registers

All registers reset are core reset unless specified.

EMAC Mode Register (offset: 0x400)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	–
Mac loop back mode control	29	RW	1	1: gate off outgoing 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

Name	*Bits*	*Access*	*Default Value*	*Description*
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 TCE	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.
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.

Name	Bits	Access	Default Value	Description
Port Mode	3:2	RW	01	11: Obsolete (Was TBI) 10: 1000BT(Copper) GMII or Serdes 1000 mode 01: 10/100BT(Copper) MII or Serdes 10/100 mode 00: None
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	31:30	RO	0	–
Reserved.	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.
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	–

Name	Bits	Access	Default Value	Description
Link State Changed	12	RO	0	Set when the link state has changed Generates an attention when enabled by bit 12 of the EMAC Event Enable register. Clear this attention by writing 1 to Sync Changed (bit 4) and Config changed (bit 3).
Reserved	11:0	RO	0	–

EMAC Event Enable Register (offset: 0x408)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	–
Enable TX Offload Error Interrupt	29	RW	0	Enables or unmasks the interrupt associated with Reg 0x404[29].
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	0	–
Reserved	15	RO	0	–
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 high (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	0	–
10 Mbps LED Status	9	RO	0	–

Name	Bits	Access	Default Value	Description
100 Mbps LED Status	8	RO	0	–
1000 Mbps LED Status	7	RO	0	–
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	–
ACPI Pointer	8:0	RW	0	Specifies the offset into the 6 KB 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.

Gigabit PCS Test Register (offset: 0x440)

Name	Bits	Access	Default Value	Description
Reserved	31:0	RO	0	–

Transmit 1000BASE-X Auto-Negotiation Register (offset: 0x444)

Name	Bits	Access	Default Value	Description
Reserved	31:0	RO	0	–

Receive 1000BASE-X Auto-Negotiation Register (offset: 0x448)

Name	Bits	Access	Default Value	Description
Reserved	31:0	RO	0	–

MII Communication Register (offset: 0x44C)

Name	Bits	Access	Default Value	Description
Reserved	31:30	RO	0	–
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 MI 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	0x8: SGMII Serdes Port0. 0x9: SGMII Serdes Port1. as strapped: External PHY Port0 as strapped: External PHY Port1
Register Address	20:16	RW	0	Address of the register to be read or written.

Name	Bits	Access	Default Value	Description
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
Reserved	31:2	RO	0	–
Mode 10 Mbps	1	RW	0	When read, a value of 1 indicates the transceiver device is operating in 10 Mbps 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
Reserved	31:21	RO	0	–
MII Clock Count	20:16	RW	0Ch	Counter to divide CORE_CLK (62.5 MHz) to generate the MI clock. The formula is: MI Clock = CORE_CLK/2/(MI Clock Count + 1).
Constant MDIO/MDC clock speed.	15	RW	0	Enable ~500Khz constant MII management interface (MDIO/MDC) frequency regardless core clock frequency. 1: Enable 0: Disable
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	–

Name	Bits	Access	Default Value	Description
Auto_control	2	RW	0	–
Use Short Preamble	1	RW	1	Use short preamble while polling, if set.
Fast_Clock	0	RW	0	–

Autopolling Status Register (offset: 0x458)

Name	Bits	Access	Default Value	Description
Reserved	31:1	RO	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	RW	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, i.e., [19:17] of this register is equal to "001".
Transmit FTQ Arbitration Mode	26:24	RW	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.

Name	*Bits*	*Access*	*Default Value*	*Description*
HTX2B Count Down Mode	23	RW	0	**When this bit is 0:** If there is an HTX2B packet to be enqueued to the HTX2B, FIFO. However, if the FIFO does not have sufficient space, TCE drops the copy of the HTX2B-bound packet and moves on. **When this bit is 1:** If there is an HTX2B packet to be enqueued to the HTX2B, FIFO. However, if the FIFO does not appear to have sufficient space at the first attempt, TCE counts down a few clocks, then makes a second attempt. If the FIFO still does not have sufficient space, TCE simply drops the copy of the HTX2B-bound packet and moves on. The number of clocks to count-down is programmable at register 0x464[31:24].
HTX2B Programmable Jumbo Frame Length	22	RW	0	**When this bit is 0:** Hardware automatically applies preset lengths to detect jumbo frames for HTX2B. **When this bit is 1:** The jumbo frame length for HTX2B is programmable via 0x464[23:16].
Reserved	21	–	–	–
TX-MBUF Burst Size	20:17	RW	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	RW	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	RW	0	These bit are there for debug purposes. Normally an offloaded TX packet that does not adhere to BCM5718 family 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	RW	0	
Do not drop if unsupported IPV6 extension found or IPv4 option found	13	RW	0	
Do not drop if SA invalid	12	RW	0	
Do not drop if AH/ESP Header not found	11	RW	0	

Name	Bits	Access	Default Value	Description
Enable TX AH Offload	10	RW	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	RW	0	A value 1 enables the TX ESP offload feature. When 0, offloaded ESP packet gets dropped. This value must be static.
TxMBUF corruption lockup fix enable	8	RW	0	When set, TXMBUF corruption lockup fix is enabled.
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 TCE	1	RW	1	Used to be enable TDE in legacy–same purpose.
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	–
Consumer Index	22:11	RO	UUUU	The Consumer Index of the erring packet is reported by this field.
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*
HTX2B Count Down Value	31:24	RW	0x1	HT2XB Count Down Value
HTX2B Jumbo Frame Length	23:16	RW	0x0	This value + 1500 is used by hardware as the maximum standard frame length for HTX2B. A frame with a length larger than that is a jumbo frame for HTX2B. The length is the effective length of a composed L2 frame as seen from the wire, including the L2 header, L2 payload, and the FCS (CRC) field.
Reserved	15:14	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*
FIX EMAC drops first packet on false carrier event	31	RW	0	This bit when set disables the fix where EMAC drops first packet on False Carrier Event. 1: Disable Fix 0: Enable Fix
Reserved	30	RO	0	–
Disable 802.3 length check fix for VLAN Tag frames	29	RW	0	If clear, 802.3 length check takes VLAN length into account properly.
Reset Management Filter Set	28	WO	0	Writing a 1 to this field generates a pulse to reset all Management Filter registers.
Enable RX AH Offload	27	RW	0	A value 1 enables the RX AH offload feature. When 0, in coming packets are not CAMed for offload consideration.
Enable RX ESP Offload	26	RW	0	A value 1 enables the RX ESP offload feature. When 0, in coming packets are not CAMed for offload consideration.
APE promiscuous mode enable	25	RW	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.

Name	*Bits*	*Access*	*Default Value*	*Description*
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	1	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.
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	15:13	RO	0	–
FIX EMAC drops a packets if incoming DA partial match both perfect and Pause Multicast address	12	RW	0	This bit disables the fix that EMAC stage 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, no 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.

Name	*Bits*	*Access*	*Default Value*	*Description*
Accept Runts	6	RW	0	If set, MAC accepts packets less than 64 bytes.
Reserved	5	RO	0	–
Keep Pause	4	RW	0	If set, MAC forwards pause frame to host buffer.
Reserved	3	RO	0	–
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	31:4	RO	0	–
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 BCM5718 family 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.
Activate Processor 2	28	RW	0	If the rule matches, the processor is activated in the queue descriptor for the Receive Queue Placement state machine.
Activate Processor 3	27	RW	0	If the rule matches, the processor is activated in the queue descriptor for the Receive Queue Placement state machine.
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	–
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)

Name	Bits	Access	Default Value	Description
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 BCM5718 family employs eight receive rules (N = 0 to 7).

Name	Bits	Access	Default Value	Description
Mask/Value	31:16	RW	0	For each bit set, the corresponding bit in the value field is ignored during the rule match process. If bit 26 of the corresponding rule control register is set, the field is used as an additional 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	5:3	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.
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).

APE_PERFECT_MATCH[1–4]_HIGH_REG (Offsets 0x540, 0x548, 0x550, 0x558)

There are total 4 Perfect (Destination Address) Match registers dedicated to APE in RX-MAC. These registers hold the higher 2 octets of the matching address.

Name	Bits	Access	Default Value	Description
Reserved	31:29	RO	000	–
MAC High Address	15:0	RW	0x0000	Upper 2-bytes of APE's [1–4]th unicast address.

APE_PERFECT_MATCH[1–4]_LOW_REG (Offsets 0x544, 0x54C, 0x554, 0x55C)

There are total 4 Perfect (Destination Address) Match registers dedicated to APE in 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 APE's [1–4]th unicast address.

SGMII Control Register (offset: 0x5B0)

Name	Bits	Access	Default Value	Description
Reserved	–	–	–	–

SGMII Status Register (offset: 0x5B4)

This register reflects various status of the respective SGMII port when enabled.

Name	Bits	Access	Default Value	Description
LP AutoNeg Capability	31:16	RO	0	Link partner advertised auto-negotiation abilities.
Reserved	15:11	RO	0	–
External CRS Detect	10	RO	0	External PHY's CRS output
PCS CRS Detect	9	RO	0	Internal PCS blocks CRS output
Media Selection mode	8	RO	0	1: SGMII/1000BaseX mode selected for this port 0: Copper Media Selected for this port
Pause TX	7	RO	0	1: enable pause TX
Pause RX	6	RO	0	1: enable pause RX
AutoNeg Next Page RX	5	RO	0	1: next auto-negotiation page received
Speed_100	4	RO	0	The SGMII Link currently operable at 100mbps data speed.
Speed_1000	3	RO	0	The SGMII Link currently operable at 1 Gbps data speed.
Duplex Status	2	RO	0	1: The Link currently is in Full Duplex mode. 0: The Link is currently in Half Duplex mode.
Link Status	1	RO	0	1: Link is up. 0: Link is down.
AutoNeg Completion	0	RO	0	Auto-negotiation process has completed.

HTX2B Perfect Match[1–4] HI Reg (offset: 0x4880, 0x4888, 0x4890, 0x4898)

There are four Perfect (Destination Address) Match registers in DMAR for HTX2B. These registers hold the higher two 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 Destination Address.

HTX2B Perfect Match[1–4] LO Reg (offset: 0x4884, 0x488C, 0x4894, 0x489C)

There are four Perfect (Destination Address) Match registers in DMAR for HTX2B. These registers hold the lower two octets of the matching address.

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Low Address	31:0	RW	0x0000	Lower 4-bytes of Destination Address.

HTX2B Protocol Filter Reg (offset: 0x6D0)

This register resides in TCE.

Name	*Bits*	*Access*	*Default Value*	*Description*
Unused	31:27	RO	0x0	
Duplicate–DHCPv6 Relay and Server multicast	26	RW	0	Write a 1 to each bit to enable the respective HTX2B protocol filter duplication. ***Note:*** The respective Protocol Filter Enable bit must be 1 for a Duplication to be meaningful.
Duplicate–IPv6 Router Advertisement	25	RW	0	
Duplicate–IPv6 Neighbor Advertisement	24	RW	0	
Duplicate–NetBios Packet	23	RW	0	
Duplicate–DHCP Server Packet	22	RW	0	
Duplicate–DHCP Client Packet	21	RW	0	
Duplicate–ARP Packet	20	RW	0	
Duplicate–HTX2B Perfect Match Address[3]	19	RW	0	
Duplicate–HTX2B Perfect Match Address[2]	18	RW	0	
Duplicate–HTX2B Perfect Match Address[1]	17	RW	0	
Duplicate–HTX2B Perfect Match Address[0]	16	RW	0	
Unused	15:11	RO	0x0	

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable–DHCPv6 Relay and Server multicast	10	RW	0	Write a 1 to each bit to enable the respective HTX2B protocol filter.
Enable–IPv6 Router Advertisement	9	RW	0	
Enable–IPv6 Neighbor Advertisement	8	RW	0	
Enable–NetBios Packet	7	RW	0	
Enable–DHCP Server Packet	6	RW	0	
Enable–DHCP Client Packet	5	RW	0	
Enable–ARP Packet	4	RW	0	
Enable–HTX2B Perfect Match Address[3]	3	RW	0	
Enable–HTX2B Perfect Match Address[2]	2	RW	0	
Enable–HTX2B Perfect Match Address[1]	1	RW	0	
Enable–HTX2B Perfect Match Address[0]	0	RW	0	

HTX2B Global Filter Reg (address: 0x6D4)

This register resides in TCE.

Name	Bits	Access	Default Value	Description
Unused	31:2	RU	0xUUUU	Unused.
HTX2B Broadcast Filter Enable	1	RW	0	Write 1 to replicate and route all Host Send broadcast packets to APE. This bit overrides any conflicting Protocol Filter setting.
HTX2B Multicast Filter Enable	0	RW	0	Write 1 to replicate and route all Host Send multicast packets to APE. This bit overrides any conflicting Protocol Filter setting.

RSS Registers

All registers reset are core reset unless specified.

Indirection Table Register 0 (offset: 0x630)

Name	Bits	Access	Default Value	Description
table_entry0	31:28	RW	0	The RSS_ring value for entry 0.
table_entry1	27:24	RW	0	The RSS_ring value for entry 1.
table_entry2	23:20	RW	0	The RSS_ring value for entry 2.
table_entry3	19:16	RW	0	The RSS_ring value for entry 3.
table_entry4	15:12	RW	0	The RSS_ring value for entry 4.
table_entry5	11:8	RW	0	The RSS_ring value for entry 5.
table_entry6	7:4	RW	0	The RSS_ring value for entry 6.
table_entry7	3:0	RW	0	The RSS_ring value for entry 7.

Indirection Table Register 2 (offset: 0x634)

Name	Bits	Access	Default Value	Description
table_entry8	31:28	RW	0	The RSS_ring value for entry 8.
table_entry9	27:24	RW	0	The RSS_ring value for entry 9.
table_entry10	23:20	RW	0	The RSS_ring value for entry 10.
table_entry11	19:16	RW	0	The RSS_ring value for entry 11.
table_entry12	15:12	RW	0	The RSS_ring value for entry 12.

Name	Bits	Access	Default Value	Description
table_entry13	11:8	RW	0	The RSS_ring value for entry 13.
table_entry14	7:4	RW	0	The RSS_ring value for entry 14.
table_entry15	3:0	RW	0	The RSS_ring value for entry 15.

Indirection Table Register 3 (offset: 0x638)

Name	Bits	Access	Default Value	Description
table_entry16	31:28	RW	0	The RSS_ring value for entry 16.
table_entry17	27:24	RW	0	The RSS_ring value for entry 17.
table_entry18	23:20	RW	0	The RSS_ring value for entry 18.
table_entry19	19:16	RW	0	The RSS_ring value for entry 19.
table_entry20	15:12	RW	0	The RSS_ring value for entry 20.
table_entry21	11:8	RW	0	The RSS_ring value for entry 21.
table_entry22	7:4	RW	0	The RSS_ring value for entry 22.
table_entry23	3:0	RW	0	The RSS_ring value for entry 23.

Indirection Table Register 4 (offset: 0x63C)

Name	Bits	Access	Default Value	Description
table_entry24	31:28	RW	0	The RSS_ring value for entry 24.
table_entry25	27:24	RW	0	The RSS_ring value for entry 25.
table_entry26	23:20	RW	0	The RSS_ring value for entry 26.
table_entry27	19:16	RW	0	The RSS_ring value for entry 27.
table_entry28	15:12	RW	0	The RSS_ring value for entry 28.
table_entry29	11:8	RW	0	The RSS_ring value for entry 29.
table_entry30	7:4	RW	0	The RSS_ring value for entry 30.
table_entry31	3:0	RW	0	The RSS_ring value for entry 31.

Indirection Table Register 5 (offset: 0x640)

Name	Bits	Access	Default Value	Description
table_entry32	31:28	RW	0	The RSS_ring value for entry 32.
table_entry33	27:24	RW	0	The RSS_ring value for entry 33.
table_entry34	23:20	RW	0	The RSS_ring value for entry 34.
table_entry35	19:16	RW	0	The RSS_ring value for entry 35.
table_entry36	15:12	RW	0	The RSS_ring value for entry 36.
table_entry37	11:8	RW	0	The RSS_ring value for entry 37.
table_entry38	7:4	RW	0	The RSS_ring value for entry 38.
table_entry39	3:0	RW	0	The RSS_ring value for entry 39.

Indirection Table Register 6 (offset: 0x644)

Name	Bits	Access	Default Value	Description
table_entry40	31:28	RW	0	The RSS_ring value for entry 40.
table_entry41	27:24	RW	0	The RSS_ring value for entry 41.
table_entry42	23:20	RW	0	The RSS_ring value for entry 42.
table_entry43	19:16	RW	0	The RSS_ring value for entry 43.
table_entry44	15:12	RW	0	The RSS_ring value for entry 44.
table_entry45	11:8	RW	0	The RSS_ring value for entry 45.
table_entry46	7:4	RW	0	The RSS_ring value for entry 46.
table_entry47	3:0	RW	0	The RSS_ring value for entry 47.

Indirection Table Register 8 (offset: 0x648)

Name	Bits	Access	Default Value	Description
table_entry48	31:28	RW	0	The RSS_ring value for entry 48.
table_entry49	27:24	RW	0	The RSS_ring value for entry 49.
table_entry50	23:20	RW	0	The RSS_ring value for entry 50.
table_entry51	19:16	RW	0	The RSS_ring value for entry 51.
table_entry52	15:12	RW	0	The RSS_ring value for entry 52.
table_entry53	11:8	RW	0	The RSS_ring value for entry 53.
table_entry54	7:4	RW	0	The RSS_ring value for entry 54.
table_entry55	3:0	RW	0	The RSS_ring value for entry 55.

Indirection Table Register 8 (offset: 0x64C)

Name	Bits	Access	Default Value	Description
table_entry56	31:28	RW	0	The RSS_ring value for entry 56.
table_entry57	27:24	RW	0	The RSS_ring value for entry 57.
table_entry58	23:20	RW	0	The RSS_ring value for entry 58.
table_entry59	19:16	RW	0	The RSS_ring value for entry 59.
table_entry60	15:12	RW	0	The RSS_ring value for entry 60.
table_entry61	11:8	RW	0	The RSS_ring value for entry 61.
table_entry62	7:4	RW	0	The RSS_ring value for entry 62.
table_entry63	3:0	RW	0	The RSS_ring value for entry 63.

Indirection Table Register 9 (offset: 0x650)

Name	Bits	Access	Default Value	Description
table_entry64	31:28	RW	0	The RSS_ring value for entry 64.
table_entry65	27:24	RW	0	The RSS_ring value for entry 65.
table_entry66	23:20	RW	0	The RSS_ring value for entry 66.
table_entry67	19:16	RW	0	The RSS_ring value for entry 67.
table_entry68	15:12	RW	0	The RSS_ring value for entry 68.
table_entry69	11:8	RW	0	The RSS_ring value for entry 69.
table_entry70	7:4	RW	0	The RSS_ring value for entry 70.
table_entry71	3:0	RW	0	The RSS_ring value for entry 71.

Indirection Table Register 10 (offset: 0x654)

Name	Bits	Access	Default Value	Description
table_entry72	31:28	RW	0	The RSS_ring value for entry 72.
table_entry73	27:24	RW	0	The RSS_ring value for entry 73.
table_entry74	23:20	RW	0	The RSS_ring value for entry 74.
table_entry75	19:16	RW	0	The RSS_ring value for entry 75.
table_entry76	15:12	RW	0	The RSS_ring value for entry 76.
table_entry77	11:8	RW	0	The RSS_ring value for entry 77.
table_entry78	7:4	RW	0	The RSS_ring value for entry 78.
table_entry79	3:0	RW	0	The RSS_ring value for entry 79.

Indirection Table Register 11 (offset: 0x658)

Name	Bits	Access	Default Value	Description
table_entry80	31:28	RW	0	The RSS_ring value for entry 80.
table_entry81	27:24	RW	0	The RSS_ring value for entry 81.
table_entry82	23:20	RW	0	The RSS_ring value for entry 82.
table_entry83	19:16	RW	0	The RSS_ring value for entry 83.
table_entry84	15:12	RW	0	The RSS_ring value for entry 84.
table_entry85	11:8	RW	0	The RSS_ring value for entry 85.
table_entry86	7:4	RW	0	The RSS_ring value for entry 86.
table_entry87	3:0	RW	0	The RSS_ring value for entry 87.

Indirection Table Register 12 (offset: 0x65C)

Name	Bits	Access	Default Value	Description
table_entry88	31:28	RW	0	The RSS_ring value for entry 88.
table_entry89	27:24	RW	0	The RSS_ring value for entry 89.
table_entry90	23:20	RW	0	The RSS_ring value for entry 90.
table_entry91	19:16	RW	0	The RSS_ring value for entry 91.
table_entry92	15:12	RW	0	The RSS_ring value for entry 92.
table_entry93	11:8	RW	0	The RSS_ring value for entry 93.
table_entry94	7:4	RW	0	The RSS_ring value for entry 94.
table_entry95	3:0	RW	0	The RSS_ring value for entry 95.

Indirection Table Register 12 (offset: 0x660)

Name	Bits	Access	Default Value	Description
table_entry96	31:28	RW	0	The RSS_ring value for entry 96.
table_entry97	27:24	RW	0	The RSS_ring value for entry 97.
table_entry98	23:20	RW	0	The RSS_ring value for entry 98.
table_entry99	19:16	RW	0	The RSS_ring value for entry 99.
table_entry100	15:12	RW	0	The RSS_ring value for entry 100.
table_entry101	11:8	RW	0	The RSS_ring value for entry 101.
table_entry102	7:4	RW	0	The RSS_ring value for entry 102.
table_entry103	3:0	RW	0	The RSS_ring value for entry 103.

Indirection Table Register 13 (offset: 0x664)

Name	Bits	Access	Default Value	Description
table_entry104	31:28	RW	0	The RSS_ring value for entry 104.
table_entry105	27:24	RW	0	The RSS_ring value for entry 105.
table_entry106	23:20	RW	0	The RSS_ring value for entry 106.
table_entry107	19:16	RW	0	The RSS_ring value for entry 99.
table_entry108	15:12	RW	0	The RSS_ring value for entry 100.
table_entry109	11:8	RW	0	The RSS_ring value for entry 101.
table_entry110	7:4	RW	0	The RSS_ring value for entry 102.
table_entry111	3:0	RW	0	The RSS_ring value for entry 103.

Indirection Table Register 14 (offset: 0x668)

Name	Bits	Access	Default Value	Description
table_entry112	31:28	RW	0	The RSS_ring value for entry 112.
table_entry113	27:24	RW	0	The RSS_ring value for entry 113.
table_entry114	23:20	RW	0	The RSS_ring value for entry 114.
table_entry115	19:16	RW	0	The RSS_ring value for entry 115.
table_entry116	15:12	RW	0	The RSS_ring value for entry 116.
table_entry117	11:8	RW	0	The RSS_ring value for entry 117.
table_entry118	7:4	RW	0	The RSS_ring value for entry 118.
table_entry119	3:0	RW	0	The RSS_ring value for entry 119.

Indirection Table Register 15 (offset: 0x66C)

Name	Bits	Access	Default Value	Description
table_entry120	31:28	RW	0	The RSS_ring value for entry 120.
table_entry121	27:24	RW	0	The RSS_ring value for entry 121.
table_entry122	23:20	RW	0	The RSS_ring value for entry 122.
table_entry123	19:16	RW	0	The RSS_ring value for entry 123.
table_entry124	15:12	RW	0	The RSS_ring value for entry 124.
table_entry125	11:8	RW	0	The RSS_ring value for entry 125.
table_entry126	7:4	RW	0	The RSS_ring value for entry 126.
table_entry127	3:0	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 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.
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	RW	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	RW	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	–
Programmable Extension Header Type #2	15:8	RW	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	RW	0	These bits contain the programmable extension header value for programmable header #1.

Statistics Registers

Note: Access method for these registers (A/C and 32bit).

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: 0x874)

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.

ifCRSERRORS (offset: 0x878)

The number of packets that have CRS errors.

iOUTDISCARDS (offset: 0x87C)

The number of packets that are discarded.

H2B Statistics Registers

There are two sets of H2B Statistics registers – though both sets are accessible from Host as well as APE, each side shall own a set. Set1 is owned by Host Driver and Set2 is owned by APE. HTX2B Stats Registers resides in TX-MAC and the B2HRX Stats Registers reside in RDI. A counter is automatically reset to 0 by HW upon a Read Operation by software. All counters roll over.

HTX2B Statistics

The HTX2B statistics reside in Transmit MAC. A counter is automatically reset to 0 by hardware upon a Read Operation. All counters roll over.

Statistics Name	*Set1 Address*	*Set2 Address*	*Size*	*Description*
HTX2B Octets	0x700	0x720	32	Total number of H2B octets.
HTX2B UCAST PKTS	0x704	0x724	32	Number of H2B unicast frames.
HTX2B MCAST PKTS	0x708	0x728	32	Number of H2B multicast.
HTX2B BCAST PKTS	0x70C	0x72C	32	Number of H2B broadcast frames.
HTX2B DROP PKTS	0x710	0x730	32	Number of H2B frames dropped by EMAC.
HTX2B DROP OCTETS	0x714	0x734	32	Total number of H2B octets dropped by EMAC.

B2HRX Statistics

The B2HRX statistics reside in RDI.

Statistics Name	*Set1 Address*	*Set2 Address*	*Size*	*Description*
B2HRX OCTETS	0x24D0	0x24E8	32	Total number of H2B octets.
B2HRX UCAST PKTS	0x24D4	0x24EC	32	Number of B2H unicast frames.
B2HRX MCAST PKTS	0x24D8	0x24F0	32	Number of B2H multicast frames.
B2HRX BCAST PKT	0x24DC	0x24F4	32	Number of B2H broadcast frames.
B2HRX DROP PKTS	0x24E0	0x24F8	32	Number of B2H frames dropped by RDI.
B2HRX DROP OCTETS	0x24E4	0x24FC	32	Total number of B2H octets dropped by RDI.

Receive MAC Static Counters

ifHCInOctets (offset: 0x880)

The number of octets received on the interface, including frame characters.

ifHCINOctets_bad (offset: 0x884)

The number of bad octets received on the interface.

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.

Ifnomorerxbd:0x224C

The number of times the NIC overran the Receive Buffer Descriptors.

Ifindiscard:0x2250

The number of inbound packets selected to be discarded (even though an error was not detected) to prevent the packets from being delivered to a higher layer protocol.

Ifinerror:0x2254

The number of inbound packets containing errors that prevented the packets from being delivered to a higher-layer protocol.

APE_NETWORK_STATS_REGS (Offsets 0x900–0x9BC)

These are duplicate network statistics registers meant for APE's private use. All of these registers accumulate their respective event-counts and the counts self clear to 0 when a GRC read accesses is performed on a register.

Statistics Name	*Address*	*Size*	*Description*
Transmit Statistics			
IFHCOUTOCTETS	0x900h	32	
ETHERSTATSCOL	0x908h	16	
OUTXONSENT	0x90Ch	16	
OUTXOFFSENT	0x910h	16	
MACXMITERRORS	0x0918	16	
SINGLECOL	0x091C	16	
MULTICOL	0x0920	16	
DEFERREDXMIT	0x0924	16	
EXCESSIVECOL	0x092C	16	
LATECOL	0x0930	16	
IFHCOUTUCAST	0x096C	32	
IFHCOUTMCAST	0x0970	32	
IFHCOUTBCAST	0x0974	32	
CRSERRORS	0x0978	16	
IFOUTDISCARDS	0x097C	16	
Receive Statistics			
IFHCINOCTETS_GOOD	0x0980	32	
IFHCINOCTETS_BAD	0x0984	32	
ETHERSTATSFRAGMENTS	0x0988	16	
IFHCINUCASTPKTS	0x098C	32	
IFHCINMULTICASTPKTS	0x0990	32	
IFHCINBROADCASTPKTS	0x0994	32	
DOT3STATSFCSERRORS	0x0998	16	
DOT3STATSALIGNMENTERRORS	0x099C	16	
XONPAUSEFRAMESRECEIVED	0x09A0	16	
XOFFPAUSEFRAMESRECEIVED	0x09A4	16	
MACCONTROLFRAMESRECEIVED	0x09A8	16	
XOFFSTATEENTERED	0x09AC	16	
DOT3STATSFRAMESTOOLONG	0x09B0	16	
ETHERSTATSJABBERS	0x09B4	16	
ETHERSTATSUNDERSIZEPKTS	0x09B8	16	

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	31:9	RO	0	–
Incorrect BD flag fix disable	8	RW	0	Disable fix for SDI sends incorrect last BD flag to SBDS.
Reserved	7:6	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 hardware 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	–
Statistics Clear	2	RW	0	If set, resets local statistics counters to zero. Clears only masked statistics. Self-clearing when done.
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	–
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 firmware can specify up to 64 KB, it should not attempt to program more than 8 KB because it would exceed the size of TXMBUF.

DMA Flag Register for TCP Segmentation (offset: 0xCEC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	RO	–	–
Reserved	25:20	RW	–	–
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 hardware if no firmware assisted TCP segmentation occurs. • Otherwise, firmware 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	–	–

Name	Bits	Access	Default Value	Description
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 hardware, this bit will be the same as bit 9 of the flag field in Send BD. • If the packet is created by firmware, it will be up to CPU whether it needs to post-process the data.
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. 0xCF4.31 is writable only if 0xCE8.15:0 is nonzero.
PASS Status	30	RO	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 hardware/firmware is servicing.

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	31:5	RO	0	–
Disable Delayed Host Coalescing	4	RW	0	A value 1 disables the Delayed HC feature introduced in BCM5718 family.
Reserved	3:2	RO	0	–
Enable	1	RW	1	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

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	31:4	RO	0	–
SBD consumer index fix disable	3	RW	0	Disable for sbd consumer index does not rollover for ring sizes 32,64,128, 256.
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	–
Index	8:0	RO		These bits contain the current NIC send BD index.

Each register applies to:

Name	*Description*
0x1440-0x1443 Send BD Ring Selector Local NIC	Send BD 1 Consumer Index RO.
0x1444-0x1447 Send BD Ring Selector Local NIC	Send BD 2 Consumer Index RO.
0x1448-0x144b Send BD Ring Selector Local NIC	Send BD 3 Consumer Index RO.
0x144c-0x144f Send BD Ring Selector Local NIC	Send BD 4 Consumer Index RO.
0x1450-0x1453 Send BD Ring Selector Local NIC	Send BD 5 Consumer Index RO.
0x1454-0x1457 Send BD Ring Selector Local NIC	Send BD 6 Consumer Index RO.
0x1458-0x145b Send BD Ring Selector Local NIC	Send BD 7 Consumer Index RO.
0x145c-0x145f Send BD Ring Selector Local NIC	Send BD 8 Consumer Index RO.
0x1460-0x1463 Send BD Ring Selector Local NIC	Send BD 9 Consumer Index RO.
0x1464-0x1467 Send BD Ring Selector Local NIC	Send BD 10 Consumer Index RO.
0x1468-0x146b Send BD Ring Selector Local NIC	Send BD 11 Consumer Index RO.
0x146c-0x146f Send BD Ring Selector Local NIC	Send BD 12 Consumer Index RO.
0x1470-0x1473 Send BD Ring Selector Local NIC	Send BD 13 Consumer Index RO.
0x1474-0x1477 Send BD Ring Selector Local NIC	Send BD 14 Consumer Index RO.
0x1478-0x147b Send BD Ring Selector Local NIC	Send BD 15 Consumer Index RO.
0x147c-0x147f Send BD Ring Selector Local NIC	Send BD 16 Consumer Index RO.

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*
	31.6			Unchanged
Reserved	31:5	RO	0	–
Multiple Send Ring mode	5	RW	0	Write a 1 to enable 16 Send Rings. Write a 0 to limit to a Single Send Ring.
	4:0			Unchanged
Pass_bit status	4	RW	0	Always return 1 when read.
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	1	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.

Each register applies to:

Table 100: Send BD Diagnostic Initiator

Send BD Initiator Local NIC	*Send BD Producer Index*
0x1808-0x180b	Send BD 1 Producer Index RO.
0x180c-0x180f	Send BD 2 Producer Index RO.
0x1810-0x1813	Send BD 3 Producer Index RO.
0x1814-0x1817	Send BD 4 Producer Index RO.
0x1818-0x181b	Send BD 5 Producer Index RO.
0x181c-0x181f	Send BD 6 Producer Index RO.
0x1820-0x1823	Send BD 7 Producer Index RO.
0x1824-0x1827	Send BD 8 Producer Index RO.
0x1828-0x182b	Send BD 9 Producer Index RO.
0x182c-0x182f	Send BD 10 Producer Index RO.
0x1830-0x1833	Send BD 11 Producer Index RO.
0x1834-0x1837	Send BD 12 Producer Index RO.
0x1838-0x183b	Send BD 13 Producer Index RO.
0x183c-0x183f	Send BD 14 Producer Index RO.
0x1840-0x1843	Send BD 15 Producer Index RO.
0x1844-0x1847	Send BD 16 Producer Index RO.

Send BD Fetch Threshold Register (offset: 0x1850)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RO	0x00	Unused
Send 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 uniformly applies to all 16 Send Rings. This parameter is meaningful only in the Multiple Send Ring mode. 0x1800[5] == 1

Send Mail Box Registers

15 more Host Send Producer Index registers are added to the High Priority Mail Box region as shown in the Table below. Similarly, 15 more NIC Send Producer Index registers are also added in the GRC space.

Table 101: Multiple Send Ring Mail Boxes

Send Ring #	*Host Send Ring Producer Index (High Priority Mail Box)*	*NIC Send Ring Producer Index*	*NIC Diagnostic Send Ring Consumer Index*	*Usable in*
Legacy /1	0x304	0x5980	0x3CC0	Legacy Mode, RSS Mode and IOV Mode
2	0x300	0x5984	0x3F04	RSS Mode and IOV Mode
3	0x30C	0x5988	0x3F08	
4	0x308	0x598C	0x3F0C	
5	0x314	0x5990	0x3F10	IOV Mode Only
6	0x310	0x5994	0x3F14	
7	0x31C	0x5998	0x3F18	
8	0x318	0x599C	0x3F1C	
9	0x324	0x59A0	0x3F20	
10	0x320	0x59A4	0x3F24	
11	0x32C	0x59A8	0x3F28	
12	0x328	0x59AC	0x3F2C	
13	0x334	0x59B0	0x3F30	
14	0x330	0x59B4	0x3F34	
15	0x33C	0x59B8	0x3F38	
16	0x338	0x59BC	0x3F3C	

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	1	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				–

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 non-empty 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 nonzero).

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	–
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	22:19	RO	0	–
Disable MACTQ Double Ack issue fix	18	RW	1	Disable MACTQ double ack issue fix. 1: Disabled 0: Enabled
Reserved	17:2	RO	N/A	–

Name	Bits	Access	Default Value	Description
Disable ASF Lockup Issue Fix	1	RW	1	Disable ASF Lockup Fix. 1: Disabled 0: Enabled
Reserved	0	RO	0	–

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 & 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 1 Count Registers (Offset: 0x2108)

These registers indicate how many frames are currently queued to the associated selector list.

Receive Selector List 2 Count Register (offset: 0x2118)

Receive Selector List 3 Count Registers (offset: 0x2128)

Receive Selector List 4 Count Registers (offset: 0x2138)

Receive Selector List 5 Count Registers (offset: 0x2148)

Receive Selector List 6 Count Registers (offset: 0x2158)

Receive Selector List 7 Count Registers (offset: 0x2168)

Receive Selector List 8 Count Registers (offset: 0x2178)

Receive Selector List 9 Count Registers (offset: 0x2188)

Receive Selector List 10 Count Registers (offset: 0x2198)

Receive Selector List 11 Count Registers (offset: 0x21a8)

Receive Selector List 12 Count Registers (offset: 0x21b8)

Receive Selector List 13 Count Registers (offset: 0x21c8)

Receive Selector List 14 Count Registers (offset: 0x21d8)

Receive Selector List 15 Count Registers (offset: 0x21e8)

Receive Selector List 16 Count Registers (offset: 0x21f8)

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	31:17	RO	0	–
Legacy	31:14		0	Defined by Legacy
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
Reserved	15	RO	0	–
No BD Discard Policy	14:13	RW	00	This field decides what to do with a packet in RXMBUF belonging to a generic VRQ, when not a single BD is available in the VRQ's BD Cache: 00 => The RDI shall discard an RX packet belonging to a VRQ, if no corresponding RX BD is available in the VRQ's BD Cache 01 => RDI shall place the RX packet in the Default VRQ instead 10 => RDI shall wait for a BD to become available in the VRQ's BD cache, no matter how long it takes. 11 => Reserved for future use. Ignored in non-IOV mode

Name	Bits	Access	Default Value	Description
HDS Look-Ahead Boundary	12:6	RW	0x00	This field indicates the size, in 1B increments, of the Header that the chip would replicate when HDS feature is enabled (meaningful only if bit[5] == 1) 0x00 => 128B 0x01 => 1B 0x02 => 2B …. …. 0x7F => 127B Ignored in non-IOV mode.
HDS Enable	5	RW	0	When this bit is written 1, the Header Data Split feature is enabled. This bit is meaningful only when IOV mode is enabled (0x400[5] == 1)
Legacy	4:0			Defined by Legacy.
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.
Jumbo Frame Enable	2	RW	0	Enable Jumbo Receive BD is needed and Jumbo Receive BD ring is disabled attention.
Reserved	1:0	RO	0	–

VRQ Status Register (offset: 0x240C)

Name	Bits	Access	Default Value	Description
Legacy	31:17	RU	0x0	Unused
VRQ Active Bit-Map	16:0	RW	0x0	A position of the Bit-Map shows a 1, in case the VRQ is currently enabled or recently disabled and is at the verge of being flushed, and thus there could possibly be associated traffic inside the chip. A bit-position shows 0, when a VRQ is completely disabled and there is no traffic whatsoever associated with it either inside the chip or in the PCIe link between the chip and the Root-Complex.

VRQ Flush Control Register (Offset: 0x2410)

Name	Bits	Access	Default Value	Description
S/W flush IOV vector	31:15	RW	0x0	S/W Flush vector [16:0].
Reserved	14:9	RW	0x0	Reserved
S/W flush reset request	8	RW/SC	0x0	Software writing 1 to this bit to force an internal index reset request. Bit-8 combining with [31:15] bit-map indicates which IOV index group to be reset. The function is identical to bit-1 hardware flush reset. This bit is self-clear.
Reserved	7:4	RW	0x0	Reserved
VRQ hardware flush drop enable	3	RW	0x0	1: Enable hardware drop packet function immediately when the IOV enable bits are disabled from driver. This bit will force RDI and WDMA engines to place the incoming packets into drop queue when the IOV enable bit is disabled in EMAC for that particular IOV child process.
VRQ status update and interrupt enable	2	RW	0x0	1: Enable hardware triggered host status interrupt and status block update request. RDI engine will generate a single pulse request signal to HC alone with current IOV status vector value for status block update.
VRQ hardware flush reset enable	1	RW	0x0	1: Enable hardware local index reset feature when VRQ flush timer expires. RDI engine will generate a single pulse clear request along with current IOV status [16:0] vector to various modules to clear local maintained index values for that particular IOV process.

Name	Bits	Access	Default Value	Description
VRQ hardware Flush enable	0	RW	0x0	This value will be loaded into a count-down counter triggered by a falling-edge of any bit of VRQ Enable register (0x560). The count-down is based on internal CORE_CLK. The purpose of this counter is to allow hardware to drain out certain pending DMA requests within WDMA pipeline and PCIE core. Once the timer expires, hardware shall invoke the Automatic VRQ Flush Procedure. A value 0x0 in this register effectively zeroes this timer count.

VRQ Flush Timer Register (offset: 0x2414)

Name	Bits	Access	Default Value	Description
IOV Flush Timer	31:0	RW	0x0	This value will be loaded into a count-down counter triggered by a falling-edge of any bit of VRQ Enable register (0x560). The count-down is based on internal CORE_CLK. The purpose of this counter is to allow hardware to drain out certain pending DMA requests within WDMA pipeline and PCIE core. Once the timer expires, hardware shall invoke the Automatic VRQ Flush Procedure. A value 0x0 in this register effectively zeroes this timer count.

RDI B2HRX Hardware Debugging Register (offset: 0x2418)

Name	Bits	Access	Default Value	Description
Legacy	31:0	RO	0x0	RDI internal B2HRX status.

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 Host Address Low Register (offset: 0x2444)

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)

Name	*Bits*	*Access*	*Default Value*	*Description*
Max Length	31:16	RW	0	Specifies the number of entries for Jumbo ring based on bit-mask (Supported values are 32,64,128,256,512,1024).
Reserved	15:2	RO	0	–
Disable Ring	1	RW	0	Set to disable the use of the ring.
Reserved	0	RO	0	–

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

Name	*Bits*	*Access*	*Default Value*	*Description*
NIC Address	31:0	RW	0x00044400	The NIC ring address is the NIC address of the first ring element. ***Note:*** For the BCM5718 family, do not initialize this register; leave as default.

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*
Max Length	31:16	RW	0	Programmable ring size: • 2048 • 1024 • 512 • 256 • 128 • 64 • 32
Reserved	15:2	RO	0	Maximum RX frame size.
Disable Ring	1	RW	0	1 = Ring Disabled 0 = Ring Enabled
Reserved	0	RO	0	Reserved

Receive Producer Ring NIC Address Register (offset: 0x245C)

Name	*Bits*	*Access*	*Default Value*	*Description*
NIC Address	31:0	RW	0x00040000	The NIC ring address is the NIC address of the first ring element. ***Note:*** For the BCM5718 family, do not initialize this register; leave as default.

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.

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 Data and Receive BD Initiator Hardware Diagnostic Register (offset: 0x24C0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Diagnostics	31:0	RO	0	Hardware Diagnostics

B2HRX Byte-count Statistics Count (offset: 0x24D0)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX OCTETS	31:0	CORW	0	Host B2HRX total packet byte count.

B2HRX Unicast Statistics Count (offset: 0x24D4)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX UCAST PKT	31:0	CORW	0	Host B2HRX unicast packet count.

B2HRX Multicast Statistics Count (offset: 0x24D8)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX MCAST PKT	31:0	CORW	0	Host B2HRX multicast packet count.

B2HRX Broadcast Statistics Count (offset: 0x24DC)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX BCAST PKT	31:0	CORW	0	Host B2HRX broadcast packet count.

B2HRX Drop Packet Count (offset: 0x24E0)

Name	Bits	Access	Default Value	Description
B2HRX DROP PKT	31:0	CORW	0	Host B2HRX packet drop count due to empty RBD.

B2HRX Drop Packet Byte Count (offset: 0x24E4)

Name	Bits	Access	Default Value	Description
B2HRX DROP OCTETS	31:0	CORW	0	Host B2HRX packet drop byte count due to empty RBD.

B2HRX APE Byte-count Statistics Count (offset: 0x24E8)

Name	Bits	Access	Default Value	Description
B2HRX OCTETS	31:0	CORW	0	APE B2HRX total packet byte count.

B2HRX APE Unicast Statistics Count (offset: 0x24EC)

Name	Bits	Access	Default Value	Description
B2HRX UCAST PKT	31:0	CORW	0	APE B2HRX unicast packet count.

B2HRX APE Multicast Statistics Count (offset: 0x24F0)

Name	Bits	Access	Default Value	Description
B2HRX MCAST PKT	31:0	CORW	0	APE B2HRX multicast packet count.

B2HRX APE Broadcast Statistics Count (offset: 0x24F4)

Name	Bits	Access	Default Value	Description
B2HRX BCAST	31:0	CORW	0	APE B2HRX broadcast packet count.

B2HRX APE Drop Packet Count (offset: 0x24F8)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX DROP PKT	31:0	CORW	0	APE B2HRX packet drop count due to empty RBD.

B2HRX APE Drop Packet Byte Count (offset: 0x24FC)

Name	*Bits*	*Access*	*Default Value*	*Description*
B2HRX DROP OCTETS	31:0	CORW	0	APE B2HRX packet drop byte count due to empty RBD.

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 Jumbo Receive BD Producer Index (offset: 0x2C08)

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

Receive BD Initiator Local NIC Receive BD Producer Index Register (offset: 0x2C0C–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)

All registers reset are core reset unless specified.

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.

Standard Replenish LWM Register (offset 0x2D00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Legacy	31:8	RU	0x0	Unused

Name	Bits	Access	Default Value	Description
Replenish Low Water Mark	7:0	RW	0x0	This field must be programmed with the desired LWM for Standard RBD cache/memory. The value equally applies to all VRQs. Whenever the number of BDs in a VRQ's BD memory falls below this (LWM+1), a BD fetch is request triggered. Recommended Settings: • Non-IOV mode: STD LWM: 32. (1/4 of total RBDs) • IOV mode: STD LWM: 16. (1/2 of total RBDs)

Jumbo Replenish LWM Register (offset 0x2D04)

Name	Bits	Access	Default Value	Description
Legacy	31:8	RU	0x0	Unused
Replenish Low Water Mark	7:0	RW	0x0	This field must be programmed with the desired LWM for Jumbo RBD cache/memory. The value equally applies to all VRQs. Whenever the number of BDs in a VRQ's BD memory falls below this (LWM+1), a BD fetch is request triggered. Recommended Settings: • Non-IOV mode: JMB LWM: 16. (1/4 of total RBDs) • IOV mode: JMB LWM: 4. (1/4 of total RBDs)

BD Fetch Limit Register (Offset 0x2D08)

This register is meaningful only in the IOV-Mode.

Table 102: BD Fetch Limit Register (Offset 0x2D08)

Name	*Bits*	*Access*	*Default Value*	*Description*
Legacy	31:5	RU	0x0	Unused
BD Fetch Limit	4:0	RW	0x1F	The number of BDs fetched by a single DMA request shall be the lesser of the following: • Space available in the respective BD cache. • Standard or Jumbo Replenish Threshold. • Number of BDs made available in the Host memory based Ring. • Programmed Value of this Field.

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.

Name	Bits	Access	Default Value	Description
Reserved	1:0	RO	0	–

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

All registers reset are core reset unless specified.

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 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	–	–

Central Power Management Unit (CPMU) Registers

CPMU Control Register (offset: 0x3600)

This register is reset by POR Reset except for Powerdown bit (bit #2)

Name	Bits	Access	Default Value	Description
Reserved	31:29	DC	0x0	–
Software controlled GPHY Force DLL on	28	RW	0x0	When this bit is enabled, GPHY DLL will be forced on by CPMU (unless the chip is in Low Power mode). This bit is intended for ASF.
Enable GPHY powerdown in D0u (this feature is not used in BCM5718 family)	27	RW	0x0	Enable CPMU to powerdown GPHY when the device enters D0u. 1: Enable 0: Disable.
Reserved	26:22	DC	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	21	DC	0x1	–
Reserved	20	DC	0x0	–
SGMII/PCS Powerdown	19	RW	0x0	Setting this bit will powerdown SGMII-PCS module.
Legacy Timer Enable	18	RW	0x0	This bit controls "cpmu_legacy_timer_enable" output. 1: Enable 0: Disable
Frequency Multiplier Enable / Disable (Reserved for BCM5719)	17	RW	0x1 for BCM5717 and BCM5718 0x0 for BCM5719	1: Enable 0: Disable
GPHY 10 MB Receive Only mode Enable	16	RW	0x0	Enables GPHY 10 MB Receive Only mode when this bit is set to 1.
Reserved	15	RW	0x0	–
Link Speed Power mode Enable	14	RW	0x0	Enable clock adjustment based on the link speed in mission mode.
Reserved	13	DC	0x0	–
Reserved	12:11	RW	0x0	–
Link Aware Power mode Enable	10	RW	0x0	Link Aware Power mode Enable. 1: Enable 0: Disable
Link Idle Power mode Enable	9	RW	0x0	Link Idle Power mode Enable. 1: Enable 0: Disable
Reserved	8:6	RW	0x0	–
APE Deep Sleep mode Enable	5	RW	0x0	Enable APE deep sleep power management mode.
APE Sleep mode Enable	4	RW	0x0	Enable APE sleep power management mode.
Reserved	3	RW	0x0	–
Power-down	2	RW	0x0	Legacy Address: 0x6804:[20] Force CPMU into Low Power State, LAN function will be powered down (GPHY, PCIE, APE). This bit is cleared by a rising edge of PERST_L.
CPMU Register Software Reset	1	RW SC	0x0	Software reset for resetting all the registers to default.
CPMU Software Reset	0	RW SC	0x0	Software reset for all the CPMU logic expect for registers.

Link Speed 10 MB/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	DC	0x000	–
MAC Clock Switch	20:16	RW	10111	Software Controlled MAC Core Clock Speed Select. 00000: Core = 62.5 MHz (GPHY DLL/2) 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) 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2)
Reserved	15:0	DC	0x0000	–

Link Speed 100 MB 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	DC	0x000	–
MAC Clock Switch	20:16	RW	10001	Software Controlled MAC Core Clock Speed Select. 00000: Core = 62.5 MHz (GPHY DLL/2) 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	DC	0x0000	–

Link Speed 1000 MB 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. This register shall be programmed to a different value only for engineering debug.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	DC	0x0	–
MAC Clock Switch	20:16	RW	0x0	Software Controlled MAC Core Clock Speed Select. 00000: Core = 62.5 MHz (GPHY DLL/2) 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	DC	0x0000	–

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, and the clock source of the clock policies programmed in this register must be the same as the host access clock policy register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:21	DC	0x000	–
MAC Clock Switch	20:16	RW	10111	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	15:13	RO	0x00	–
APE Clock Switch (Reserved in BCM5719)	12:8	RW	10001	Software Controlled APE Clock Speed Select 00001: 60.0 MHz (Alt Source/2) 00011: 30.0 MHz (Alt Source/4) 00101: 15.0 MHz (Alt Source/8) 00111: 7.5 MHz (Alt Source/16) 01001: 3.75 MHz (Alt Source/32) 10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16)
Reserved	7:0	DC	0x0000	–

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	DC	0x000	–
MAC Clock Switch	20:16	RW	10011	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	DC	0x0000	–

Link Idle Power Mode Clock Policy Register (offset: 0x3618)

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	DC	0x000	–
MAC Clock Switch	20:16	RW	10011	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)

Name	Bits	Access	Default Value	Description
Reserved	15:0	DC	0x0000	–

APE CLK Policy Register (offset: 0x361C)

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
(BCM5718) Reserved	31:13	DC	0x00000	–
(BCM5719) APE Sleep mode Enable	31	RW	0x0	Enable APE sleep power management mode.
(BCM5719) APE Deep Sleep mode Enable	30	RW	0x0	Enable APE deep sleep power management mode.
(BCM5719) APE Clock Speed Override Enable	29	RW	0x0	Enable APE clock speed override.
(BCM5719) Force APE FCLK Disable	28	RW	0x0	APE FCLK clock Disable. 1: Disable APE FCLK clock 0: Enable APE FCLK clock
(BCM5719) Force APE HCLK Disable	27	RW	0x0	APE HCLK Disable. 1: Disable APE HCLK clock 0: Enable APE HCLK clock
(BCM5719) APE Memory Bank 2 Deselect	26	RW	0x0	When this bit is set, deselect APE Scratchpad Memory Bank 2.
(BCM5719) APE Memory Bank 1 Deselect	25	RW	0x0	When this bit is set, deselect APE Scratchpad Memory Bank 1.
(BCM5719) APE Memory Bank 0 Deselect	24	RW	0x0	When this bit is set, deselect APE Scratchpad Memory Bank 0.
(BCM5719) Reserved	23:21	DC	0x0	–
(BCM5719) Clock Override APE Clock Switch	20:16	RW		Software Controlled APE Clock Speed Select for Clock Override. 00000: 62.5 MHz (NCSI DLL/2)10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16) 11110: 125 MHz

Name	*Bits*	*Access*	*Default Value*	*Description*
APE Clock Switch	12:8	RW	00000	Software Controlled APE Clock Speed Select. BCM5718: 00000: 62.5 MHz (CLK125/2) 00001: 60.0 MHz (Alt Source/2) 00011: 30.0 MHz (Alt Source/4) 00101: 15.0 MHz (Alt Source/8) 00111: 7.5 MHz (Alt Source/16) 01001: 3.75 MHz (Alt Source/32) 10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16) 11110: 83 MHz BCM5719: 00000: 62.5 MHz (NCSI DLL/2) 10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16) 11110: 125 MHz
Reserved	7:5	DC	0	–
LAPM APE Clock Switch	4:0	RW	10001	Software Controlled APE Clock Speed Select in Link Aware Power mode in BCM5719 10001: 25.0 MHz (CK25) (BCM5717) Reserved 10011: 12.5 MHz (CK25/2) (BCM5718) 10101: 6.25MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16)

APE Sleep State Clock Policy Register (offset: 0x3620)

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*
APE Sleep HCLK Disable	31	RW	0x1	Software Controlled APE HCLK shutoff in sleep and deep sleep state.
Reserved	30:21	DC	0x000	–
APE Deep Sleep FCLK Switch	20:16	RW	10001	Software Controlled APE Clock Speed Select. 00001: 60.0 MHz (Alt Source/2) 00011: 30.0 MHz (Alt Source/4) 00101: 15.0 MHz (Alt Source/8) 00111: 7.5 MHz (Alt Source/16) 01001: 3.75 MHz (Alt Source/32) 10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16)
Reserved	15:5	DC	0x000	–
APE Sleep FCLK Switch	4:0	RW	10001	Software Controlled APE Clock Speed Select. 00001: 60.0 MHz (Alt Source/2) 00011: 30.0 MHz (Alt Source/4) 00101: 15.0 MHz (Alt Source/8) 00111: 7.5 MHz (Alt Source/16) 01001: 3.75 MHz (Alt Source/32) 10001: 25.0 MHz (CK25) 10011: 12.5 MHz (CK25/2) 10101: 6.25 MHz (CK25/4) 10111: 3.125 MHz (CK25/8) 11001: 1.563 MHz (CK25/16)

Clock Speed Override Policy Register (offset: 0x3624)

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Clock Speed Override Enable	31	RW	0x0	Enable MAC clock speed override*. ***Note:*** For 5719 and 5720 only
Reserved	30:21	DC	0x000	–
MAC Clock Switch	20:16	RW	00000	Software Controlled MAC Core Clock Speed Select 00000: Core = 62.5 MHz (NCSI 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) For 5718 Software Controlled MAC Core Clock Speed Select00000: Core = 62.5MHz (GPHY DLL/2) 00001: Core = 60.0MHz (Alt Source/2) 00011: Core = 30.0MHz (Alt Source/4) 00101: Core = 15.0MHz (Alt Source/8) 00111: Core = 7.5MHz (Alt Source/16) 01001: Core = 3.75MHz (Alt Source/32) 10001: Core = 12.5MHz (CK25/2)10011: Core = 6.25MHz (CK25/4)10101: Core = 3.125MHz (CK25/8)10111: Core = 1.563MHz (CK25/16)11001: Core = 781KHz (CK25/32) 11111: Core = 12.5MHz/1.25MHz (MII_CLK/2)
Reserved	15:0	DC	0x0	–

Clock Override Enable Register (offset: 0x3628)

This register is reset by POR Reset or CPMU Register Software Reset. *The Force Disable bit has higher priority than Override Enable.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	DC	0x0	–
MAC Clock Speed Override Enable	13	RW	0x0	Enable MAC clock speed override*. 1: Enable 0: Disable
APE Clock Speed Override Enable	12	RW	0x0	Enable APE clock speed override*. 1: Enable 0: Disable

Name	Bits	Access	Default Value	Description
Flash Clock Speed Override Enable	11	RW	0x0	Enable Flash clock override*. 1: Enable 0: Disable
Reserved	10:7	DC	0x0	–
Force Flash Clock Disable	6	RW	0x0	Flash clock Disable*. 1: Disable Flash clock 0: Enable Flash clock
Reserved	5:2	DC	0x0	–
Force APE FCLK Disable	1	RW	0x0	APE FCLK clock Disable*. 1: Disable APE FCLK clock 0: Enable APE FCLK clock
Force APE HCLK Disable	0	RW	0x0	APE HCLK Disable*. 1: Disable APE HCLK clock 0: Enable APE HCLK clock

Status Register (offset: 0x362C)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31	RO	–	For 5719 four port devices, the bit31 will be used by FUNC_NUMBER. For 5718/5720 two port devices will not.
FUNC_NUMBER	31:30	RO	0x0	PCIE function number 1: function number 3: function 3 2: function 2 1: function 1 0: function 0 ***Note:*** CPMU port0 is tied to function 0, CPMU port1 is tied to function1. CPMU port2 is tied to function2. CPMU port3 is tied to functiion3.
FUNC_ENABLE	29:25	RO	–	Function Enable input from System BIOS.
APE status	24:23	RO	–	APE Engine Status 11: Reserved 10: Deep Sleep State 01: Sleep State 00: Active State
WOL ACPI Detection Enable Status of Port 1	22	RO	–	1: ACPI detection enabled 0: ACPI detection disabled

Name	*Bits*	*Access*	*Default Value*	*Description*
WOL Magic Packet Detection Enable Status of Port 1	21	RO	–	1: Magic Packet Detection enabled 0: Magic Packet Detection disabled
Ethernet link status	20:19	RO	–	EthernetLink Status 11: no link 10: 10 Mb 01: 100 Mb 00: 1000 Mb
Link idle status	18	RO	–	Link Idle status 1: Idle 0: Active
GPHY DLL lock status	17	RO	–	1: Locked 0: Not locked
NCSI DLL lock status	16	RO	–	1: Locked 0: Not locked
WOL ACPI Detection Enable Status of Port 0	15	RO	–	1: ACPI detection enabled. 0: ACPI detection disabled.
WOL Magic Packet Detection Enable Status of Port 0	14	RO	–	1: Magic Packet Detection Enabled. 0: Magic Packet Detection Disabled.
VMAIN power status	13	RO	–	VMAIN Power Status 1: On 0: Off
Reserved	12:10	DC	0x0	–
Power State	9:8	RO	–	Device Power State Status. 11: D3 00: D0
Energy Detect Status	7	RO	–	Energy Status 1: On 0: Off
CPMU Power State	6:4	RO	–	Indicates the current power state of the CPMU.
Power Management State Machine State	3:0	RO	–	Indicates the current state of hardware power management state machine.

Clock Status Register (offset: 0x3630)

Note: This is for BCM5718. See "Clock Status Register (offset: 0x3630)" on page 391 for BCM5719.

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	RO	–	Flash clock disable status
Reserved	28	DC	–	Reserved
Reserved	27:26	RO	–	–
APE HCLK Disable Status	25	RO	–	APE HCLK clock disable status
APE FCLK Disable Status	24	RO	–	APE FCLK clock disable status
PERST_N status	23	RO	–	PERST_N status
Reserved	22:21	DC	0x0	–
MAC Clock Switch Status	20:16	RO	–	MAC Core Clock Speed Select Status
Reserved	15:13	DC	0x0	–
APE Clock Switch Status	12:8	RO	–	APE Clock Speed Select Status
Reserved	7	DC	0x0	–
Flash Clock Switch Status	6:4	RO	–	Flash Clock Speed Select Status
Reserved	3:0	DC	0x0	–

Clock Status Register (offset: 0x3630)

Note: This is for BCM5719. See "Clock Status Register (offset: 0x3630)" on page 391 for BCM5718.

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	RO	–	Flash clock disable status
Reserved	28:27	DC	–	–
Reserved	26	RO	–	–
APE FCLK Disable Status	25	RO	–	APE FCLK clock disable status
APE HCLK Disable Status	24	RO	–	APE HCLK clock disable status
Reserved	23:21	DC	0x0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
MAC Clock Switch Status	20:16	RO	–	MAC Core Clock Speed Select Status
Reserved	15:13	DC	0x0	–
APE Clock Switch Status	12:8	RO	–	APE Clock Speed Select Status
Flash Clock Switch Status	7:5	RO	–	Flash Clock Speed Select Status
Reserved	4:0	DC	0x0	–

GPHY Control/Status Register (offset: 0x3638)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:14	DC	0x0	–
(BCM5719) Reserved	31:29	DC	0x0	–
(BCM5719) Keep NCSI PLL on During Low Power mode	28	RW	0x0	Keep NCSI PLL on during low power mode. 0: NCSI PLL is powered off in low power mode 1: NCSI PLL is kept running in low power mode
(BCM5719) Switching Regulator Power Down	27	RW	0x0	Switching regulator power down control bit 0: Switching regulator on 1: Switching regulator off
(BCM5719) TLP Clock Source	26	RW	0x0	TLP clock source mux control. 0: From PCIE SERDES 1: 125Mhz from NCSI PLL
(BCM5719) NCSI PLL Lock Status	25	RO	0x0	NCSI PLL Lock Status
(BCM5719) OTP SERDES PLL Lock Status	24	RO	0x0	OTP SERDES PLL Lock Status
(BCM5719) GPHY PLL Lock Status	23	RO	0x0	GPHY PLL Lock Status
(BCM5719) PCIE SERDES PLL Lock Status	22	RO	0x0	PCIE SERDES PLL Lock Status
(BCM5719) NCSI PLL Test Select	21:18	RW	0x0	NCSI PLL Test Select
(BCM5719) NCSI PLL Test Enable	17	RW	0x0	NCSI PLL Test Enable
(BCM5719) NCSI PLL Power Down	16	RW	0x0	NCSI PLL Power Down
(BCM5719) SGMII/PCS Powerdown	15	RW	0x0	Setting this bit will powerdown SGMII-PCS module.
(BCM5719) SGMII/PCS Reset	14	RW	0x0	Setting this bit will reset SGMII-PCS module.
GPHY 10 MB Receive Only mode TX Idle Debounce Timer	13:12	RW	0x1	10 MB Receive Only mode TX Idle Debounce Timer. 0x1 = 6 µs
(BCM5718) Reserved	11	DC	0x0	–
(BCM5719) Software controlled GPHY Force DLL on	11	RW	0x0	When this bit is enabled, GPHY DLL will be forced on by CPMU (unless the chip is in Low Power mode). This bit is intended for ASF.
GPHY DLL IDDQ state	10	RO	0x0	Gphy_iddq_dll_act state
GPHY pwrdn	9	RO		CPMU controlled output to GPHY
GPHY set_bias_iddq	8	RO		CPMU controlled output to GPHY

Name	Bits	Access	Default Value	Description
GPHY force_dll_on	7	RO		CPMU controlled output to GPHY
GPHY dll_pwrdn_ok	6	RO		CPMU controlled output to GPHY
SW controlled POR to GPHY	5	RW	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.
(BCM5717 and BCM5718) Reserved	4	DC	0x0	–
(BCM5719) Power-down	4	RW	0x0	Legacy Address: 0x6804:[20] Force CPMU into Low Power State, LAN function will be powered down (GPHY, PCIE, IPSEC, APE). This bit is cleared by a rising edge of PERST_L.
(BCM5717 and BCM5718) Reserved	3	DC	0x0	–
CPMU Register Software Reset	3	RW SC	0x0	Software reset for resetting all the registers to default.
Reserved	2	DC	0x0	–
CPMU Software Reset	2	RW	0x0	Software reset for all the CPMU logic expect for registers.
BIAS IDDQ	1	RW	0x0	Legacy Address: 0x6804:[22] When this bit is set, BIAS will be powered down.
GPHY IDDQ	0	RW	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	RW	0x0	Legacy Address: 0x6804:[24] Core RAM power down
BD RAM Power down	30	RW	0x0	Legacy Address: 0x6804:[24] BD RAM power down
Reserved	29:0	DC	0	–

Core Idle Detection De-Bounce Control Register (offset: 0x3648)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Link Idle Detection De-bounce Timer	31:0	RW	0x20	De-bounce timer setting for core link from active to non-active. Unit is in number of CPMU clock cycles Range: up to 232 CPMU clock cycles Default Value (10 CPMU CLK) multiplied by given Core CLK scale parameter below. x1 00000: Core = 62.5 MHz (GPHY DLL/2) x1 00001: Core = 60.0 MHz (Alt Source/2) x2 00011: Core = 30.0 MHz (Alt Source/4) x4 00101: Core = 15.0 MHz (Alt Source/8) x8 00111: Core = 7.5 MHz (Alt Source/16) x16 01001: Core = 3.75 MHz (Alt Source/32) 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) x128 11001: Core = 781 kHz (CK25/32), x64 11111: Core = 12.5 MHz/1.25 MHz (MII_CLK/2)

PCIE Idle Detection De-Bounce Control Register (offset: 0x364C)

This register is reset by POR Reset or CPMU Register Software Reset. This register is shared by all MAC ports. User must first gain grant from the global MUTEX registers (0x36F0 & 0x36F4) before writing to it.

Name	*Bits*	*Access*	*Default Value*	*Description*
PCIE Idle Detection De-bounce Timer	31:0	RW	0xD for BCM5718 0x19 for BCM5719	De-bounce timer setting for PCIE link from active to non-active. Unit is in number of CPMU clock cycles. Range: up to 232 CPMU clock cycles. Default: 10 CPMU clocks.

Energy Detection De-Bounce Timer (offset: 0x3650)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:10	RO	0	–
Energy Detect Select	9	RW	0x0	Legacy Address: 0x6890:[26] 1: use energy_det_apd from GPHY core 0: use output from the energy debounce logic
Select HW_Energy_Det	8	RW	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 De-bounce Logic 0: Select Combination of Software Force_Energy_Det and Output of De-bounce Logic to generate System Energy_Det
Select SW_HW_Oring_Energy_Det	7	RW	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. 1: Generate System Energy_Det by Oaring the SW_Force_Energy_Det with the Output of the De-bounce Logic 0: Generate System Energy_Det based on the SW_Force_Energy_Det Output

Name	*Bits*	*Access*	*Default Value*	*Description*
SW_Force_Energy_Det_Value	6	RW	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 De-bounce Low	5	RW	0x1	This bit is used to disable the Energy_Det_Debounce_Low Logic from de-bouncing 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 De-bounce Low 1: Disable De-bounce Low
Disable Energy_Det De-bounce High	4	RW	0x1	This bit is used to disable the Energy_Det_Debounce_High Logic from de-bouncing 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 De-bounce High 1: Disable De-bounce High
Energy_Det De-bounce High Limit	3:2	RW	0x0	This parameter is used to control the de-bounce 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)
Energy_Det De-bounce Low Limit	1:0	RW	0x1	This parameter is used to control the de-bounce 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*
(BCM5718) Alternate Clock Lock Timer	31:16	RW	0x61FF	This parameter is used to preset the Alternate DLL Lock Timer. This timer is always enabled. Unit is in number of CPMU clock cycles Range: up to 2^{16} CPMU clock cycles Default: 25088 CPMU clocks (1ms if CPMU clock frequency is 25 MHz)
(BCM5719) NCSI PLL Lock Timer	31:16	RW	0xF	This parameter is used to preset the NCSI PLL Lock Timer. This timer is always enabled. Unit is in number of CPMU clock cycles Range: up to 2^{16} CPMU clock cycles Default: 15 CPMU clocks
Reserved	15:11	DC	0	–
GPHY DLL Lock Timer	10:0	RW	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*
Reserved	31:28	DC	0x0	–
Chip ID	27:12	RO	0x5717 0x5718 0x5719	–
Base Layer Revision Information	11:8	RO	0000	Base layer revision history 0000: A0 0001: B0 0010: C0
Metal Layer Revision Information	7:0	RO	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	31:16	DC	0	–
Set Request/Request Pending	15:0	RW1S	0x0	Writing a 1 to any of these bits pends a Mutex lock request on behalf of a software agent. The bit is subsequently latched by hardware 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.

Mutex Grant Register (offset: 0x3660)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:16	DC	0	–
Set Request/Request Pending	15:0	RW1S	0x0	Reading this field 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 one, and the pairing bit in the Mutex_Request_Reg shall be cleared. Writing a 0 to any bits has no effect.

GPHY Strap Register (offset: 0x3664)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	DC	0x0	Readable and writeable reserved bits.
APE CM3 Big Endian Enable	8	RW	0x0	Enable APE CM3 Big Endian Setting.
Reserved	7:6	DC	0x0	Readable and writeable reserved bits.

Name	Bits	Access	Default Value	Description
Current port strap value	5	RW	0x0	This bit is used by the APE to deposit the current port's strap value decoded from the CPMU debug register. Please note, the bit is only applicable to BCM5718 family A0 when APE path code is enabled. 0x0: GPHY mode 0x1: SERDES mode
RXCPU SPAD ECC ENABLE	4	RW	0x0	Enable ECC for rxcpu scratchpad.
RXMBUF ECC ENABLE	3	RW	0x0	Enable RXMBUF ECC.
TXMBUF ECC ENABLE	2	RW	0x0	Enable TXMBUF ECC.
Debug UART Port Switch	1	RW	0x0	Invert the current value of this bit to switch to the other port.
Mission board SerDes mode enable	0	RW	0x0	Enable SerDes test mode on mission board.

Padring Control Register (offset: 0x3668)

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:27	DC	0x0	Readable and writeable reserved bits.
APE Status	26	RW	0	This bit is used by bootcode to post the APE initialization status.
Eclk switch using link status Disable	25	RW	0x0 BCM5718 0x1 BCM5719	Switch emac clocks to core clock based on lnk state. 0: Enable 1: Disable
OOB Power Consumption FIX for 5784 Option 4	24	RW	0x0	Enable fix for OOB Power Consumption FIX for 5784 option 4. 1 = Enable 0 = Disable
OOB Power Consumption FIX for 5784 Option 3	23	RW	0x0	Disable fix for OOB Power Consumption FIX for 5784 option 3. 1 = Enable 0 = Disable
OOB Power Consumption FIX for 5784 Option 2	22	RW	0x0	Disable fix for OOB Power Consumption FIX for 5784 option 2. 1 = Disable 0 = Enable

Name	Bits	Access	Default Value	Description
OOB Power Consumption FIX for 5784 Option 1	21	RW	0x0	Enable fix for OOB Power Consumption FIX for 5784 option 1. 1 = Enable 0 = Disable
Incorrect checksum on LSO packets Fix Disable	20	RW	0x0	Disable fix for incorrect checksum on LSO packets. 1 = Disable 0 = Enable
PCIE pcie_tmux_sel[1:0]	19:18	RW	0x0	–
Reserved	17:13	DC	0	–
PLLisUp signal drive	12	RW	0x0	When 0, force the PLLisUp signal to be 1. When 1, let the hardware drive the PLLisUp signal.
PCIE pcie_tmux_sel[3:2]	11:10	RW	0x0	–
Reserved	9	RW	0	–
Capability version for completion timeout ECN for PCIE 1.1	8	RW	0x0	1 = Version 1; Fix Disable 0 = Version 2; Fix Enable
Reserved	7:0	DC	0x0	–

Flash Clock Policy Register (offset: 0x366C)

For BCM5719/5720 only. This register is reset by POR Reset or CPMU Register Software Reset. The Force Disable bit has higher priority than Override Enable. This register is shared by all 4 MAC ports. User must first gain grant from the global MUTEX registers (0x36F0 & 0x36F4) before writing to this register.

Name	Bits	Access	Default Value	Description
Flash Clock Speed	31	RW	0x0	Enable Flash clock override*. Override Enable 1: Enable Flash clock override 0: Disable Flash clock override
Force Flash Clock Disable	30	RW	0x0	Flash clock Disable*. 1: Disable Flash clock 0: Enable Flash clock
Flash Idle mode Enable	29	RW	0x0	Flash Idle mode Enable 1: Enable Flash Idle mode 0: Disable Flash Idle mode
Force EAV Clock Disable	28	RW	0x0	EAV clock Disable 1: Disable EAV clock 0: Enable EAV clock

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	27:20	DC	0	–
EAV Clock Policy	19:12	RW	0x0	Software Controlled EAV Clock Speed Select 00000000: divide-by-256 -- Clock frequency 4.8 MHz 00000001: divide-by-1 -- Clock frequency 1250 MHz (Unused for EAV) 00000010: divide-by-2 -- Clock frequency 625 MHz (Unused for EAV) ... 00001010: divide-by-10 -- Clock frequency of 125 MHz 00001011: divide-by-11 -- Clock frequency of 113.6 MHz .... 11111111=divide-by-255 -- Clock frequency of 4.9 MHz
Reserved	11	DC	0	–
Flash Idle Clock Policy	10:8	RW	0x2	Software Controlled Flash Clock Speed Select000: 62.5 MHz (NCSI DLL) 010: 25 MHz (CK25)
Reserved	7	DC	0	–
Flash Clock Policy	6:4	RW	0x0	Software Controlled Flash Clock Speed Select000: 62.5 MHz (NCSI DLL) 010: 25 MHz (CK25)
Reserved	3	DC	0	–
Override Flash Clock Switch	2:0	RW	0x2	Software Controlled Flash Clock Speed Select000: 62.5 MHz (NCSI DLL) 010: 25 MHz (CK25)

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	31:25	DC	0	Readable and writeable reserved bits.
PCIE Idle	24	RW	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
APE ATP Empty	23	RW	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.
APE ATPM Idle	22	RW	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	RW	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	RW	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	RW	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	RW	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	RW	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*
Reserved	16	DC	0	–
WDMA Idle	15	RW	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	RW	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	RW	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	RW	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	RW	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	RW	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	RW	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	RW	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*
RXER = 0	7	RW	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.
RXDV = 0	6	RW	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	RW	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.
FTQ empty	4	RW	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	RW	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.
MBUF empty	2	RW	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	RW	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.
No core reset	0	RW	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*	*Default Value*	*Description*
Reserved	31:25	DC	—	Readable and writeable reserved bits
PCIE Idle	24	RO	—	Idle Status. 1: Idle 0: Busy
APE ATP Empty	23	RO	—	
APE ATPM Idle	22	RO	—	
DBU Idle	21	RO	—	
NVM Idle	20	RO	—	
SBDI Idle	19	RO	—	
RBDI Idle	18	RO	—	
MB Idle	17	RO	—	
Reserved	16	DC	—	–
WDMA Idle	15	RO	—	Idle Status. 1: Idle 0: Busy
RDMA Idle	14	RO	—	
MSI Idle	13	RO	—	
TXMAC FIFO empty	12	RO	—	
RXMAC FIFO empty	11	RO	—	
COL = 0	10	RO	—	
CRS = 0	9	RO	—	
TXAMAC Idle	8	RO	—	
RXER = 0	7	RO	—	
RXDV = 0	6	RO	—	
MDIO Idle	5	RO	—	
FTQ empty	4	RO	—	
GRC Idle	3	RO	—	
MBUF empty	2	RO	—	
MA Idle	1	RO	—	
No core reset	0	RO	—	

Top Level Miscellaneous Control 1 Register (offset: 0x367C)

For BCM5719/5720 only. This register is reset by POR Reset or CPMU Register Software Reset. This register is shared by 4 MAC ports. User must first gain grant from the global MUTEX registers (0x36F0 & 0x36F4) before writing to this register.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:6	RW	0	–
Low power IDDQ mode	5	RW	0	1–Enable to put all ports in GPHY mode during low power IDDQ to select GPHY mode to save power. 0–Serdes mode in low power IDDQ mode by default, draws more current
NCSI Clock output disable	4	RW	0	1–To enable the NCSI clock disable feature, NCSI clock signal pin is driven low 0–NCSI clock output active (default)
47147 fix disable	3	RW	0	47147 fix disable–PCIe MDIO reset fix disable. 1–Disable 0–Enable
47173 fix disable	2	RW	0	47173 fix disable–CLKREQ_L tri-state fix disable. 1–Disable 0–Enable
Debug UART Port Selection	1	RW	0	Select the Debug UART Port. 00: port0 01: port1 10: port2 11: port3

Miscellaneous Control Register (offset: 0x36AC)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:26	DC	0x0	–
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
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 10.240 µS. 0x0: no delay 0x1: 10.24 µs delay 0x2: 20.48 µs delay .. .. 0xFF: 2621.44 µs 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 µs with no delay 1000 mb/s 64byte packet–advances l1aspm exiting by 2.488 µs with no delay 100 mb/s 1518byte packet–advances l1aspm exiting by 129.377 µs with no delay 100 mb/s 64byte packet–advances l1aspm exiting by 13.217 µs with no delay 10 mb/s 1518byte packet–advances l1aspm exiting by 1274.94 µs with no delay 10 mb/s 64byte packet–advances l1aspm exiting by 113.98 µs with no delay
Reserved	15:0	DC	0x0	–

EEE Mode Register (offset: 0x36B0)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:20	RO	0x0	–
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	RW	0x0	Block out time from de-assertion of EEE MAC TX request to when the EEE module considers new request to enter LPI again.
Auto Wake Enable	10	RW	0x0	Enable automatic removal of Link from LPI state when MAC wants to transmit.
RX LPI Enable	9	RW	0x0	Enable LPI indication in RX direction.
TX LPI Enable	8	RW	0x0	Enable LPI indication in TX direction.
User LPI Enable	7	RW	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 µs.
EEE Link Idle Debounce Timer	15:0	RW	0x3F	EEE link idle debounce timer in us. Default is 63 µs.

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 µs.
Send Index Equal Debounce Timer	15:0	RW	0x3F	Send producer and consumer index equal debounce timer in µs. Default is 63 µs.

EEE Link Idle Control Register (offset: 0x36BC)

Name	Bits	Access	Default Value	Description
Reserved	31:25	RO	0x0	–
PCIE not in L0 State	24	RW	0x0	PCIE 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.IE is in L0s, L1 or L2 state.
Reserved	23:4	RW	0x0	–
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.
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	31:25	RO	0x0	–
PCIE not in L0 State	24	RO	0x0	PCIE is in L0s, L1 or L2 state.
Reserved	23:4	RO	0x0	–
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 is empty.
LAN TX Packet Buffer Empty	0	RO	0x0	Internal packet buffers in LAN core for TX is empty.

EEE Statistic Counter 1 Register (offset: 0x36C4)

Name	Bits	Access	Default Value	Description
EEE Mode Entering Counter	31:0	RW	0x0	This counter counts the number of times 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	RW	0x0	This counter counts the number of times the receive side goes into LPI. 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	RW	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	RW	0x0	The minimum time from start of LPI message transmission to end of LPI message transmission.
EEE Exit Time	15:0	RW	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	RW	0x0	–
Freeze 2s Average	4	RW	0x0	Stop the update of the 2 second average register. The accumulation and holding registers will still update.
Freeze Count	3	RW	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	RW	0x0	Restart the calibration process and capture a single 500 µs accumulation period.
Current Measurement Read Select	1:0	RW	0x0	0x0: Current Measurement Upper 32-bit Read Register = 0x0 Current Measurement Lower 32-bit Read Register = 500 µs 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	RO	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	RO	0x0	Please refer to 0x36D4[1:0]

Global Mutex Request Register (offset: 0x36F0)

Note: This register is for BCM5719/BCM5720 only.

This register is reset by POR Reset or CPMU Register Software Reset.

Name	Bits	Access	Default Value	Description
Reserved	31:8	DC	0	–
Set Request/Request Pending	7:0	RW 1S	0x0	Writing a 1 to any of these bits pends a Mutex lock request on behalf of a s/w agent. The bit is subsequently latched by hardware 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.

Global Mutex Grant Register (offset: 0x36F4)

Note: This register is for BCM5719/BCM5720 only.

This register is reset by POR Reset or CPMU Register Software Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:8	DC	0	–
Set Request/Request Pending	7:0	RW 1S	0x0	Reading this field 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, 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 paring bit in the Mutex_Request_Reg shall be cleared. Writing a 0 to any bits has no effect.

Temperature Monitor Control Register (offset: 0x36FC)

Note: This register is for BCM5719 and BCM5720 only.

This register is reset by POR Reset or CPMU Register Software Reset. This register is shared by 4 MAC ports. User must first gain grant from the global MUTEX registers (0x36F0 & 0x36F4) before writing to it.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:19	RO	0	–
Temperature Monitor Power Down	18	RW	0	Temperature Monitor Power Down
Temperature Monitor Hold	17	RW	0	Temperature Monitor Hold
Temperature Data	16:8	RO	0	Temperature Data
Bias Adjust	7:1	RW	0	Bias Adjust
ADC Test Enable	0	RW	1	ADC Test Enable

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*
Reserved	31:13	RO	0	–
No Interrupt on DMAD Force	12	RW	–	When set, the COAL_NOW bit of the buffer descriptor may be set to force a status block update without a corresponding interrupt.
No Interrupt on Force update	11	RW	–	When set, writing the Coalesce Now bit will cause a status without a corresponding interrupt event.
Reserved	10	RO	0	When set, the TX Host Coalescing Tick counter initializes to the idle state and begins counting only after a transmit BD event is detected.
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.
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.

Name	Bits	Access	Default Value	Description
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 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.

IOV 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

Receive Coalescing Ticks Register for VRQ 5 => 0x3DE0

Receive Coalescing Ticks Register for VRQ 6 => 0x3DF8

Receive Coalescing Ticks Register for VRQ 7 => 0x3E10

Receive Coalescing Ticks Register for VRQ 8 => 0x3E28

Receive Coalescing Ticks Register for VRQ 9 => 0x3E40

Receive Coalescing Ticks Register for VRQ 10 => 0x3E58

Receive Coalescing Ticks Register for VRQ 11 => 0x3E70

Receive Coalescing Ticks Register for VRQ 12 => 0x3E88

Receive Coalescing Ticks Register for VRQ 13 => 0x3EA0

Receive Coalescing Ticks Register for VRQ 14 => 0x3EB8

Receive Coalescing Ticks Register for VRQ 15 => 0x3ED0

Receive Coalescing Ticks Register for VRQ 16 => 0x3EE8

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 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.

IOV + Multiple TXQ:

Send Coalescing Ticks Register for TXQ 0 => 0x3C0C

Send Coalescing Ticks Register for TXQ 1 => 0x3D84

Send Coalescing Ticks Register for TXQ 2 => 0x3D9C

Send Coalescing Ticks Register for TXQ 3 => 0x3DB4

Send Coalescing Ticks Register for TXQ 4 => 0x3DCC

Send Coalescing Ticks Register for TXQ 5 => 0x3DE4

Send Coalescing Ticks Register for TXQ 6 => 0x3DFC

Send Coalescing Ticks Register for TXQ 7 => 0x3E14

Send Coalescing Ticks Register for TXQ 8 => 0x3E2C

Send Coalescing Ticks Register for TXQ 9 => 0x3E44

Send Coalescing Ticks Register for TXQ 10 => 0x3E5C

Send Coalescing Ticks Register for TXQ 11 => 0x3E74

Send Coalescing Ticks Register for TXQ 12 => 0x3E8C

Send Coalescing Ticks Register for TXQ 13 => 0x3EA4

Send Coalescing Ticks Register for TXQ 14 => 0x3EBC

Send Coalescing Ticks Register for TXQ 15 => 0x3ED4

Send Coalescing Ticks Register for TXQ 16 => 0x3EEC

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.

When IOV 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

Receive Max Coalesced BD Count Register for VRQ 5 => 0x3DE8

Receive Max Coalesced BD Count Register for VRQ 6 => 0x3E00

Receive Max Coalesced BD Count Register for VRQ 7 => 0x3E18

Receive Max Coalesced BD Count Register for VRQ 8 => 0x3E30

Receive Max Coalesced BD Count Register for VRQ 9 => 0x3E48

Receive Max Coalesced BD Count Register for VRQ 10 => 0x3E60

Receive Max Coalesced BD Count Register for VRQ 11 => 0x3E78

Receive Max Coalesced BD Count Register for VRQ 12 => 0x3E90

Receive Max Coalesced BD Count Register for VRQ 13 => 0x3EA8

Receive Max Coalesced BD Count Register for VRQ 14 => 0x3EC0

Receive Max Coalesced BD Count Register for VRQ 15 => 0x3ED8

Receive Max Coalesced BD Count Register for VRQ 16 => 0x3EF0

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.

IOV + Multiple TXQ:

Send Max Coalesced BD Count Register for TXQ 0 => 0x3C14

Send Max Coalesced BD Count Register for TXQ 1 => 0x3D8C

Send Max Coalesced BD Count Register for TXQ 2 => 0x3DA4

Send Max Coalesced BD Count Register for TXQ 3 => 0x3DBC

Send Max Coalesced BD Count Register for TXQ 4 => 0x3DD4

Send Max Coalesced BD Count Register for TXQ 5 => 0x3DEC

Send Max Coalesced BD Count Register for TXQ 6 => 0x3E04

Send Max Coalesced BD Count Register for TXQ 7 => 0x3E1C

Send Max Coalesced BD Count Register for TXQ 8 => 0x3E34

Send Max Coalesced BD Count Register for TXQ 9 => 0x3E4C

Send Max Coalesced BD Count Register for TXQ 10 => 0x3E64

Send Max Coalesced BD Count Register for TXQ 11 => 0x3E7C

Send Max Coalesced BD Count Register for TXQ 12 => 0x3E94

Send Max Coalesced BD Count Register for TXQ 13 => 0x3EAC

Send Max Coalesced BD Count Register for TXQ 14 => 0x3EC4

Send Max Coalesced BD Count Register for TXQ 15 => 0x3EDC

Send Max Coalesced BD Count Register for TXQ 16 => 0x3EF4

Receive Max Coalesced BD Count During Interrupt Register (offset: 0x3C18)

This register has the same attribute that of 0x3C10 except that this parameter is active only during the During Interrupt state – which is the state during which the ISR has acknowledged an interrupt by writing a non-zero value to the MailBox register and thus the interrupt is in a masked state. If this parameter triggers (while in During Interrupt state), the chip will DMA the latest Status Block to the host memory, but the interrupt will remain de-asserted.

When IOV 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

Receive Max Coalesced BD Count During Interrupt Register for VRQ 5 => 0x3DF0

Receive Max Coalesced BD Count During Interrupt Register for VRQ 6 => 0x3E08

Receive Max Coalesced BD Count During Interrupt Register for VRQ 7 => 0x3E20

Receive Max Coalesced BD Count During Interrupt Register for VRQ 8 => 0x3E38

Receive Max Coalesced BD Count During Interrupt Register for VRQ 9 => 0x3E50

Receive Max Coalesced BD Count During Interrupt Register for VRQ 10 => 0x3E68

Receive Max Coalesced BD Count During Interrupt Register for VRQ 11 => 0x3E80

Receive Max Coalesced BD Count During Interrupt Register for VRQ 12 => 0x3E98

Receive Max Coalesced BD Count During Interrupt Register for VRQ 13 => 0x3EB0

Receive Max Coalesced BD Count During Interrupt Register for VRQ 14 => 0x3EC8

Receive Max Coalesced BD Count During Interrupt Register for VRQ 15 => 0x3EE0

Receive Max Coalesced BD Count During Interrupt Register for VRQ 16 => 0x3EF8

Send Max Coalesced BD Count During Interrupt Register (offset: 0x3C1C)

This register has the same attribute that of 0x3C14 except that this parameter is active only during the During Interrupt state – which is the state during which the ISR has acknowledged an interrupt by writing a non-zero value to the MailBox register and thus the interrupt is in a masked state. If this parameter triggers (while in During Interrupt state), the chip will DMA the latest Status Block to the host memory, but the interrupt will remain de-asserted.

IOV + Multiple TXQ:

Send Max Coalesced BD Count During Interrupt Register for TXQ 0 => 0x3C1C

Send Max Coalesced BD Count During Interrupt Register for TXQ 1 => 0x3D94

Send Max Coalesced BD Count During Interrupt Register for TXQ 2 => 0x3DAC

Send Max Coalesced BD Count During Interrupt Register for TXQ 3 => 0x3DC4

Send Max Coalesced BD Count During Interrupt Register for TXQ 4 => 0x3DDC

Send Max Coalesced BD Count During Interrupt Register for TXQ 5 => 0x3DF4

Send Max Coalesced BD Count During Interrupt Register for TXQ 6 => 0x3E0C

Send Max Coalesced BD Count During Interrupt Register for TXQ 7 => 0x3E24

Send Max Coalesced BD Count During Interrupt Register for TXQ 8 => 0x3E3C

Send Max Coalesced BD Count During Interrupt Register for TXQ 9 => 0x3E54

Send Max Coalesced BD Count During Interrupt Register for TXQ 10 => 0x3E6C

Send Max Coalesced BD Count During Interrupt Register for TXQ 11 => 0x3E84

Send Max Coalesced BD Count During Interrupt Register for TXQ 12 => 0x3E9C

Send Max Coalesced BD Count During Interrupt Register for TXQ 13 => 0x3EB4

Send Max Coalesced BD Count During Interrupt Register for TXQ 14 => 0x3ECC

Send Max Coalesced BD Count During Interrupt Register for TXQ 15 => 0x3EE4

Send Max Coalesced BD Count During Interrupt Register for TXQ 16 => 0x3EFC

HC Parameter Set Reset Register (Offset: 0x3C28)

Place this in HC block.

Table 103: HC Parameter Set Reset Register (Offset: 0x3C28)

Name	*Bits*	*Access*	*Default Value*	*Description*
Legacy	31:17	RU	0x0	Unused
HC Parameter Set Reset Bit-Map	16:0	RW	0x0	Write a 1 to the position of this bit map to clear the respective HC Parameter Register Set – the associated internal HC states are also cleared by this action. This is useful for VRQ Flush-Synch purposes while in IOV Mode.

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.

When IOV is enabled:

Status Block 0 Host Address Register (offset: 0x3C38)

Status Block 1 Host Address Register (offset: 0x3D00)

Status Block 2 Host Address Register (offset: 0x3D08)

Status Block 3 Host Address Register (offset: 0x3D10)

Status Block 4 Host Address Register (offset: 0x3D18)

Status Block 5 Host Address Register (offset: 0x3D20)

Status Block 6 Host Address Register (offset: 0x3D28)

Status Block 7 Host Address Register (offset: 0x3D30)

Status Block 8 Host Address Register (offset: 0x3D38)

Status Block 9 Host Address Register (offset: 0x3D40)

Status Block 10 Host Address Register (offset: 0x3D48)

Status Block 11 Host Address Register (offset: 0x3D50)

Status Block 12 Host Address Register (offset: 0x3D58)

Status Block 13 Host Address Register (offset: 0x3D60)

Status Block 14 Host Address Register (offset: 0x3D68)

Status Block 15 Host Address Register (offset: 0x3D70)

Status Block 16 Host Address Register (offset: 0x3D78)

Status Block Base Address Register (offset: 0x3C44)

This 32-bit register is the location of the status block structure in NIC memory.

Note: This register should always be set to 0xB00. See Appendix C: "Device Register and Memory Map," on page 588 for details.

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.

Name	Bits	Access	Default Value	Description
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 Jumbo 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 104: 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 105: 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 BCM5718 family. This value will be masked by the RCB bit-mask in WDMA module before be DMAed in status block.

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

Name	Bits	Access	Default Value	Description
Received BD jumbo Producer Ring consumer Index	7:0	RW	0	Current Received BD Consumer index.

NIC Standard Receive BD Consumer Index Register (offset: 0x3C54)

This registers is shared by the Receive BD Completion and the Receive Data and Receive BD Initiator state machines which is used to keep track of the receive BDs that have been DMAed to the NIC.

NIC Mini Receive BD Consumer Index (offset: 0x3c58)

This register is shared by the Receive BD Completion and the Receive Data and Receive BD Initiator state machines which is 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	–
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.

Name	Bits	Access	Default Value	Description
TX-MAC Addr Trap Enable	2	RW	0	Transmit MAC Memory Arbiter request trap enable.
Enable	1	RW	1	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	RWC	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.

Name	Bits	Access	Default Value	Description
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	0xA000h	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	0xA000h	Specifies the length of MBUF assigned for receive packet. The default is 32 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.

MAC RX 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.

Read DMA 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 allocation Response register for the TXMBUF pointer.

Name	Bits	Access	Default Value	Description
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	–
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	31:26	RO	0	–
RXMBUF Count	25:17	RO	1	The number of RXMBUFs that were allocated.
Reserved	16	RO	0	–
TXMBUF Count	14:9	RO	1	The number of TXMBUFs that were allocated.
RXMBUF Left	8:0	RO	0x13f	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.

RDMA Registers

All registers reset are core reset unless specified.

LSO Read DMA Mode Register (offset: 0x4800)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:30	RO	0	–
In-Band VLAN Tag Enable	29	RW	1	In Band VLAN Tag Enable 1: Enable In_Band VLAN Tag 0: Disable In-Band VLAN Tag
Hardware IPv6 Post-DMA Processing Enable	28	RW	0	Enables hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets.
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.
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 is recommended that this bit should not be set by firmware or software. 1: Enable Address Overflow Error Logging 0: Disable Address Overflow Error Logging.
Disable Multiple Outstanding Read DMA	24	WO	0	0: Enable Multiple Outstanding Read DMA 1: Disable Multiple Outstanding Read DMA This bit will always read back as 0, even if written as 1. This feature should be disabled in 5718 A0 Chip
Reserved	23:18	RO	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
PCI Request Burst Length for LSO engine	17:16	RW	0	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 ***Note:*** The actual size of the resulting PCIe memory read transaction may be further limited by the PCIe Maximum Read Request Size (MRRS) field of the PCIe Device Status Control Register (0xB4).
Reserved	15:11	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.

LSO Read DMA Status Register (offset: 0x4804)

Name	Bits	Access	Default Value	Description
Reserved	31:11	RO	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 PCI Parity Error	4	W2C	0	Read DMA PCI Parity Error.
Read DMA PCI Master Abort	3	W2C	0	Read DMA PCI Master Abort Error.
Read DMA PCI Target Abort	2	W2C	0	Read DMA PCI Target Abort Error.
Reserved	1:0	RO	0	–

LSO Read DMA Programmable IPv6 Extension Header Register (offset: 0x4808)

Name	Bits	Access	Default Value	Description
Programmable Extension Header Type #2 Enable	31	RW	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	RW	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].

Name	Bits	Access	Default Value	Description
Reserved	29:16	RO	0	–
Programmable Extension Header Type #2	15:8	RW	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	RW	0	These bits contain the programmable extension header value for programmable header #1.

LSO Read DMA Reserved Control Register (offset: 0x4900)

Name	Bits	Access	Default Value	Description
FIX for controller stop passing traffic	31:21	RW	0	Fix the controller stop passing traffic when flow control is enabled. It appears that the chip can get stuck in a permanent XOFF state under heavy bi-directional netperf traffic when flow control is enabled.
Fix in A1 for Rx discard counter update in offset x2250 when multicast/unicast/broadcast packets are dropped	20	RW	0	0: Enable fix 1: Disable fix
FIFO High Mark	19:12	RW	0x90	–
FIFO Low Mark	11:4	RW	0x40	–
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.
Fix for the DMA FIFO overrun	2	RW	0	When set, this bit enables the fix, 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.
Late Collision Fix Enable	1	RW	0	0: Disable Fix 1: 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.

LSO Read DMA Flow Reserved Control Register (offset: 0x4904)

Name	Bits	Access	Default Value	Description
Fix for frequent TX time out.	31:24	RW	0	This register contains various controls to configure hardware fix for the chip getting stuck in a permanent XOFF state under heavy bi-directional netperf traffic when flow control is enabled.
Fifo_threshold_mbuf_req	23:16	RW	0x30	–
MBUF Threshold MBUF Request	15:8	RW	0x54	–
Reserved	7	RW	0	–
Clock Request Fix Enable	6	RW	0	–
MBUF Threshold Clk Req	5:0	RW	0x7	–

LSO/Non-LSO/BD Read DMA Corruption Enable Control Register (offset: 0x4910)

Name	Bits	Access	Default Value	Description
DTM stress failure 8bytes duplication fix disable in Non-LSO engine	31	RW	0	0: Enable fix 1: Disable fix
Reserved	30:28	RW	0	–
Disable fix in A1 for B2B Pcie completion good pkt followed by bad pkt (that is, packet with LCRC error)	27	RW	0	0: Enable fix 1: Disable fix
Disable fix in A1 for B2B Pcie completion good pkt followed by good pkt	26	RW	0	0: Enable fix 1: Disable fix
Disable fix in A1 for DMA Read engine miscalculation of TXMBUF Available Space	25	RW	0	0: Enable fix 1: Disable fix
Disable fix in A1 for DMA Read Underrun when running Core Clk same speed as TL Clock when CLKREQ is enabled	24	RW	0	0: Enable fix 1: Disable fix
Disable fix in A1 for RDMA FIFO Overrun	23	RW	0	0: Enable fix 1: Disable fix
DTM stress failure DMA too large error fix disable for Non-LSO engine	22	RW	0	0: Enable fix 1: Disable fix

Name	*Bits*	*Access*	*Default Value*	*Description*
DTM stress test 8bytes missing failure fix disable for Non-LSO engine	21	RW	0	0: Enable fix 1: Disable fix
DTM stress failure corruption fix disable for Non-LSO engine	20	RW	0	0: Enable fix 1: Disable fix
PCI Request Burst Length for Non-LSO RDMA engine	19:18	RW	0	The two bits define the burst length that the Non-LSO RDMA read engine would request to the PCI block. • 00 = 128B • 01 = 256B • 10 = 512B • 11 = 4KB
PCI Request Burst Length for BD RDMA engine	17:16	RW	0	The two bits define the burst length that the BD RDMA read engine would request to the PCI block. • 00 = 128B • 01 = 256B • 10 = 512B • 11 = 4KB
cq46758_fix_enable	15	RW	0	0 0: Enable fix 1: Disable fix
sbd_8b_less_fix_enable3	14	RW	0	0: Enable fix 1: Disable fix
sbd_8b_less_fix_enable2	13	RW	0	0: Enable fix 1: Disable fix
mem_too_large_fix_enable2	12	RW	0	0: Enable fix 1: Disable fix
mem_too_large_fix_enable1	11	RW	0	0: Enable fix 1: Disable fix
mem_too_large_fix_enable	10	RW	0	0: Enable fix 1: Disable fix
sbd_8b_less_fix_enable_fast_return	9	RW	0	0: Enable fix 1: Disable fix
sbd_8b_less_fix_enable	8	RW	0	0: Enable fix 1: Disable fix
Enable hardware fix for the wrong TCP checksum when LSO send out	7	RW	0	The fix involves wiping out the tcp_checksum calculation. 0: Enable Fix 1: Disable Fix
Reserved	6	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Fix for IP header checksum error in high transfer rate	5	RW	0	This control bit is used to enable the fix for the IP Header Checksum Corruption occurs when an IPV4 payload contents match an IPV6 Header Type. 1: Disable fix 0: enable fix
Enable hardware fix for TX Read DMA lock-up LOCKUP	4	RW	1	Set to 1 to enable fix for clock request gap problem of Tx Read DMA lock-up issue. Note that 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	3	RW	1	Set to 1 to enable fix for clock switching problem of Tx Read DMA lock-up issue. Note that 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	2	RW	1	Set to 1 to enable fix for Tx Read DMA lock-up issue. Note that 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	0	RW	0	–

BD Read DMA Mode Register (offset: 0x4A00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
Hardware IPv6 Post-DMA Processing Enable	28	RO	0	Enables hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets.
Hardware IPv4 Post-DMA Processing Enable	27	RO	0	Enables hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets.
Post-DMA Debug Enable	26	RO	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	RO	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 is 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	RO	0	• 00 = 128 bytes • 01 = 256 bytes • 10 = 512 bytes • 11 = 4096 bytes ***Note:*** The actual size of the resulting PCIe memory read transaction may be further limited by the PCIe Maximum Read Request Size (MRRS) field of the PCIe Device Status Control Register (0xB4).
Reserved	15:11	RO	0	–
Read DMA PCI-X Split Transaction Timeout Expired Attention Enable	10	RO	0	Enable read DMA PCI-X split transaction timeout expired attention.
Read DMA Local Memory Write Longer Than DMA Length Attention Enable	9	RO	0	Enable Read DMA Local Memory Write Longer Than DMA Length Attention.
Read DMA PCI FIFO Overread Attention Enable	8	RO	0	Enable Read DMA PCI FIFO Overread Attention (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Attention Enable	7	RO	0	Enable Read DMA PCI FIFO Underrun Attention.

Name	Bits	Access	Default Value	Description
Read DMA PCI FIFO Overrun Attention Enable	6	RO	0	Enable Read DMA PCI FIFO Overrun Attention.
Read DMA PCI Host Address Overflow Error Attention Enable	5	RO	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	RO	0	Enable Read DMA PCI Parity Error Attention.
Read DMA PCI Master Abort Attention Enable	3	RO	0	Enable Read DMA PCI Master Abort Attention.
Read DMA PCI Target Abort Attention Enable	2	RO	0	Enable Read DMA PCI Target Abort Attention.
Enable	1	RO	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	RO	0	When this bit is set to 1, the Read DMA state machine is reset. This is a self-clearing bit.

BD READ DMA Status Register (offset: 0x4A04)

Name	Bits	Access	Default Value	Description
rdmad_length_b_0	31:16	RO	0	–
Reserved	15:11	RO	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.

Name	Bits	Access	Default Value	Description
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 PCI Parity Error	4	W2C	0	Read DMA PCI Parity Error.
Read DMA PCI Master Abort	3	W2C	0	Read DMA PCI Master Abort Error.
Read DMA PCI Target Abort	2	W2C	0	Read DMA PCI Target Abort Error.
Reserved	1:0	RO	0	–

BD READ DMA Reserved Control Register (offset: 0x4A70)

Name	Bits	Access	Default Value	Description
Txmbuf_margin	31:21	RO	0	–
Reserved	20	RO	0	–
FIFO High Mark	19:12	RO	0x90	–
FIFO Low Mark	11:4	RO	0x40	–
Slow Clock Fix Disable	3	RO	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.
Stop communicating after heavy stress Fix	2	RO	0	When set, this bit enables the fix for 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.
Late Collision Fix Enable	1	RO	0	0: Disable Fix 1: Enable Fix
Select FED Enable	0	RO	0	Ensure only 1 request is generated upon any condition where the core clock is switching from slow to fast or vice-versa.

BD READ DMA Flow Reserved Control Register (offset: 0x4A74)

Name	Bits	Access	Default Value	Description
Fifo_threshold_bd_req	31:24	RO	0	–
Fifo_threshold_mbuf_req	23:16	RO	0x30	–

Name	Bits	Access	Default Value	Description
MBUF Threshold MBUF Request	15:8	RO	0x54	–
Reserved	7	RO	0	–
Clock Request Fix Enable	6	RO	0	–
MBUF Threshold Clk Req	5:0	RO	0x7	–

BD READ DMA Corruption Enable Control Register (offset: 0x4A78)

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0	–
Enable hardware fix for wrong TCP checksum	7	RO	0	0: Enable Fix 1: Disable Fix
Reserved	6	RO	0	–
Enable hardware fix for IP Header checksum corruption	5	RO	0	This control bit is used to enable the fix for the IP Header Checksum Corruption occurs when an IPV4 payload contents match an IPV6 Header Type. 1: Disable fix 0: enable fix
DMA Read Engine may be unable to generate Memory Read Request to PCIE Core upon transitioning from L1 to L0 with Clkreq Enable	4	RO	1	Set to 1 to enable fix for clock request gap problem of Tx Read DMA lock-up issue. Note that 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 for clock switching problem	3	RO	1	Set to 1 to enable fix for clock switching problem of Tx Read DMA lock-up issue. Note that 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 for PCIE client lockup	2	RO	1	Set to 1 to enable fix for Tx Read DMA lock-up issue. Note that 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	RO	0	–
Reserved	0	RO	0	–

Non_LSO Read DMA Mode Register (offset: 0x4B00)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:29	RO	0	–
Hardware IPv6 Post-DMA Processing Enable	28	RO	0	Enables hardware processing of LSO IPv6 packets. This bit has no effect on Post-DMA processing of IPv4 packets.
Hardware IPv4 Post-DMA Processing Enable	27	RO	0	Enables hardware processing of LSO IPv4 packets. This bit has no effect on Post-DMA processing of IPv6 packets.
Post-DMA Debug Enable	26	RO	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	RO	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 is 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	RO	0	00 = 128 bytes 01 = 256 bytes 10 = 512 bytes 11 = 4096 bytes ***Note:*** The actual size of the resulting PCIe memory read transaction may be further limited by the PCIe Maximum Read Request Size (MRRS) field of the PCIe Device Status Control Register (0xB4).
Reserved	15:11	RO	0	–
Read DMA PCI-X Split Transaction Timeout Expired Attention Enable	10	RO	0	Enable read DMA PCI-X split transaction timeout expired attention.
Read DMA Local Memory Write Longer Than DMA Length Attention Enable	9	RO	0	Enable Read DMA Local Memory Write Longer Than DMA Length Attention.
Read DMA PCI FIFO Overread Attention Enable	8	RO	0	Enable Read DMA PCI FIFO Overread Attention (PCI read longer than DMA length.)
Read DMA PCI FIFO Underrun Attention Enable	7	RO	0	Enable Read DMA PCI FIFO Underrun Attention.

Name	Bits	Access	Default Value	Description
Read DMA PCI FIFO Overrun Attention Enable	6	RO	0	Enable Read DMA PCI FIFO Overrun Attention.
Read DMA PCI Host Address Overflow Error Attention Enable	5	RO	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	RO	0	Enable Read DMA PCI Parity Error Attention.
Read DMA PCI Master Abort Attention Enable	3	RO	0	Enable Read DMA PCI Master Abort Attention.
Read DMA PCI Target Abort Attention Enable	2	RO	0	Enable Read DMA PCI Target Abort Attention.
Enable	1	RO	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	RO	0	When this bit is set to 1, the Read DMA state machine is reset. This is a self-clearing bit.

Non-LSO Read DMA Status Register (offset: 0x4B04)

Name	Bits	Access	Default Value	Description
rdmad_length_d_0	31:16	RO	0	–
Reserved	15:11	RO	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.

Name	Bits	Access	Default Value	Description
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 PCI Parity Error	4	W2C	0	Read DMA PCI Parity Error.
Read DMA PCI Master Abort	3	W2C	0	Read DMA PCI Master Abort Error.
Read DMA PCI Target Abort	2	W2C	0	Read DMA PCI Target Abort Error.
Reserved	1:0	RO	3	–

Non-LSO Read DMA Programmable IPv6 Extension Header Register (offset: 0x4B08)

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	29:28	RO	0	–
ch0_rlctr	27:19	RO	0	–
Reserved	18:16	RO	0	–
Programmable Extension Header Type #2	15:8	RO	0	These bits contain the programmable extension header value for programmable header #2.
Programmable Extension Header Type #1	7:0	RO	0	These bits contain the programmable extension header value for programmable header #1.

Host Address for the DMA Read Channel 0 (Offset: 0x4B28)

Name	Bits	Access	Default Value	Description
Host Addr	31:0	RO	0	Latched host address

Host Address for the DMA Read Channel 1 (offset: 0x4B30)

Name	Bits	Access	Default Value	Description
Host Addr	31:0	RO	0	Latched host address

Host Address for the DMA Read Channel 2 (offset: 0x4B38)

Name	Bits	Access	Default Value	Description
Host Addr	31:0	RO	0	Latched host address

Host Address for the DMA Read Channel 3 (offset: 0x4B40)

Name	Bits	Access	Default Value	Description
Host Addr	31:0	RO	0	Latched host address

Non-LSO Read DMA Reserved Control Register (offset: 0x4B74)

Name	Bits	Access	Default Value	Description
Txmbuf_margin	31:21	RO	0	–
Reserved	20	RO	0	–
FIFO High Mark	19:12	RO	0x90	–
FIFO Low Mark	11:4	RO	0x40	–
Slow Clock Fix Disable	3	RO	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.
Hardware fix enable for DMA FIFO Overrun	2	RO	0	When set, this bit enables the fix for 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.
Late Collision Fix Enable	1	RO	0	0: Disable Fix 1: Enable Fix
Select FED Enable	0	RO	0	Ensure only 1 request is generated upon any condition where the core clock is switching from slow to fast or vice-versa.

Non-LSO Read DMA Flow Reserved Control Register (offset: 0x4B78)

Name	Bits	Access	Default Value	Description
Fix for frequent TX time out.	31:24	RW	0	This register contains various controls to configure hardware fix for the chip getting stuck in a permanent XOFF state under heavy bi-directional netperf traffic when flow control is enabled.
Fifo_threshold_mbuf_req	23:16	RO	0x30	–
MBUF Threshold MBUF Request	15:8	RO	0x54	–
Reserved	7	RO	0	–
Clock Request Fix Enable	6	RO	0	–
MBUF Threshold Clk Req	5:0	RO	0x7	–

Non-LSO Read DMA Corruption Enable Control Register (offset: 0x4B7C)

Name	Bits	Access	Default Value	Description
Reserved	31:8	RO	0	–
Enable hardware fix for wrong TCP checksum	7	RO	0	0: Enable Fix 1: Disable Fix
Reserved	6	RO	0	–
Enable hardware fix for IP Header checksum corruption	5	RO	0	This control bit is used to enable the fix for the IP Header Checksum Corruption occurs when an IPV4 payload contents match an IPV6 Header Type. 1: Disable fix 0: enable fix
DMA Read Engine may be unable to generate Memory Read Request to PCIE Core upon transitioning from L1 to L0 with Clkreq Enable	4	RO	1	Set to 1 to enable fix for clock request gap problem of Tx Read DMA lock-up issue. Note that 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 for clock switching problem	3	RO	1	Set to 1 to enable fix for clock switching problem of Tx Read DMA lock-up issue. Note that 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.

Name	*Bits*	*Access*	*Default Value*	*Description*
Enable hardware fix for PCIE client lockup	2	RO	1	Set to 1 to enable fix for Tx Read DMA lock-up issue. Note that 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	RO	0	–
Reserved	0	RO	0	–

Write DMA 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	–
Enable FIX for host coalescing bug	29	RW	0	1: Enable the FIX 0: Disable the FIX To Fix the bug that device will send out Status Block before the interrupt message.
(BCM5719/5720 only) WDMA small packet improvement disable	28	RW	0	wdma_small_packet_improvement_disable 0: Use new local RRR data instead of fetching from RBD memory (default enable). 1: Use old implementation to fetch RRR data from RBD memory.
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.
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.

Name	Bits	Access	Default Value	Description
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.
Write DMA PCI Parity Error	4	W2C	0	Write DMA PCI Parity Error.
Write DMA PCI Master Abort Error	3	W2C	0	Write DMA PCI Master Abort Error.
Write DMA PCI Target Abort Error	2	W2C	0	Write DMA PCI Target Abort Error.
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.
Halt RX RISC	10	RW	0	Set by TX RISC or the host to halt the RX RISC.Cleared on reset and Watchdog interrupt.
Flush Instruction Cache	9	WO	0	Self-clearing bit which forces the instruction cache to flush.
Enable Instruction Cache prefetch	8	RW	0	Enables prefetch logic within the instruction cache. When disabled only a single cache line is read on a cache miss. Cleared on reset.
Enable Watchdog	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.
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.
Enable Data Cache	5	RW	0	Enables the data cache. Cleared on reset. Note: Firmware developers should take care to clear this bit before polling internal SRAM memory locations, because the RX RISC processor uses a two-element LRU caching algorithm, which is not affected by writes from the PCI interface.

Name	Bits	Access	Default Value	Description
Enable Write Post Buffers	4	RW	0	Enables absorption of multiple software operations for SRAM and register writes. When this bit is disabled, only one write at a time will be absorbed by the write post buffers. Cleared on reset. ***Note:*** Setting this bit on the BCM5705, BCM5721, and BCM5751 may cause unpredictable behavior.
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 non-disable-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.
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	–

Name	*Bits*	*Access*	*Default Value*	*Description*
Instruction fetch stall	15	RO	0	The processor is currently stalled due to an instruction fetch.
Data access stall	14	RO	0	The processor is currently stalled due to a data access.
Reserved	13:11	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.
Hardware Breakpoint	0	W2C	0	When enabled in mode register, indicates hardware breakpoint has been reached.

RX RISC Program Counter (offset: 0x501C)

The program counter register can be used to read or write the current Program Counter of the each CPU. Reads can occur at any time, however writes can only be performed when the CPU is halted. Writes will also clear any pending instruction in the decode stage of the pipeline. Bits 31-2 are implemented. 1s written to bits 1-0 are ignored.

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.

VRQ Statistics

Apart from the Generic VRQ statistics, EMAC shall also offer statistics on the RX Default VRQ and RX Drop VRQ.

Table 106: Generic VRQ Statistics (Offset 0x0BFF - 0x0A00)

VRQ#	*Statistics Name*	*Address*	*Size*	*Description*
1	IFHCOUTOCTETS	0x0A00	32	Queue#1 Transmit
	IFHCOUTUCAST	0x0A04	32	Queue#1 Transmit
	IFHCOUTMCAST	0x0A08	32	Queue#1 Transmit
	IFHCOUTBCAST	0x0A0C	32	Queue#1 Transmit
	IFHCINOCTETS_GOOD	0x0A10	32	VRQ#1 Receive
	Reserved	0x0A14	32	VRQ#1 Receive
	IFHCINUCASTPKTS	0x0A18	32	VRQ#1 Receive
	IFHCINMULTICASTPKTS	0x0A1C	32	VRQ#1 Receive
2	IFHCOUTOCTETS	0x0A20	32	Queue#2 Transmit
	IFHCOUTUCAST	0x0A24	32	Queue#2 Transmit
	IFHCOUTMCAST	0x0A28	32	Queue#2 Transmit
	IFHCOUTBCAST	0x0A2C	32	Queue#2 Transmit
	IFHCINOCTETS_GOOD	0x0A30	32	VRQ#2 Receive
	Reserved	0x0A34	32	VRQ#2 Receive
	IFHCINUCASTPKTS	0x0A38	32	VRQ#2 Receive
	IFHCINMULTICASTPKTS	0x0A3C	32	VRQ#2 Receive
….	…….	……..	–	–

Table 106: Generic VRQ Statistics (Offset 0x0BFF - 0x0A00) (Cont.)

VRQ#	Statistics Name	Address	Size	Description
….	…….	……...	–	–
16	IFHCOUTOCTETS	0x0BE0	32	Queue#16 Transmit
	IFHCOUTUCAST	0x0BE4	32	Queue#16 Transmit
	IFHCOUTMCAST	0x0BE8	32	Queue#16 Transmit
	IFHCOUTBCAST	0x0BEC	32	Queue#16 Transmit
	IFHCINOCTETS_GOOD	0x0BF0	32	VRQ#16 Receive
	Reserved	0x0BF4	32	VRQ#16 Receive
	IFHCINUCASTPKTS	0x0BF8	32	VRQ#16 Receive
	IFHCINMULTICASTPKTS	0x0BFC	32	VRQ#16 Receive

Table 107: Default & Drop VRQ Statistics (Offset 0x09F7 - 0x09D0)

VRQ#	Statistics Name	Address	Size	Description
0 (Default)	IFHCINOCTETS_GOOD	0x09D0	32	Default VRQ Stats (Receive)
	IFHCINOCTETS_BAD	0x09D4	32	
	IFHCINUCASTPKTS	0x09D8	32	
	IFHCINMULTICASTPKTS	0x09DC	32	
	IFHCINBROADCASTPKTS	0x09E0	32	
17 (Drop)	IFHCINOCTETS_GOOD	0x09E4	32	Drop VRQ Stats (Receive)
	Reserved	0x09E8	32	
	IFHCINUCASTPKTS	0x09EC	32	
	IFHCINMULTICASTPKTS	0x09F0	32	
	IFHCINBROADCASTPKTS	0x09F4	32	

VRQ Filter Set Registers

The filter block may be programmed via the following set of registers:

- VRQ_FILT_ELEM_CFG_REG [0 through 31]: (Offset 0x5400 - 0x547C)
- VRQ_FILT_ELEM_PTTRN_REG [0 through 31]: (Offset 0x5480 - 0x54FC)
- VRQ_FILT_SET_MASK_REG [1 through 31]: (Offset 0x5504 - 0x557C)

Note: There is no equivalent of FILT_SET_CFG_REG(s) in VRQ-Filter Block. Also, Set#0 is non-existent.

VRQ Mapper Registers

Each VRQ Mapper Entry is represented by a pair of 32-bit registers, named 1st Half of Entry and 2nd Half of Entry. There shall be 18 such register pairs in order to map to 18 VRQ entries - including the Default VRQ and the Drop VRQ. This register pair is used to program an SOP equation expressed as:

Filter Result = (Perf A AND/OR Set A) || (Perf B AND/OR Set B) || (Perf C AND/OR Set C) || (Perf D AND/OR Set D).

Table 108: First Half VRQ Mapper Entry Register

Name	*Bits*	*Access*	*Default Value*	*Description*
Product Term D				
Reserved	31:30	RO	0	–
Activate Perfect Match in this Term	29	RW	0	Write a 1 in this field to activate the Perfect Match Address Reg selected by field [28:24].
Perfect Match Address Reg #	28:24	RW	0x0	Write 0 – 23 in this field to select the Perfect Match Address Reg Term D.
Product Term C				
Reserved	23:22	RO	0	–
Activate Perfect Match in this Term	21	RW	0	Write a 1 in this field to activate the Perfect Match Address Reg selected by field [20:16]
Perfect Match Address Reg #	20:16	RW	0x0	Write 0 – 23 in this field to select the Perfect Match Address Reg for Term C.
Product Term B				
Reserved	15:14	RO	0	–
Activate Perfect Match in this Term	13	RW	0	Write a 1 in this field to activate the Perfect Match Address Reg selected by field [12:8].
Perfect Match Address Reg #	12:8	RW	0x0	Write 0 – 23 in this field to select the Perfect Match Address Reg for Term B.
Product Term A				
Reserved	7:6	RO	0	–
Activate Perfect Match in this Term	5	RW	0	Write a 1 in this field to activate the Perfect Match Address Reg selected by field [4:0].
Perfect Match Address Reg #	4:0	RW	0x0	Write 0 – 23 in this field to select the Perfect Match Address Reg for Term A.

Table 109: Second Half VRQ Mapper Entry Register

Name	*Bits*	*Access*	*Default Value*	*Description*
Product Term D				

Table 109: Second Half VRQ Mapper Entry Register (Cont.)

Name	*Bits*	*Access*	*Default Value*	*Description*
Activate Product Term D	31	RW	0	Write a 1 to activate this whole Product Term.
AND/OR	30	RW	0	0 => Perfect Match Addr && VRQ Filter Set 1 => Perfect Match Addr || VRQ Filter Set
Activate VRQ Filter Set in this Term	29	RW	0	Write a 1 in this field to activate the VRQ Filter Set selected by field [28:24].
VRQ Filter Set #	28:24	RW	0x0	Write 1 – 31 in this field to select the VRQ Filter Set for Term D.
Product Term C				
Activate Product Term C	23	RW	0	Write a 1 to activate this whole Product Term
AND/OR	22	RW	0	0 => Perfect Match Addr && VRQ Filter Set 1 => Perfect Match Addr || VRQ Filter Set
Activate VRQ Filter Set in this Term	21	RW	0	Write a 1 in this field to activate the VRQ Filter Set selected by field [20:16].
VRQ Filter Set #	20:16	RW	0x0	Write 1 – 31 in this field to select the VRQ Filter Set for Term C.
Product Term B				
Activate Product Term B	15	RW	0	Write a 1 to activate this whole Product Term
AND/OR	14	RW	0	0 => Perfect Match Addr && VRQ Filter Set 1 => Perfect Match Addr || VRQ Filter Set
Activate VRQ Filter Set in this Term	13	RW	0	Write a 1 in this field to activate the VRQ Filter Set selected by field [12:8].
VRQ Filter Set #	12:8	RW	0x0	Write 1 – 31 in this field to select the VRQ Filter Set for Term B.
Product Term A				
Activate Product Term A	7	RW	0	Write a 1 to activate this whole Product Term.
AND/OR	6	RW	0	0 => Perfect Match Addr && VRQ Filter Set 1 => Perfect Match Addr || VRQ Filter Set
Activate VRQ Filter Set in this Term	5	RW	0	Write a 1 in this field to activate the VRQ Set selected by field [4:0].
VRQ Filter Set #	4:0	RW	0x0	Write 1 – 31 in this field to select the VRQ Filter Set for Term A.

The table below lists registers addresses of all 18 pairs:

Table 110: VRQ Mapper Register List

VRQ Entry #	Second Half Register Address	First Half Register Address
0 (Default)	0x5604	0x5600
1	0x560C	0x5608
2	0x5614	0x5610

Table 110: VRQ Mapper Register List (Cont.)

VRQ Entry #	Second Half Register Address	First Half Register Address
3	0x561C	0x5618
4	0x5624	0x5620
5	0x562C	0x5628
6	0x5634	0x5630
7	0x563C	0x5638
8	0x5644	0x5640
9	0x564C	0x5648
10	0x5654	0x5650
11	0x565C	0x5658
12	0x5664	0x5660
13	0x566C	0x5668
14	0x5674	0x5670
15	0x567C	0x5678
16	0x5684	0x5680
17 (Drop)	0x568C	0x5688

VRQ Enable Register (Offset 0x560)

This register contains an 18-bit field to provide the ability to enable or disable desired number of VRQs.

Table 111: VRQ Enable Register (Offset 0x560)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:18	RO	000	Reserved
VRQ Enable Bit-Map	17:0	RW	0x0000	When a bit position is written 1 the respective VRQ# is enabled and is capable of receiving packets. When a bit position is written 0, the respective VRQ# is not capable of receiving packets – even though its VRQ Mapper Entry is programmed. Bit[0] => Default VRQ Bit[16:1] => Generic VRQs Bit[17] => Drop Queue

RX Mail Box Registers for VRQ

A set of new High Priority Mail Box Registers are introduced. The register addresses are as shown in the table below:

Table 112: High Priority Mail Box Registers for VRQ Rings

VRQ #	*RX Jumbo Ring Producer Index*	*RX Standard Ring Producer Index*	*RX Return Ring Consumer Index*		*Comments*	*NIC Diagnostic RX Return Ring Producer Index*
0 (Default)	0x270	UNDI - 0x98 HP - 0x268	UNDI – 0x88 HP - 0x280		64-bit Registers	0x3C80
1	0x2D4	0x354	0x288	64-bit Regs	The new Registers are 32-bit	0x3C84
2	0x2D8	0x358	0x290			0x3C88
3	0x2DC	0x35C	0x298			0x3C8C
4	0x2E0	0x360	0x2A0		The new Registers are 32-bit	0x3C90
5	0x2E4	0x364	0x2A4			0x3C94
6	0x2E8	0x368	0x2A8			0x3C98
7	0x2EC	0x36C	0x2AC			0x3C9C
8	0x2F0	0x370	0x2B0			0x3CA0
9	0x2F4	0x374	0x2B4			0x3CA4
10	0x2F8	0x378	0x2B8			0x3CA8
11	0x2FC	0x37C	0x2BC			0x3CAC
12	0x340	0x380	0x2C0			0x3CB0
13	0x344	0x384	0x2C4			0x3CB4
14	0x348	0x388	0x2C8			0x3CB8
15	0x34C	0x38C	0x2CC			0x3CBC
16	0x350	0x390	0x2D0			0x3CC4

Note: 0x268 - 0x298, 0x88, 0x98 are legacy registers.

NIC Diagnostic Standard RBD Consumer Index [1 - 16] == 0x3F40 - 0x3F7C

NIC Diagnostic Jumbo RBD Consumer Index [1 - 16] == 0x3F80 - 0x3FFC.

Perfect Match Destination Address Registers

Twenty more Perfect Match Destination Address Registers are added in RX-EMAC for VRQ Filtering purposes - Register [0 -3] are legacy, whilst [4 -23] are shown here:

VRQ_PERFECT_MATCH[4 - 23]_HIGH_REG (Offsets: 0x5690, 0x5698, 0x56A0 … 0x5728)

There are total 24 Perfect (Destination Address) Match registers for VRQ Filtering in RX-MAC. These registers hold the higher 2 octets of the matching address.

Table 113: VRQ_PERFECT_MATCH[4 - 23]_HIGH_REG (Offsets: 0x5690, 0x5698, 0x56A0 … 0x5728)

Name	Bits	Access	Default Value	Description
Reserved	31:29	RO	000	Reserved
MAC High Address	15:0	RW	0x0000	Upper 2-bytes of MAC address

VRQ_PERFECT_MATCH[4 - 23]_LOW_REG (Offsets: 0x5694, 0x569C, 0x56A4 … 0x572C)

There are total 24 Perfect (Destination Address) Match registers for VRQ Filtering in RX-MAC. These registers hold the higher 2 octets of the matching address.

Table 114: VRQ_PERFECT_MATCH[4 - 23]_LOW_REG (Offsets: 0x5694, 0x569C, 0x56A4 … 0x572C)

Name	Bits	Access	Default Value	Description
MAC Low Address	31:0	RO	0x0000	Lower 4-bytes of MAC 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. 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.

Interrupt Mailbox 0 Register (offset: 0x5800)

This mailbox serves two functions. When the host writes it, the interrupt (IntA) is cleared. It is also used by the Host Coalescing engine to determine if the host is in the interrupt handler. If it is non-zero this indicates the host is in the interrupt handler. If it is zero this indicates the host is not in the interrupt handler. The Host Coalescing engine uses this information to determine which set of coalescing parameters it should use.

Other Interrupt Mailbox Register (offset: 0x5808–0x5818)

This mailbox serves two functions. When the host writes it, the interrupt (IntA) is cleared. It is also used by the Host Coalescing engine to determine if the host is in the interrupt handler. If it is non-zero this indicates the host is in the interrupt handler. If it is zero this indicates the host is not in the interrupt handler. The Host Coalescing engine uses this information to determine which set of coalescing parameters it should use.

General Mailbox Registers 1-8 (offset: 0x5820–0x5824)

This mailbox serves two functions. When the host writes it, the interrupt (IntA) is cleared. It is also used by the Host Coalescing engine to determine if the host is in the interrupt handler. If it is non-zero this indicates the host is in the interrupt handler. If it is zero this indicates the host is not in the interrupt handler. The Host Coalescing engine uses this information to determine which set of coalescing parameters it should use.

Receive BD Standard Producer Ring Index Register (offset: 0x5868)

The Receive BD Standard Producer Ring Index register, contain 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 Jumbo Producer Ring Index Register (offset: 0x5870-5877)

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 NIC to DMA into NIC memory. Host software writes this register whenever it updates the extended producer ring. This register must be initialized to 0.

Name	*Bits*	*Access*	*Default Value*	*Description*
Received BD jumbo Producer Ring Index	7:0	RW	–	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 BD Return Ring 0 Consumer Index Register (offset: 0x5880-0x5887)

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 0. This register must be initialized to 0.

Receive BD Return Ring 1 Consumer Index 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 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 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 Host Producer Index Register (offset: 0x5900)

The Send BD Ring Host Producer Index register, contain 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. The host producer indices may not be used at the same time as the NIC producer indices.

When multiple TXQ is enabled:

Send BD Ring Host Producer Index Register for Ring 1 (offset: 0x5900)

Send BD Ring Host Producer Index Register for Ring 2 (offset: 0x5908)

Send BD Ring Host Producer Index Register for Ring 3 (offset: 0x590C)

Send BD Ring Host Producer Index Register for Ring 4 (offset: 0x5910)

Send BD Ring Host Producer Index Register for Ring 5 (offset: 0x5914)

Send BD Ring Host Producer Index Register for Ring 6 (offset: 0x5918)

Send BD Ring Host Producer Index Register for Ring 7 (offset: 0x591C)

Send BD Ring Host Producer Index Register for Ring 8 (offset: 0x5920)

Send BD Ring Host Producer Index Register for Ring 9 (offset: 0x5924)

Send BD Ring Host Producer Index Register for Ring 10 (offset: 0x5928)

Send BD Ring Host Producer Index Register for Ring 11 (offset: 0x592C)

Send BD Ring Host Producer Index Register for Ring 12 (offset: 0x5930)

Send BD Ring Host Producer Index Register for Ring 13 (offset: 0x5934)

Send BD Ring Host Producer Index Register for Ring 14 (offset: 0x5938)

Send BD Ring Host Producer Index Register for Ring 15 (offset: 0x593C)

Send BD Ring Host Producer Index Register for Ring 16 (offset: 0x5940)

Send BD Ring NIC Producer Index Register (offset: 0x5980)

The Send BD Ring NIC Producer Index register contains the index of the next buffer descriptor for a given send ring that will be produced in the host directly into NIC memory. The host software writes this register whenever it updates the given send ring. This register must be initialized to 0. The host producer indices may not be used at the same time as the NIC producer indices.

When multiple TXQ is enabled:

Send BD Ring Producer Index Register for Ring 1 (offset: 0x5980)

Send BD Ring Producer Index Register for Ring 2 (offset: 0x5984)

Send BD Ring Producer Index Register for Ring 3 (offset: 0x5988)

Send BD Ring Producer Index Register for Ring 4 (offset: 0x598C)

Send BD Ring Producer Index Register for Ring 5 (offset: 0x5990)

Send BD Ring Producer Index Register for Ring 6 (offset: 0x5994)

Send BD Ring Producer Index Register for Ring 7 (offset: 0x5998)

Send BD Ring Producer Index Register for Ring 8 (offset: 0x599C)

Send BD Ring Producer Index Register for Ring 9 (offset: 0x59A0)

Send BD Ring Producer Index Register for Ring 10 (offset: 0x59A4)

Send BD Ring Producer Index Register for Ring 11 (offset: 0x59A8)

Send BD Ring Producer Index Register for Ring 12 (offset: 0x59AC)

Send BD Ring Producer Index Register for Ring 13 (offset: 0x59B0)

Send BD Ring Producer Index Register for Ring 14 (offset: 0x59B4)

Send BD Ring Producer Index Register for Ring 15 (offset: 0x59B8)

Send BD Ring Producer Index Register for Ring 16 (offset: 0x59BC)

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	–

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.

The TXMBUF cluster for the ASF message is defaulted to the uppermost three TXMBUFs.

Name	Bits	Access	Default Value	Description
Reserved	31	WO	0	Must Write 1 to transmit a packet
Reserved	30:16	WO	0	Must Write 0
Head TXMBUF Pointer	15:8	WO	0	Specifies the first MBUF of the TXMBUF cluster for the transmit packet.
Tail TXMBUF Pointer	7:0	WO	3Fh	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 is set, 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
(BCM5718) Reserved	29:11	RO	0	–
(BCM5719) Reserved	29:24	RO	0	–
(BCM5719) MSI ST Lower	23:16	RW	0x01	This 8-bit value is used as the TPH Steering Tag when both TPH Interrupt Vector mode and MSI mode are enabled.
(BCM5719) Reserved	15;11	RO	0	–

Name	Bits	Access	Default Value	Description
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.
(BCM5718) Reserved	7	RO	0	–
(BCM5719) MSIX Multimode Vector Enable	7	RW	0	–
MSI Byte Swap	6		0	–
MSI Single Shot 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 Tag Interrupt mode. The Tag Interrupt mode is enabled by setting bit 9 of Register 0x68. When One-Shot MSI mode is enabled, the hardware automatically auto-mask and auto-acknowledge when MSI Interrupt is generated by the hardware. Software needs to un-mask the interrupt after it has serviced the current MSI Interrupt to allow future interrupt to generate by the hardware. The mechanism for software to un-mask 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. See note below.
PCI Parity Error Attn	4	RW	0	PCI parity error attention enable.
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

Name	*Bits*	*Access*	*Default Value*	*Description*
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.

DMA Completion Registers

All registers reset are core reset unless specified.

DMA Completion Mode Register (offset: 0x6400)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:2	RW	–	Reserved
Enable	1	RW	0	This bit controls whether the DMA 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 bit is set to '1' the DMA Completion state machine is reset. This is a self-clearing bit.

GRC Registers

All registers reset are core reset unless specified.

Mode Control Register (offset: 0x6800)

Name	*Bits*	*Access*	*Default Value*	*Description*
Pcie TL/DL/PL mapping bit	31	RW	0	Bit[31][22][29] remap PCIE core TL/DL/PL register to GRC space from 0x6400 to 0x67ff. [31]: • 0: Select lower 1 KB of each TL/DL/PL • 1 Select higher 1 KB of each TL/DL/Pl [22][29]: • 00: select TL register • 01: select DL register • 10: select PL register
Multicast enable	30	RW	0	Multicast enable bit.
Pcie TL/DL/PL mapping bit	29	RW	–	–
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.

Name	Bits	Access	Default Value	Description
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.
TXCP_ATTN_INT_EN	24	RO	0	TXCP attention bit.
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.
Pcie TL/DL/PL mapping bit	22	RO	0	–
NVRAM Write Enable	21	RW	0	The host must set this bit before attempting to update the Flash or SEEPROM ***Note:*** This bit is only valid when set from GRC port 0. Setting/clearing of this bit from GRC port 1 does not affect NVRAM write access control. For BCM5718 family, it is recommended to use 0x7024[1] from all ports to control NVRAM write access.
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.
Time Sync Mode Enable	19	RW	0	Write 1 to this bit to enable Time Sync Mode.
HTX2B Feature Enable (BCM5720)	18	RW	0	Write 1 to enable TX-MAC to route HTX2B packets to the APE. Firmware may enable/disable this in flight with careful coordination.
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.
B2HRX Feature Enable	15	RW	0	Write 1 to enable DMA Engine to route B2HRX packets to PCIe. Firmware may enable/disable this in flight with careful coordination.
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.
DW_SYS_ATTN	12	RO	0	WDMA attention bit.
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	–

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.
IOV Mode Enable	8	RW	0	When this bit is written 1, the chip enters IOV-MODE. When this bit is written 0, the chip operates in legacy mode. This bit must only be configured during boot-up and must not be changed afterwards.
Legacy	7:0			Defined by Legacy
Byte Swap for B2HRX Data	7	RW	0	This bit must be 1 for proper B2HRX operation in case of Little-Endian Host machines.
Word Swap for B2HRX Data	6	RW	0	This bit must be 1 for proper B2HRX operation in case of Little-Endian Host machines.
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.
FLR state	3	RO	0	When this bit is set, it means PCIE is in FLR state for this MAC core.
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.
Reserved	0	RO	0	–

Miscellaneous Configuration Register (offset: 0x6804)

Name	Bits	Access	Default Value	Description
ID7	31	RO	ID7	Bond ID 7
ID6	30	RO	ID6	Bond ID 6
Disable GRC Reset on PCIE block	29	RW	0	Setting this bit will prevent reset to PCIE block.
ID5	28	RO	ID5	Bond ID 5
ID4	27	RO	ID4	Bond ID 4

Name	Bits	Access	Default Value	Description
Reserved	26	RW	0	–
Reserved	25	RW	0	–
Reserved	24	RW	0	–
Reserved	23	RW	0	–
Reserved	22	RW	0	–
Reserved	21	RW	0	–
Powerdown	20	RW	0	Setting this bit will power down the device (power consumption is ~20 mW). This bit is cleared by PCI reset.
PME EN State	19	RO	1	State of PME Enable 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
Bond ID	16:13	RO	ID 3:0	Bond ID
Reserved	12:8	RO	0	–
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.
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 disable Traffic LED Fix.
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.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	25	RO	0	–
Auto SEEPROM Access	24	RW	0	If set, access to serial EEPROM goes through the serial EEPROM address and data registers. Otherwise, serial EEPROM control register should be used.
APE_GPIO IN(6:0)	23:17	RO	APE GPIO default setting	APE GPIO Status Holds the value of APE GPIO (6:0) pins
GPIO(2:0) Output	16:14	RW	0	Outputs which are defined by board level design.
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.
GPIO(3) Output	7	RW	0	GPIO3 Output value.
GPIO(3) Output Enable	6	RW	0	When set to 1, GPIO3 pin will be enabled as an output pin.
GPIO(3) Input	5	RW	0	Input value of GPIO3.
Reserved	4	RO	0	–
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;
SW Event 12	30	RO	0	SW Event 12 is set; This bit is VPD Attention
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

Name	Bits	Access	Default Value	Description
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

Serial EEPROM Address Register

This 32-bit register is used by the RISCs in conjunction with the Serial EEPROM Data Register to read and/or write serial EEPROM data. The address register specifies the address and the direction of the transfer. When the transfer is complete (for either a read or a write), the complete bit is set.

To use this register pair to read the serial EEPROM, set the address and ensure the read/write bit is set in the address register. Loop reading the address register until the complete bit is set. When it is read the data from the data register. Clear the complete bit by writing the bit. No other transfer will occur when the complete bit is set. The Device ID must be programmed to select the appropriate device (A2 must be 0 for 128K/256Kx8 device).

To use this register pair to write the serial EEPROM, place the data into the data register. Then write the address into the address register ensuring that the write bit is clear. Loop reading the address register until the complete bit is set. When it is, the write is complete. Clear the complete bit by writing the bit. No other transfer will occur when the complete bit is set. It is the responsibility of software to control the timing between successive read/write access to the serial EEPROM.

All registers reset are core reset unless specified.

Serial EEPROM Delay Register (offset: 0x6848)

This 32-bit RW register specifies the delay between the EEPROM access in 15 ns interval and is used for VPD access. Since the requirement of back-to-back write for Serial EEPROMs is 10ms, firmware currently programs this register to 0xA2C2A.

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	–
VPD	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 hardware, cleared by software.
Reserved	23	RW	0	–
High Priority Mailbox	22	RW	0	–
Low Priority Mailbox	21	RW	0	–

Name	*Bits*	*Access*	*Default Value*	*Description*
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	–
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	–

Miscellaneous Control Registers

Miscellaneous Control Register (offset: 0x6890)

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31	RW	0	–
Cq14161_fix_ena_n	30	RW	0	This bit, when clear, enables the CQ14161 fix where the proper Bus Number & Device Number are returned in the completion prior to the initial Configuration Write Request. 0: Enable Fix 1: Disable Fix This bit is reset by Hard_Reset (POR, Exit Low Power mode, Lost of Vmain while in D0)
Reserved	29:25	RW	0	–
MISC2 Bit	24	RW	0	Reserved RW bit that gets reset by Power-On-Reset.
MISC 1 Bits	23:1	RW	0	Reserved RW bits that get reset by GRC Reset.
Reserved	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 codestatus.
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 (offset: 0x68A4)

Some of the bit in this register is initialized by hard_reset and GRC Reset.

Name	*Bits*	*Access*	*Default Value*	*Description*
Powerdown Restart PCIE PLL Enable	31	RW	0	1: Enable restart PCIE PLL when PCIE PLL is locked up in the powerdown mode, or IDDQ mode, or during POR, or any combinations. 0: Disable restart PCIE PLL in powerdown mode. This bit is NOT Self-Clear. Software need to generate a pulse by writing a 1 followed by 0 in order to restart the PLL. It's for debug purpose.
Normal Restart PCIE PLL Enable	30	RW	0	1: Enable restart PCIE PLL in normal mode. Firmware needs to write 1 and then write 0 to restart it (pulse restart). 0: Disable restart PCIE PLL in normal mode. This bit is NOT Self-Clear. Software need to generate a pulse by writing a 1 followed by 0 in order to restart the PLL. It's for debugging purposes.
Select Core Clock Override	29	RW	0	1: Enable switching of core clock to be used for PCIE block clocks. 0: Disable switching of core clock to be used for PCIE block clocks. This bit is the same as bit 16 in the PCIE Data Link Layer Register 0x7D00.
Reserved	28:18	RO	0	–
Reserved	17	RW	0	–
PERST Override	16	RW	0	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
Reserved	15:10	RO	0	–
irefselo	9	RO	X	PCIE Reference Select. 0: PCIE 100 MHz Reference Clock 1: Local 25 MHz Crystal
irxFastAcq	8	RO	X	PCIE PLL Fast Acquisition Select. 0: Disable Fast Acquisition 1: Enable Fast Acquisition
irxSeqStart	7	RO	X	PCIE RX Sequence Start
ipllSeqStart	6	RO	X	PCIE PLL Sequence Start
irxpowerdown	5	RO	X	RX Power Down Status

Name	Bits	Access	Default Value	Description
itxpowerdown	4	RO	X	TX Power Down Status
ipllpowerdown	3	RO	X	PLL Power Down Status
Iclkreq_oe_l	2	RO	X	Clock Request Output Enable
obsvElecIdle	1	RO	X	Observe Electrical Idle Status
PLLisUp	0	RO	X	PCIE PLL Status

5755ME Miscellaneous Control Register (offset: 0x68B0)

Name	Bits	Access	Default Value	Description
Reserved	31:19	RW	X	–
EMAC Legacy Bug Fix	18	RW	0	This bit when set enable the EMAC to receive packet without dropping it after receiving a short invalid packet size. 1: Enable Bug Fix 0: Disable Bug Fix
Reserved	17:0	RW	X	–

Memory TM control1 (offset: 0x68E0)

Name	Bits	Access	Default Value	Description
Reserved	31:24	RW	0x0	–
Memory TM control retrybuf	23:16	RW	0x00	TM control of retry buffer
Memory TM control txmbuf	15:8	RW	0x00	TM control for txmbuf
Memory TM control rxmbuf	7:0	RW	0x00	TM control for rxmbuf

Memory TM control 2 (offset: 0x68E4)

Name	Bits	Access	Default Value	Description
Reserved	31:29	RW	0x0	–
APE ROM tm	28:24	RW	0x0	TM control for APE ROM (mac0 only).
APE SCPAD2 mem tm	23:20	RW	0x0	TM control for APE SCPAD 0 (mac0 only).
APE SCPAD1 mem tm	19:16	RW	0x0	TM control for APE SCPAD 0 (mac0 only).
APE SCPAD0 mem tm	15:12	RW	0x0	TM control for APE SCPAD 0 (mac0 only).
APE ARB mem tm	11:10	RW	0x0	TM control for APE ARB.

Name	Bits	Access	Default Value	Description
APE ATB mem tm	9:8	RW	0x0	TM control for APE ATB.
APE shared mem tmb	7:4	RW	0x0	TMb control for APE shared mem.
APE shared mem tma	3:0	RW	0x0	TMa control for APE shared mem.

Mem TM control 3(offset: 0x68E8)

Name	Bits	Access	Default Value	Description
Reserved	31:9	RW	0x0	–
Memory TM control boot rom	8:4	RW	0x00	TM control for rom
Memory TM control scratchpad	3:0	RW	0x00	TM control for scratchpad

Expansion ROM Address Register (offset: 0x68EC)

Name	Bits	Access	Default Value	Description
Expansion ROM Test bits	31:24	RW	0	Reserved–keep at 0 for normal operation.
Expansion ROM base address	23:0	RW	0	Expansion ROM base address, expect to be d-word aligned.

BCM5719/BCM5720 Registers

The registers in this section are available only for the BCM5719 and BCM5720.

Mem TM Control 4 (offset: 0x68F8)

Name	Bits	Access	Default Value	Description
Reserved	31:28	RW	0x0	Reserved
WDMA FIFO tmb control	27:26	RW	0x00	TM control for WDMA engine memory.
WDMA FIFO tma control	25:24	RW	0x00	TM control for WDMA engine memory.
Non-LSO Ch3 RDMA FIFO tmb control	23:22	RW	0x00	TM control for Ch3 non-LSO RDMA engine memory.
Non-LSO Ch3 RDMA FIFO tma control	21:20	RW	0x00	TM control for Ch3 non-LSO RDMA engine memory.

Name	Bits	Access	Default Value	Description
Non-LSO Ch2 RDMA FIFO tmb control	19:18	RW	0x00	TM control for Ch2 non-LSO RDMA engine memory.
Non-LSO Ch2 RDMA FIFO tma control	17:16	RW	0x00	TM control for Ch2 non-LSO RDMA engine memory.
Non-LSO Ch1 RDMA FIFO tmb control	15:14	RW	0x00	TM control for Ch1 non-LSO RDMA engine memory.
Non-LSO Ch1 RDMA FIFO tma control	13:12	RW	0x00	TM control for Ch1 non-LSO RDMA engine memory.
Non-LSO Ch0 RDMA FIFO tmb control	11:10	RW	0x00	TM control for Ch0 non-LSO RDMA engine memory.
Non-LSO Ch0 RDMA FIFO tma control	9:8	RW	0x00	TM control for Ch0 non-LSO RDMA engine memory.
BD RDMA FIFO tmb control	7:6	RW	0x00	TM control for BD RDMA engine memory.
BD RDMA FIFO tma control	5:4	RW	0x00	TM control for BD RDMA engine memory.
LSO RDMA FIFO tmb control	3:2	RW	0x00	TM control for LSO RDMA engine.
LSO RDMA FIFO tma control	1:0	RW	0x00	TM control for LSO RDMA engine.

TPH Hint Register (Offset: 0x68FC)

This register can enable TLP Hint assertion and Processing Hint on each Host Bound DMA transaction classes individually.

Name		Bits	Access	Default Value	Description
Reserved		31:28	RW	0	–
MSI/MSI-X Vector Write	Resvd	27	RW0	0	–
	PH Value	26:25	RW	10	–
	TH Enable	24	RW	0	–
Status Block Write	Resvd	23	RW	0	–
	PH Value	22:21	RW	10	–
	TH Enable	20	RW	0	–
Receive Return BD Write	Resvd	19	RW	0	–
	PH Value	18:17	RW	10	–
	TH Enable	16	RW	0	–
Packet Data Write	Resvd	15	RW	0	–
	PH Value	14:13	RW	10	–
	TH Enable	12	RW	0	–

Name		Bits	Access	Default Value	Description
Receive Producer BD Read	Resvd	11	RW	0	–
	PH Value	10:9	RW	10	–
	TH Enable	8	RW	0	–
Send BD Read	Resvd	7	RW	0	–
	PH Value	6:5	RW	10	–
	TH Enable	4	RW	0	–
Packet Data Read	Resvd	3	RW	0	–
	PH Value	2:1	RW	10	–
	TH Enable	0	RW	0	–

EAV Ref Count Capture LSB Reg (offset: 0x6900)

The MSB and LSB registers are interface to the actual EAV Ref Counter hardware. The Counter value may be read via this pair anytime and even be overwritten by this pair anytime. While reading the pair, the hardware Counter does not stop, only its value is latched in this pair.

The only two legal sequences of accessing this pair is Read-LSB followed by Read-MSB and Write-LSB followed by Write-MSB.

Name	Bits	Access	Default Value	Description
EAV Reference Count [lower half]	31:3	RW	UUUU	LSB of the EAV Reference Count–Reading this LSB latches a Count in this pair until the time the corresponding MSB is read. Writing to this LSB latches the value in this pair and the subsequent write to the MSB transfers the 64-bit value to EAV Ref Counter and counting immediately resumes from there.
Reserved	2:0	RO	000	[2:0] shall always be 000

EAV Ref Count Capture MSB Reg (offset: 0x6904)

Name	Bits	Access	Default Value	Description
EAV Reference Count [Upper half]	31:0	RW	UUUU	MSB of the EAV Reference Count–See the pairing LSB register. Reading this register returns the MSB of the 64-bit EAV Ref count previously latched by performing an LSB read. Writing to this MSB transfers the 64-bit value, this plus previously latched LSB, to EAV Ref Counter and counting immediately resumes from there. Back to back writes to this MSB has no effect.

EAV Ref Clock Control Reg (offset: 0x6908)

This register controls the EAV Reference Counter and the TimeSync related GPIO pins. Each MAC's 1588 hardware owns a dedicated TimeSync_GPIO pin which may be connected to any of its four Snap-shot/ WatchDog hardware 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 hardware and four MAC-1588 hardware. Therefore, a platform must design-in these pins and have individual BootCode or firmware configure this register and APE-GPIO register accordingly.

Note: Hardware 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
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	0x0	–
Reset on Network Link Down -> Up	11	RW	0	–
Reset on Network Link Up -> Down	10	RW	0	–
Reset on GRC Reset & PCIe FLR	9	RW	0	Reset on GRC Reset pulse.
Reset on PCIe reset	8	RW	0	Reset on de-asserting edge of PCIe Reset.

Name	Bits	Access	Default Value	Description
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 Snapshot LSB[0] Reg (offset 0x6910)

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 snapshot by Timesync/APE GPIO. [2:0] shall always be 000

EAV Ref Count Snapshot MSB[0] Reg (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 snapshot by TimeSync/APE GPIO.

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)" on page 488.

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].

Name	Bits	Access	Default Value	Description
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[1] Reg (offset: 0x6924)" on page 489.

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 required Timesync/APE GPIO toggles as soon as EAV Reference Count[62:0] increments to match this 63-bit value. ***Note:*** Setting this time value back in time produces no toggle.

EAV Ref Corrector Reg [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	If 0, the correction is an addition. If 1, the correction is a subtraction.
Reserved	29:24	RO	0x00	Reserved

Name	Bits	Access	Default Value	Description
Correction Value	23:0	RW	0x000001	This value is accumulated in an accumulator every EAV REF CLK tick until it overflows–upon which the EAV REF COUNT is corrected by a 8ns unit & the acc is reloaded. A 0x0 value is not permissible.

EAV Ref Count Snapshot 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 snapshot by TimeSync/APE_GPIO. [2:0] shall always be 000

EAV Ref Count Snapshot 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 snap-shot by TimeSync/APE_GPIO.

Non-Volatile 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	26:20	RO	0	–
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.

NVM Write Register (offset: 0x7008)

Name	Bits	Access	Default Value	Description
Write Data	31:0	RW	0	32bits of write data are used when write commands are executed.

NVM Address Register (offset: 0x700C)

Name	Bits	Access	Default Value	Description
Reserved	31:24	RO	0	–
Address	23:0	RW	0	The 24 bit address for a read or write operation (must be 4 byte aligned).

NVM Read Register (offset: 0x7010)

Name	Bits	Access	Default Value	Description
Read Data	31:0	RO	0	32bits of read data are used when read commands are executed.

NVM Config 1 Register (offset: 0x7014)

Name	Bits	Access	Default Value	Description
SEE_CLK_DIV Enable	31	0	RW	This bit enables 0x7014[21:11] as the SEE_CLK_DIV count for EPROM clock generation. 1 = Enable 0 = Disable
Page Size	30:28	Depends on flash strapping	RW	These bits indicate the page size of the attached flash device. They 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: reserved 111b: reserved
Reserved	27	RO		–
Reserved–Safe Erase	26	RO	0	–
Flash Size–strap bit 3	25	RO	Pin	–
Protect Mode–strap bit 2	24	RO	pin	–
Strap bit 5	23	RO	pin	–
Strap bit 4	22	RO	pin	–
SEE_CLK_DIV	21:11	RW	16	This field is a divisor used to create all 1x times for all SEEPROM interface I/O pin timing definitions. A value of 0 means that an SCL transitions at a minimum of each CORE_CLK rising edge. The equation to calculate the clock freq. for SCL is: CORE_CLK / ((SEE_CLK_DIV + 1) * 4)

Name	Bits	Access	Default Value	Description
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)
Status	6:4	RW	0 if flash_ mode. 7 if buffer_ mode. X other-wise	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: 0x7018)

This register is reset by POR only.

Name	Bits	Access	Default Value	Description
Reserved	31:24	RO	0	–
Status Command	23:16	RW	0x05 if pin strap = ST 0xD7 if pin strap = Atmel	This is the Flash status register read command.
Dummy	15:8	RW	X	–
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	0x0B 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, software entity must win arbitration before accessing the NVRAM.

Name	*Bits*	*Access*	*Default Value*	*Description*
Reserved	31:24	RO	0	–
REQ5	23	RO	0	Software request bit 5. 1 in this bit indicates that the request5 is active.
ARB_WON5	22	RO	0	Arbitration won bit 5 (see bit 8, ARB_WON0).
REQ_CLR5	21	WO	X	Write 1 to clear REQ5 bit.
REQ_SET5	20	WO	X	Write 1 to set REQ5 bit.
REQ4	19	RO	0	Software request bit 4. 1 in this bit indicates that the request4 is active.

Name	*Bits*	*Access*	*Default Value*	*Description*
ARB_WON4	18	RO	0	Arbitration won bit 4(see bit 8, ARB_WON0)
REQ_CLR4	17	WO	X	Write 1 to clear REQ4 bit
REQ_SET4	16	WO	X	Write 1 to set REQ4 bit
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: 0x7024)

Name	Bits	Access	Default Value	Description
Reserved	31:5	RO	0	–
Disable Auto EEPROM reset	4	RW	1'b0	This bit controls the nvm state machine to issue EEPROM reset. 1'b0: Issue EEROM reset sequence before any EPROM RW access right after a hardware or software reset. 1'b1: EEPROM reset sequence is not issued automatically.
EPROM SDA_OE mode	3	RW	1'b1	This bit controls EPROM SDA_OE generation.
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 bit21 of the mode control register 0x6800.
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)

Name	Bits	Access	Default Value	Description
Reserved	31-16	–	–	–
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).

Arbitration Watchdog Timer Register (offset: 0x702C)

Name	Bits	Access	Default Value	Description
Reserved	31-0	RO	1	–
Reserved	27-24	RO	0	–
Reserved	23:8	RW	0x1000000	–
Reserved	7		0	–
Reserved	6	RW	0	–
Reserved	5	RW	1	–
Reserved	4-0	RO	0	–

NVM Auto-Sense Status Register (offset: 0x7038)

Name	Bits	Access	Default Value	Description
Reserved	31:21	RO	0	–
Auto Detected Device ID	20:16	RO	0x1F	AT45DB011D: 5'b0_0100; AT45DB021D: 5'b0_0011; AT45DB041D: 5'b0_0000; STM25PE10: 5'b0_1011; STM25PE20: 5'b0_1010; STM25PE40: 5'b0_1000; STM45PE10: 5'b0_1100; STM45PE20: 5'b0_1101; STM45PE40: 5'b0_1110; Unsupported AT26: 5'b1_0000; Unsupported AT45: 5'b1_0001; Unsupported ST25: 5'b1_1000; Unsupported ST45: 5'b1_1100 Device Unknown: 5'b1_1111;
Reserved	15:13	RO	0	–
Auto-config State	12:8	RO	0	Auto Config FSM state.
Reserved	7:6	RO	0	–
Auto Config Successful	5	RO	0	Auto config is successful.
Auto Config Enable	4	RO	0	Auto config feature is enabled through pin strap.
Reserved–Auto_conf_states	3:1	RO	0	The auto-sense FSM state.
Auto_config_busy	0	RO	0	1: auto-config FSM is busy 0: auto-config is complete

Section 14: 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 318) of the MAC. The integrated transceiver contains the set of registers shown in the tables below.

BCM5718 Family MII Bus PHY Addressing

Table 115: BCM5717

Port 0		*Port 1*	
Block	***PHY Address***	***Block***	***PHY Address***
GPHY	0x01	GPHY	0x02

Table 116: BCM5718

Port 0		*Port 1*	
Block	***PHY Address***	***Block***	***PHY Address***
GPHY	0x01	GPHY	0x02
SERDES	0x08	SERDES	0x09

Table 117: BCM5719

Port 0		*Port 1*		*Port 2*		*Port 3*	
Block	***PHY Address***	***Block***	***PHY Address***	***Block***	***PHY Address***	***Block***	***PHY Address***
GPHY	0x01	GPHY	0x02	GPHY	0x03	GPHY	0x04
SERDES	0x08	SERDES	0x09	SERDES	0x0A	SERDES	0x0B

Table 118: BCM5720

Port 0		*Port 1*	
Block	***PHY Address***	***Block***	***PHY Address***
GPHY	0x01	GPHY	0x02

Table 118: BCM5720 (Cont.)

Port 0		*Port 1*	
Block	***PHY Address***	***Block***	***PHY Address***
SERDES	0x08	SERDES	0x09

Register Field Access Type

RW = 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
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

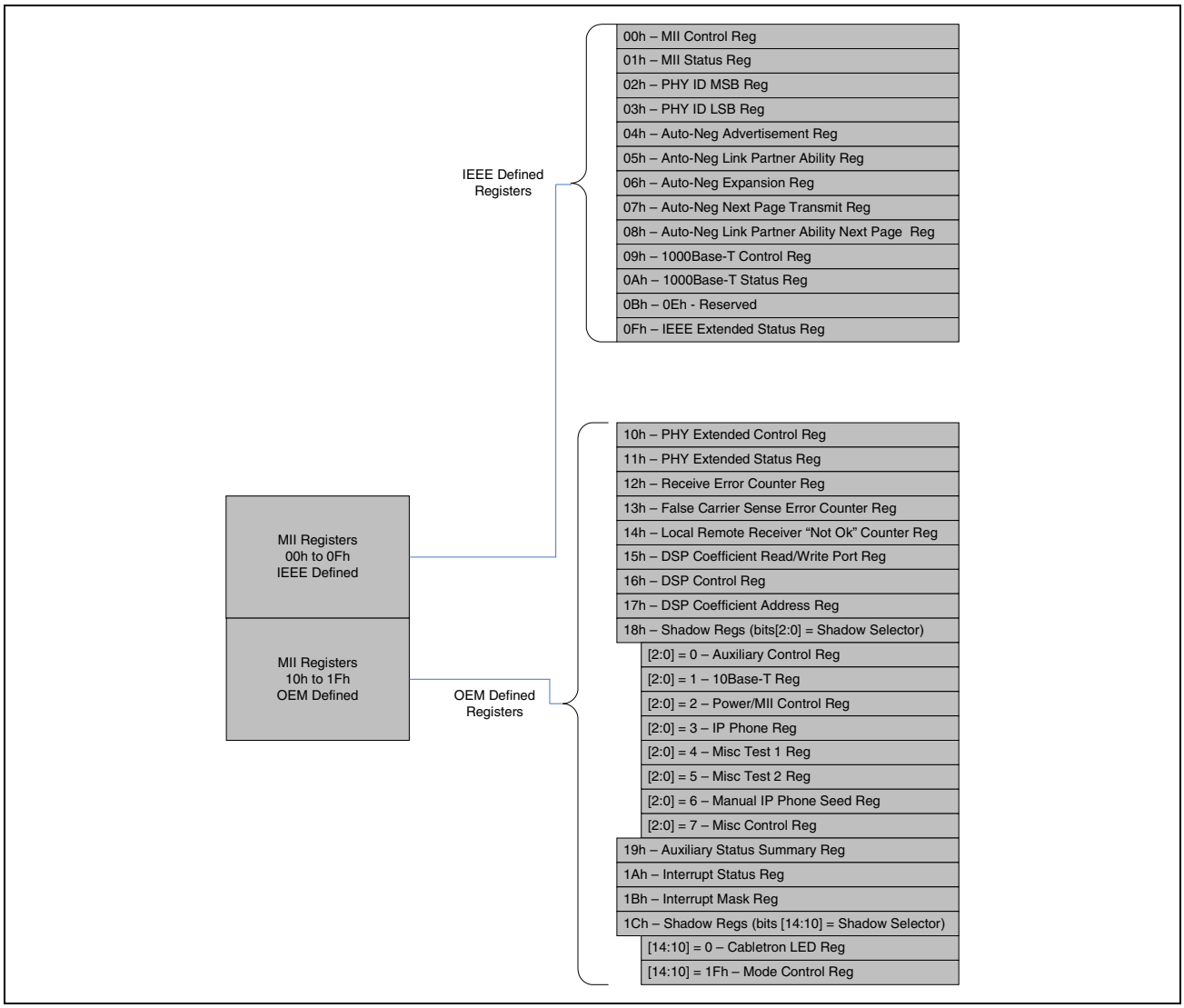
(Cont.)

Address	*Name*
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

(Cont.)

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

Figure 58: Copper PHY Register Mapping Table



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*	*RW*	*Default*	*Description*
15	RESET	RW SC	0	1 = PHY Reset 0 = normal operation
14	LOOPBACK	RW	0	1 = loopback mode 0 = normal operation
13	SPEED_SELECT_LSB	RW	1	0.6, 0.13: 11 = Reserved 10 = 1000 Mbit/s 01 = 100 Mbit/s 00 = 10 Mbit/s
12	AUTONEGOTIATION_ENABLE	RW	1	1 = auto-negotiation enabled 0 = auto-negotiation disabled
11	POWER_DOWN	RW	0	1 = low power mode 0 = normal operation
10	ISOLATE	RW	0	1 = isolate PHY from MII 0 = normal operation
9	RESTART_AUTONEGOTIATION	RW SC	0	1 = restart auto-negotiation process 0 = normal operation
8	DUPLEX_MODE	RW	1	1 = full duplex 0 = half duplex
7	COLLISION_TEST	RW	0	1 = collision test mode enabled 0 = collision test mode disabled
6	SPEED_SELECT_MSB	RW	0	0.6, 0.13: 11 = Reserved 10 = 1000 Mbit/s 01 = 100 Mbit/s 00 = 10 Mbit/s
5	UNIDIRECTIONAL_ENABLE	RW	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
4:0	RESERVED	RW	000000	write as 0, ignore on read

01h: MII_Status_Register

Bit	*Name*	*RW*	*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
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

Table 119: 02h: PHY_Identifier_MSB_Register

Bit	Name	RW	Default	Description
15:0	OUI_MSB	RO	0362h	Bits 3:18 of organizationally unique identifier.

03h: PHY_Identifier_LSB_Register

Table 120: 03h: PHY_Identifier_LSB_Register

Bit	Name	RW	Default	Description
15:10	OUI_LSB	RO	010111	Bits 19:24 of organizationally unique identifier.
9:4	MODEL	RO	see chart	Device model number (metal programmable) 5717B0 = 100000 5717C0 = 110110 5718 = 100000 5719 = 100010 5720 = 110110
3:0	REVISION	RO	0000	Device revision number (metal programmable)

04h: Auto_Negot_Advertisement_Register

Bit	Name	RW	Default	Description
15	NXT_PAGE	RW	0	1 = next page ability supported 0 = next page ability not supported
14	RESERVED	RW	0	write as 0, ignore on read
13	REMOTE FAULT	RW	0	1 = advertise remote fault detected 0 = advertise no remote fault detected
12	RESERVED	RW	0	write as 0, ignore on read
11	ASYMMETRIC_PAUSE	RW	0	1 = Advertise asymmetric pause 0 = Advertise no asymmetric pause
10	PAUSABLE	RW	1	1 = capable of full duplex Pause operation 0 = not capable of Pause operation
9	100BASET4_CAPABLE	RW	0	1 = 100BASE-T4 capable 0 = not 100BASE-T4 capable
8	100BASETX_FULL_DUPLEX_CAPABLE	RW	1	1 = 100BASE-TX full duplex capable 0 = not 100BASE-TX full duplex capable
7	100BASETX_HALF_DUPLEX_CAPABLE	RW	1	1 = 100BASE-TX capable 0 = not 100BASE-TX capable

(Cont.)

Bit	Name	RW	Default	Description
6	10BASET_FULL_DUPLEX_CAPABLE	RW	1	1 = 10BASE-T full duplex capable 0 = not 10BASE-T full duplex capable
5	10BASET_HALF_DUPLEX_CAPABLE	RW	1	1 = 10BASE-T half duplex capable 0 = not 10BASE-T half duplex capable
4:0	PROTOCOL_SELECT	RW	00001	00001 = IEEE 802.3 CSMA/CD

05h: Auto_Negot_Link_Partner_Ability_Base_Pg_Register

Bit	Name	RW	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 encoding)

* RW when "writeable link partner ability test mode" (reg 1Fh bit 10) is set

06h: Auto_Negot_Expansion_Register

Bit	Name	RW	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	RW	Default	Description
15	NXT_PG	RW	0	1 = additional next pages will follow 0 = sending last page
14	RESERVED	RO	0	write as 0, ignore on read
13	MESSAGE_PAGE	RW	1	1 = message page 0 = unformatted page
12	ACKNOWLEDGE 2	RW	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	RW	0000000001	Message code field or unformatted code field

08h: Auto_Negot_Link_Partner_Ability_Nxt_Pg_Register

Bit	Name	RW	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

* RW 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	RW	Default	Description
15:13	TEST_MODE	RW	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 (#)	RW	0	1 = enable Master/Slave manual config value 0 = disable Master/Slave manual config value
11	MASTER_SLAVE_CONFIG_ VALUE (#)	RW	0	1 = configure PHY as Master when 9.12 is set 0 = configure PHY as Slave when 9.12 is set
10	REPEATER_DTE (#)	RW	0	1 = Repeater/switch device port 0 = DTE device port
9	ADV_1000BASE_T_FULL_ DUPLEX (#)	RW	0	1 = Advertise 1000BASE-T full duplex capable 0 = Advertise not 1000BASE-T full duplex capable

Bit	*Name*	*RW*	*Default*	*Description*
8	ADV_1000BASE_T_HALF_DUPLEX (#)	RW	0	1 = Advertise 1000BASE-T half duplex capable 0 = Advertise not 1000BASE-T half duplex capable
7:0	RESERVED	RW	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	RW	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 & *=RW when "writeable link partner ability test mode" (reg 1Fh bit 10) is set

0Eh: BroadReach LRE Access Register

Bit	Name	RW	Default	Description
15:3	RESERVED	RO	000h	Ignore on Read
2	ENABLE LRE REGISTER ACCESS OVERRIDE	RW	0	1 = Allow access to BroadReach LRE Registers 0 = BroadReach LRE Registers enabled only by normal operation
1	LRE REGISTER OVERRIDE VALUE	RW	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*	*RW*	*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–1Fh Register Map Detailed Description

10h: PHY_Extended_Control_Register

Bit	*Name*	*RW*	*Default*	*Description*
15	MAC_PHY_INTERFACE_MODE	RW	0	1 = 10B interface mode 0 = GMII mode
14	DISABLE_AUTOMATIC_MDI_ CROSSOVER	RW	0	1 = automatic MDI crossover disabled 0 = automatic MDI crossover enabled
13	TRANSMIT_DISABLE	RW	0	1 = force transmit output to high impedance 0 = normal operation
12	INTERRUPT_DISABLE	RW	0	1 = interrupts disabled 0 = interrupts enabled
11	FORCE_INTERRUPT	RW	0	1 = force interrupt status to "active" 0 = normal interrupt operation
10	BYPASS_ENCODER	RW	0	1 = bypass 4B5B encoder and decoder 0 = normal operation
9	BYPASS_SCRAMBLER	RW	0	1 = bypass scrambler and descrambler 0 = normal operation
8	BYPASS_NRZI_MLT3	RW	0	1 = bypass NRZI/MLT3 encoder and decoder 0 = normal operation
7	BYPASS_ALIGNMENT	RW	0	1 = bypass receive symbol alignment 0 = normal operation
6	RESET_SCRAMBLER	RW SC	0	1 = reset scrambler to all 1's state 0 = normal scrambler operation
5	ENABLE_LED_TRAFFIC_MODE	RW	0	1 = LED traffic mode enabled 0 = LED traffic mode disabled
4	FORCE_LEDS_ON	RW	0	1 = force all LED's into "ON" state 0 = normal LED operation
3	FORCE_LEDS_OFF	RW	0	1 = force all LED's into "OFF" state 0 = normal LED operation

Bit	Name	RW	Default	Description
2	BLOCK_TXEN_MODE	RW	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	RW	0	1 = able to transmit packets when no link 0 = requires link in order to transmit packets Draggy kell
0	GMII_RGMII_FIFO_ELASTICITY[0]	RW	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
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

Bit	Name	R/W	Default	Description
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	RW	Default	Description
15:0	RECEIVE_ERROR_COUNTER	RW 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	RW	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)	RW 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	RW	Default	Description
15:8	LOCAL_RECEIVER_NOT_OK_COUNTER	RW CR	00h	number of times local receiver status was not OK since last read. Freezes at FFh.
7:0	REMOTE_RECEIVER_NOT_OK_ COUNTER	RW CR	00h	number of times remote receiver status was not OK since last read. Freezes at FFh.
15:0	CRC_ERROR_COUNTER	RW 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.

18h: Auxiliary Control Register (Shadow Register Selector = "000")

Bit	*Name*	*RW*	*Description*	*Default*
15	EXTERNAL LOOPBACK	RW	1 = external loopback enabled 0 = normal operation	0
14	EXTENDED PACKET LENGTH	RW	1 = allow reception of extended length packets (GBIC) 0 = allow normal length Ethernet packets only	0
13:12	EDGERATE CONTROL (1000T) (LSB or'ed with ER pin)	RW	00 = 4.0ns (1000T) 01 = 5.0ns (1000T) 10 = 3.0ns (1000T) 11 = 0.0ns (1000T)	00
11	ENABLE SM_DSP CLOCK	RW	1 = Clock is enabled. 0 = Clock is gated off.	0
10	TRANSMIT 6dB CODING	RW	1 = transmit using 6dB coding 0 = transmit using 3dB coding	1
9:8	RECEIVE SLICING	RW	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	RW	1 = transmitter partial response filter disabled 0 = transmitter partial response filter enabled	0
6	DISABLE INVERSE PRF	RW	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)	RW	00 = 4.0 ns (100TX) 01 = 5.0 ns (100TX) 10 = 3.0 ns (100TX) 11 = 0.0 ns (100TX)	00
3	DIAGNOSTIC MODE	RW	1 = When convergence fails, hold in failed state until cleared 0 = Normal operation, retrain on failure	0

Bit	Name	RW	Description	Default
2:0	SHADOW REGISTER SELECTOR (These bits are written on all writes to 18h regardless of the value)	RW	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

Note: **Register Reads/Writes Description**

Write register 18h, bits [2:0] = 111 This selects the Miscellaneous Control register, shadow 7h.
All reads must be performed through the Miscellaneous Control register.
Bit 15 = 0 This allows only bits [14:12] and bits [2:0] to be written.
Bits [14:12] = zzz This selects shadow register zzz to be read.
Bits [11: 3] = <don't care> When bit 15 = 0, these bits are ignored.
Bits [2:0] = 111 This sets the Shadow Register Select to 111 (Miscellaneous Control register).
Read register 18h Data read back is the value from shadow register zzz.

Note: **Register Writes Description**

Set Bits [15:3] = Preferred write values Bits [15:3] contain the desired bits to be written to.
Set Bits [2:0] = yyy This enables shadow register yyy to be written.
For shadow 7h, bit 15 must also be written.

```
-----------------------------------------
PHY 0x18 Shadow 0x1 register read Procedure
-----------------------------------------
int value;

phy_write(0x18, 0x1007); //switch to shadow 0x1

valu = phy_read(0x18);


-----------------------------------------
PHY 0x18 Shadow 0x2 register write Procedure
-----------------------------------------
int wdata;

phy_write(0x18, 0x2007); //switch to shadow 0x2

phy_write(0x18, wdata | 0x2 );
```

18h: 10BASE-T Register (Shadow Register Selector = "001")

Bit	Name	RW	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)	RW	1 = block rxdv for 4 additional rxc cycles for ipg 0 = normal operation	0
11	10BT TXC INVERT MODE	RW	1 = invert TXC output 0 = normal operation	0
10	CLASS A/B LINE DRIVER SELECT (CLASSB_BT)	RW	1 = select class A line driver 0 = select class B line driver	0
9:	JABBER DISABLE	RW	1 = Jabber function disabled 0 = Jabber function enabled (half-duplex only)	0
8	10BT SIGNAL DETECT AUTOSWITCH	RW	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	RW	0 = high signal detect threshold 1 = low signal detect threshold	0
6	10BT ECHO MODE	RW	1 = echo transmit data to receive data 0 = normal operation	0
5	SQE ENABLE MODE	RW	1 = enable SQE 0 = disable SQE	0
4	10BASE-T NO DRIBBLE	RW	1 = correct 10BT dribble nibble 0 = normal operation	0
3	10BASE-T POLARITY ERROR COUNT MAX	RW	1 = 1 polarity error needed for 10BT polarity swap 0 = 7 polarity errors needed for 10BT polarity swap	0
2:0	SHADOW REGISTER SELECTOR	RW	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	RW	Description	Default
15:6	Reserved	–	–	–
5	SUPER ISOLATE	RW	1 = isolate mode with no linkpulses transmitted 0 = normal operation	0
4:3	Reserved	–	–	–
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	RW	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	RW	Description	Default
15	NON-STOP IP PHONE DETECT MODE	RW	1 = IP status will always update 0 = IP status will halt after detecting an IP PHONE	0
14:10	EXTENDED LINK PULSE WIDTH COUNTER	RW	00000 = normal link pulse width, otherwise additional link pulse width in 50 ns increments	00000
9	ALTERNATE RANDOM SEED GENERATION	RW	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	RW SC	1 = restart auto-negotiation 0 = normal operation	0
7	IP PHONE WINDOW MODE	RW	1 = enable IP Phone window detection mode 0 = normal operation	0
6	EXTENDED LINK PULSE WIDTH ENABLE	RW	1 = enable extended link pulse width transmission 0 = normal operation	0
5	ENABLE IP PHONE DETECTION	RW	1 = IP Phone detection enabled 0 = IP Phone detection disabled	0

Bit	Name	RW	Description	Default
4	DISABLE BLOCK LINK10 WINDOW MODE ON WRITES, IP PHONE MISMATCH ON READS	RW	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	RW	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)	RW	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	RW	Description	Default
15	REMOTE LOOPBACK ENABLE	RW	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	RW	1 = TDK fix (extend EOP on transmit 10BT packets)	1
13	ENABLE DEDICATED 10BT DLL BYPASS CLOCK	RW	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	RW	1 = prevent 10BT from restarting auto-negotiation in order to break the link 0 = normal operation	0

Bit	Name	RW	Description	Default
11	REMOTE LOOPBACK TRISTATE	RW	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	RW	1 = enable 10BT dac	0
9	10BT POLARITY BYPASS	RW	1 = enable polarity bypass 0 = normal operation	0
8	10BT IDLE BYPASS	RW	1 = enable idle bypass 0 = normal operation	0
7	10BT CLOCK RESET ENABLE	RW	1 = clock reset controlled from tpin11 0 = normal operation	0
6	10BT BYPASS ADC	RW	1 = bypass 10BT adc 0 = normal operation	0
5	10BT BYPASS CRS	RW	1 = bypass 10BT crs 0 = normal operation	0
4	SWAP RXMDIX	RW	1 = rx and tx operate on same pair 0 = normal operation	0
3	HALFOUT	RW	1 = transmit half amplitude, all speeds 0 = normal operation also see exp reg f9.1	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	RW	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	RW	Description	Default
15	COPPER ENERGY DETECT OVERRIDE	RW	1 = override copper energy detect 0 = normal operation	0
14	ADCFIFO_TX_FIX	RW	0 = enable adcFIFO fix for TX mode	0
13	Class A/B DVT enable (5478-A4/A5 and later)	RW	1 = enable, 0 = disable	0
12	CLASS A/B ENABLE (100tx speed)	RW	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	RW	00 = no scaling 01 = scaled by 0.5 10 = scaled by 0.25 others: no scaling	00
9	SPARE	RW	write as 0, ignore on read	0

Bit	Name	RW	Description	Default
8	Disable Auto Encoding Correction	RW	0 = Autoencoding correction enabled (overrides bits 6 and 7) 1 = Autoencoding correction disabled	0
7	Old PCS Encoding RX	RW	0 = Select IEEE compliant PCS encoding (for PCS receive) 1 = Select old PCS encoding (for PCS receive)	0
6	Old PCS Encoding TX	RW	0 = Select IEEE compliant PCS encoding (for PCS receive) 1 = Select old PCS encoding (for PCS receive)	0
5	Enable EC as NEXT	RW	1 = enable, 0 = disable	0
4	Enable force_mdix	RW	1 = enable, 0 = disable	0
3	En_PWRDNTDAC	RW	1 = enable, 0 = disable	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	RW	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	RW	Description	Default
15	WRITABLE IP PHONE SEED	RW	1 = use manual IP Phone seed 0 = normal operation	0
14	SPARE	RW	write as 0, ignore on read	0
13:3	LOCAL IP PHONE SEED	RW	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)	RW	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	RW	Description	Default
15	WRITE ENABLE (BITS [8:3])	RW SC	1 = write bits [8:3] 0 = only write bits [14:12]	0
14:12	SHADOW REGISTER READ SELECTOR	RW	000 = shadow register 0 read select 001 = shadow register 1 read select ... 111 = shadow register 7 read select	000
11	PACKET COUNTER MODE	RW	1 = count packets received 0 = count packets transmitted	0
10	BYPASS WIRESPEED TIMER	RW	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.	1
9	FORCE AUTO MDIX MODE	RW	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	0
8	RGMII TIMING MODE (output delay only)	RW	1 = clock delayed 90 degrees 0 = clock and data aligned	0
7	RGMII MODE	RW	1 = use reduced GMII mode 0 = normal GMII/MII operation	0

Bit	Name	RW	Description	Default
6	RGMII RXER MODE	RW	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	RW	1 = send regular rx data during IPG 0 = send out-of-band status info in RGMII mode	1
4	WIRESPEED ENABLE	RW	1 = enable wirespeed mode 0 = normal operation Note: Can be set only if gphy port wirespd_enable = 1.	0
3	MDIO ALL PHY SELECT	RW	1 = all phy selected during MDIO writes when the phy address = "00000" 0 = normal operation	0
2:0	SHADOW REGISTER SELECTOR (REFERENCE ONLY)	RW	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	RW	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
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

Bit	Name	RW	Description	Default
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	RW	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
	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

Bit	Name	RW	Description	Default
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	RW	Description	Default
15:0	INTERRUPT MASK VECTOR	RW	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")

Reading from and writing to register 1Ch is through register 1Ch bits [15:10]. Bits [14:10] set the shadow of register 1Ch, and bit 15 enables the writing of bits [9:0]. Setting bit 15 allows writing to bits [9:0] of register 1Ch. To read register 1C shadow zzzzz, first set writes to register 1Ch with bit 15 = 0, and bits [14:10] to zzzzz. The subsequent register read from register 1Ch contains the shadow zzzzz register value.

```
-----------------------------------------
PHY 0x1C Shadow 0x1 register read Procedure
-----------------------------------------
int value;

phy_write(0x1C, 0x0400); //switch to shadow 0x1

value = phy_read(0x1C);

return value;

-----------------------------------------
PHY 0x1C Shadow 0x2 register write Procedure
-----------------------------------------
int wdata;

phy_write(0x1C, 0x0800); //switch to shadow 0x2

phy_write(0x1C, wdata | 0x8800 );
```

Bit	*Name*	*RW*	*Description*	*Default*
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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	RW	Write as 0, ignore on read	-

Bit	Name	RW	Description	Default
7:0	CABLETRON LED REGISTER	RW	[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

1Ch: DLL Selection Register (Shadow Register Selector = "01h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00001
9:6	RESERVED	RW	write as 0, ignore on read	000
5:0	HIGH QUALITY CLOCK TEST MODE	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0

Bit	Name	RW	Description	Default
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00010
9	SD status	RO	1= sd input active	0
8	FORCE SD ON	RW	1= force sd on. disable sd pin input	0
7	INVERT SD PIN	RW	1 = invert sd pin	0
6	CFC_INITFILTER ENABLE	RW	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	RW	1= enable filter on sd input pin	0
4	100FX MODE COPPER PATH	RW	1 = enable 100BASE-FX on TRD+/- pins 0 = normal copper operation on MDI pairs	0
3	RESERVED	RW	write as 0, ignore on read	0
2	BICOLOR LINK SPEED LED MODE	RW	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	RW	When 0, enables lost token fix reset circuits	1
0	LINK LED MODE	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00011
9	GTXCLK delay bypass disable (input delay only)	RW	1 = do not bypass gtxclk delay 0 = bypass gtxclk delay	0
8	GMII CLOCK ALIGNMENT STROBE	RW	Delay value is latched into selected GMII clock delay line on rising edge of this bit.	0

Bit	Name	RW	Description	Default
7	RXCLK ALIGNMENT STROBE	RW	Delay value is latched into selected RX clock delay line on rising edge of this bit.	0
6:4	DELAY VALUE	RW	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	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00100
9	SPARE	RW	Flop exists, but does not change any function.	0
8	wirespeed retry disable (fixed in 54980)	RW	1 = downgrade after 1 failed link attempt 0 = use wirespeed retry limit (1c-04.4:2)	0
7	select tpout_rxd	RW	1 = tpout_rxd port muxed to rxd 0 = normal operation for rxd	0

Bit	Name	RW	Description	Default
6	DISABLE PHYA2	RW	1 = internally disable phya2 input (consult Testability document for suggested usage).	0
5	enable rbc0/1 & txc/rxc tri sate	RW	1 = enable tristating of rbc0/1 or txc/rxc 0 = rbc0/1 & txc/rxc not tristated.	0
4:2	WIRESPEED RETRY LIMIT	RW	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	RW	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	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00101
9	DLL lock enable during auto-power down	RW	1 = allow time for dll to lock before enabling clocks and 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	RW	1 = disable txc/rxc during auto-power down when there's no energy on the cable	0
7	10BT CARRIER REJECT FILTER ENABLE	RW	1 = enable 10BT 15 MHz Carrier Rejection Filter	0
6	TXC OFF ENABLE	RW	1 = gates off TXC output in 1000T/1000-X/SGMII slave 1000 mode	0
5	SD/energy detect change mux select	RW	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	RW	1 = disable low power ENC mode	1
3	DISABLE LOW POWER 10BASE-T LINK MODE	RW	1 = disable low power 10BASET link mode	1
2	SIGDET DEASSERT TIMER LENGTHEN	RW	1 = 100TX Sigdet Deassert Timer = 40 us 0 = Sigdet Deassert Timer =1 us	1

(Cont.)

Bit	Name	RW	Description	Default
1	AUTO-POWER DOWN DLL OFF DISABLE	RW	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	RW	1 = enable CLK125 output 0 = disable CLK125 output	1

1Ch: TDR Control 1 Register (Shadow Register Selector = "06h"

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00110
9	SPARE	RW	write as 0, ignore on read	0
8	TDR LINK TIME OUT	R	tdr linkpulse time out status	0
7:5	TEST PULSE SIZE	RW	size of test pulse in increments of 8 ns. minimum value = 1.	001
4:3	TX CHANNEL SEL	RW	channel to transmit test pulse	00
2:1	RX CHANNEL SEL	RW	channel to receive test data	00
0	TDR START/DONE	RW SC	write: 1: tdr start. self clearing read: 1: tdr done status	0

1Ch: TDR Control 2 Register (Shadow Register Selector = "07h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	00111
9:8	SPARE	RW	write as 0, ignore on read	00
7	PHASE STATUS	R	phase status	0
6	PHASE STATUS CLEAR	RW	1: clear phase status on bit 7	0
5	FASTTIMERS	RW	1: enable fasttimers	0
4	FEXT	RW	1: enable fext mode 0: disable fext mode	0

Bit	Name	RW	Description	Default
3	MASTER	RW	1: master mode 0: slave mode	0
2	EXTERNAL PHY NO AUTO-NEG	RW	1: tdr test with external phy without auto-negotiation 0: tdr test with external phy with auto-negotiation	0
1	EXTERNAL PHY	RW	1: tdr test with external phy 0: tdr test as stand-alone phy	0
0	TDR MODE ENABLE	RW	1: enable tdr mode 0: disable tdr mode	0

1Ch: LED Status Register (Shadow Register Selector = "08h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	01001

Bit	Name	RW	Description	Default
9	Collision blink led mode	RW	1 = blink fdxled when collision occurs 0 = fdxled indicates duplex status only	0
8	Activity link led MSB	RW	See description for bit 4 below.	0
7	SPARE	RW	write as 0, ignore on read	0
6	external_serdes_inuse led mode	RW	1= drive transmit led low when external SerDes is selected, inactive when not selected 0= normal operation	0
5	Override GBIC LED mode	RW	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	RW	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	RW	1 = normal operation (activity led indicates transmit or receive activity). 0 = activity led indicates receive activity only	1
2	REMOTE FAULT LED ENABLE	RW	1 = drive remote fault on quality led 0 = normal operation	0
1:0	LINK UTILIZATION LED SELECTOR (NORTEL LED)	RW	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: SGMII Slave Register (Shadow Register Selector = "15h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	10101

Bit	Name	RW	Description	Default
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	00000
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)	RW	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	00000
1	SGMII SLAVE MODE (register 1c shadow 1fh bit [2:1] must be "01")	RW	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	00000
0	SGMII SLAVE AUTO-DETECTION (register 1c shadow 1fh bit [2] must be 0)	RW	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	00000

1Ch: Misc 1000-X Control 2 Register (Shadow Register Selector = "16h")

Bit	Name	RW	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	RW	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")	RW	1 = enable remote copper mode (used in backplane applications to attach a remote phy to a mac). 0 = normal operation	00
8	ENABLE REMOTE ACCESS	RW	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	00
7	RESTART SERDES AUTONEG	RW 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	RW	Description	Default
6	REGISTER WRITE SELECTOR	RW	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	RW	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	1
4	SNOOP MODE (register 1c shadow 1fh bits[2:1] must be "00")	RW	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	0
3	FILTER FORCED LINK	RW	1 = synchronization status must be valid for 10 ms before link will come up when auto-negotiation is disabled. 0 = normal operation	1
2	DISABLE FALSE LINK	RW	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	RW	1 = turn auto-negotiation on/off in order to link up with link partner. Algorithm based on received code words 0 = disable parallel detection	1
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)	RW	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]	0

1Ch: Misc 1000-X Control Register (Shadow Register Selector = "17h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	10111
9	QUALIFY DISABLE CARRIER EXTEND AND DISABLE TRRR WITH FULL-DUPLEX	RW	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	RW	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	RW	1 = power down for 3 seconds before waking up 0 = power down for 5 seconds before waking up	0
6	SERDES TRANSMIT DISABLE	RW	1 = force all SerDes transmit data to 0 0 = normal operation	0
5	SIGNAL DETECT ENABLE	RW	1 = force synchronization to fail if signal detect is not active. Disabled if register 1ch shadow 14h [1] = 1. 0 = ignore signal detect pin	1
4	DISABLE GBIC UPDATES	RW	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	RW	1 = force xmit=data regardless of state of receive channel 0 = normal operation	0

Bit	Name	RW	Description	Default
2	DISABLE SERDES AMPLITUDE SWITCHING	RW	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	RW	1 = Fiber super-isolate	0
0	DISABLE 1000-X POWERDOWN	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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	RW	1 = enable SerDes crc checker to count fragments as crc errors 0 = ignore fragments as crc errors	0

Bit	Name	RW	Description	Default
3	MEDIA CONVERTER MODE (register 1c shadow 1fh bits [2:1] must be "11")	RW	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	0
2	GMII FIFO ELASTICITY [0] in SGMII/GBIC mode (1000T PCS transmit FIFO in SGMII/GBIC mode: from SerDes to copper link partner)	RW	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]	0
1	SGMII 10/100 RX FIFO FREQUENCY LOCK MODE (forced high internally when => sgmii_slave_mode and not remote_lpbk)	RW	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	RW	1 = enable auto-detection between SGMII and GBIC modes 0 = normal operation	0

1Ch: Test 1000-X Register (Shadow Register Selector = "19h")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	11001
9	USE COPPER SPEED	RW	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	RW	1 = enable dll bypass clock (tpin 11) 0 = disable dll bypass clock	0
7	TX FIFO SGMII 10/100 EARLY PREAMBLE MODE => towards SerDes link partner	RW	1 = enable transmitting early preamble in SGMII 10/ 100 mode in order to reduce latency in half-duplex. (TX FIFO) 0 = normal operation	0

Bit	Name	RW	Description	Default
6	RX FIFO SGMII 10/100 EARLY PREAMBLE MODE => towards copper link partner	RW	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	RW	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	RW	write as 0, ignore on read	0
3	FORCE TXFIFO ON (dig1000x_tx_fifo) towards SerDes link partner	RW	1 = force TXFIFO to be always active in 1000X 0 = normal operation	0
2	BYPASS PCS RECEIVE	RW	1 = bypass pcs receive 0 = normal operation	0
1	BYPASS PCS TRANSMIT	RW	1 = bypass pcs transmit 0 = normal operation	0
0	ZERO COMMA DETECT PHASE	RW	1 = force comma detect phase to zero 0 = normal operation	0

1Ch: Autoneg 1000-X Debug Register (Shadow Register Selector = "1ah")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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
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

Bit	Name	RW	Description	Default
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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	11011
9	USE SERDES MODE COUNTERS	RW	1 = use registers 12-14h for SerDes data 0 = normal operation	0
8	AUTONEG FAST TIMERS	RW	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	RW	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

Bit	Name	RW	Description	Default
6	DISABLE CARRIER EXTEND	RW	1 = force rxer, rxd to zeros in TRR+extend state (pcs receive state) 0 = normal operation	0
5	DISABLE TRRR	RW	1 = bypass extend_by_1 state (pcs transmit state) 0 = normal operation	0
4	DISABLE REMOTE FAULT SENSING	RW	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	RW	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	RW	1 = enable comma detection 0 = disable comma detection	1
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)	RW	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]	0
0	DISABLE CRC CHECKER	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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
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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	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	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	11110
9	2ND SERDES AUTO-DETECTION	RW	1 = enable dual SerDes auto-detect medium (switch between SGMII->fiber and SGMII->copper) 0 = disable dual SerDes auto-detect medium	0
8	INVERT FIBER SD FROM PIN	RW	1 = invert fiber sd from pin (active low pin) 0 = normal operation (active high pin)	0
7	FIBER IN-USE LED MODE	RW	1 = drive transmit led active low when fiber is selected, inactive when copper selected 0 = normal transmit led operation	0
6	FIBER LED MODE	RW	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	0
5	QUALIFY FIBER SD WITH SYNC_STATUS/100FX LINK	RW	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)	RW	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	0
3	SIGNAL DETECT ENABLE OVERRIDE	RW	1 = force signal detect enable to 1 in auto-detect medium block. Used for test purposes only! 0 = normal operation	0
2	AUTO-DETECT MEDIUM DEFAULT	RW	1 = fiber selected when no medium active 0 = copper selected when no medium active	0
1	AUTO-DETECT MEDIUM PRIORITY	RW	1 = fiber selected when both medium active 0 = copper selected when both medium active	1
0	AUTO-DETECT MEDIUM ENABLE	RW	1 = enable auto-detect medium (switch between GMII/RGMII->copper or GMII/RGMII->fiber) 0 = disable auto-detect medium	0

1Ch: Mode Control Register (Shadow Register Selector = "1fh")

Bit	Name	RW	Description	Default
15	WRITE ENABLE	RW	1 = write bits [9:0] 0 = read bits [9:0]	0
14:10	SHADOW REGISTER SELECTOR	RW	Shadow Register Selector	11111
9	DUAL SERDES CAPABLE	RO	1 = chip supports SGMII->fiber mode 0 = chip does not support SGMII->fiber mode	0
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
3	SERDES CAPABLE	RO	1 = SerDes capable device 0 = not SerDes capable device	0
2:1	MODE SELECT	RW	00 = copper-qsgmii 01 = fiber-qsgmii 10 = SGMII-copper 11 = GBIC	0
0	ENABLE 1000-X REGISTERS	RW	1 = select 1000-X regs 0-0fh 0 = select copper regs 0-0fh	0

1Dh: Master/Slave Seed Register (Bit 15 = 0)

Bit	Name	RW	Description	Default
15	ENABLE SHADOW REGISTER Writes to the selected register are done on a single cycle (no setup required).	RW	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	RW	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

* RW when "writeable link partner ability test mode" (reg 1Fh bit 10) is set

1Dh: HCD Status Register (Bit 15 = 1)

Bit	Name	RW	Description	Default
15	ENABLE SHADOW REGISTER Writes to the selected register are done on a single cycle (no setup required).	RW	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

Bit	Name	RW	Description	Default
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
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	RW	Description	Default
15	CRC ERROR COUNT VISIBILITY	RW	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	RW	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	RW	1 = forces counters into test mode 0 = normal operation	0
12	FORCE LINK	RW	1 = force link state machine into pass state 0 = normal operation	0
11	FORCE LOCK	RW	1 = force descrambler in to locked state 0 = normal operation	0
10	SCRAMBLER TEST	RW	1 = speed up descrambler unlock detect timer 0 = normal operation	0
9	EXTERNAL LINK	RW	1 = use tpin11 input as link status 0 = normal operation	0
8	FAST TIMERS	RW	1 = timers are sped up for LSI test 0 = normal operation	0
7	MANUAL SWAP MDI STATE	RW	1 = Swap 0 = off	0
6	RECEIVE WATCHDOG TIMER DISABLE	RW	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	RW	1 = disable 1000BASE-T polarity encoding 0 = normal operation	0
4	ENABLE SOFTWARE TRIM SETTING (MAIN DAC)	RW	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)	RW	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	RW	Description	Default
15:13	TEST SELECT AUTONEG FSM	RW	000 = ARB 001 = RX1000 010 = RX 011 = TX1000 100 = TX 101 = BASET LINK	000
12	TEST AUTO-NEG TIMER	RW	1 = auto-negotiation timer test mode 0 = normal operation	0
11	TEST MASTER/SLAVE SEED	RW	1 = use MDIO programmable master/slave seed 0 = normal operation	0
10	WRITEABLE LINK PARTNER ABILITY	RW	1 = link partner advertised ability may be overwritten by MII management 0 = normal operation	0
9	FORCE HCD	RW	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	RW	1 = link partner master/slave seed may be overwritten by MII management 0 = normal operation	0
7	TRANSMIT 10B MODE	RW	1 = force GMII transmit into 10B mode 0 = use normal mode bit	0
6	RECEIVE 10B MODE	RW	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)	RW	1 = transmit FIFO bypassed 0 = normal operation	0
4	SAME SCRAMBLER SEEDS	RW	1 = receive scrambler uses transmit seed 0 = normal operation	0
3	JITTER TEST MODE	RW	1 = jitter test mode 0 = normal operation	0
2	TEST ATMP COUNTER	RW	1 = force master slave seed attempt counter into test mode 0 = normal mode	0

Bit	Name	RW	Description	Default
1	LATENCY MEASURE	RW	1 = send special short packet to measure receive latency on remote PHY (hold high) 0 = normal operation	0
0	DISABLE ACTIVE HYBRID	RW	1 = active hybrid disabled 0 = active hybrid enabled	0

SerDes PHY Register Definitions

The PHY registers are broken into two blocks:

- Block 0 is for IEEE and non-IEEE controls.
- Blocks 0, 2 and 3 are non-IEEE blocks, where the analog section or the SerDes is controlled.

To access block 0, 2 or 3, write to the block number into the Block Address register at address 0x1F (reserved on each block).

Register Map

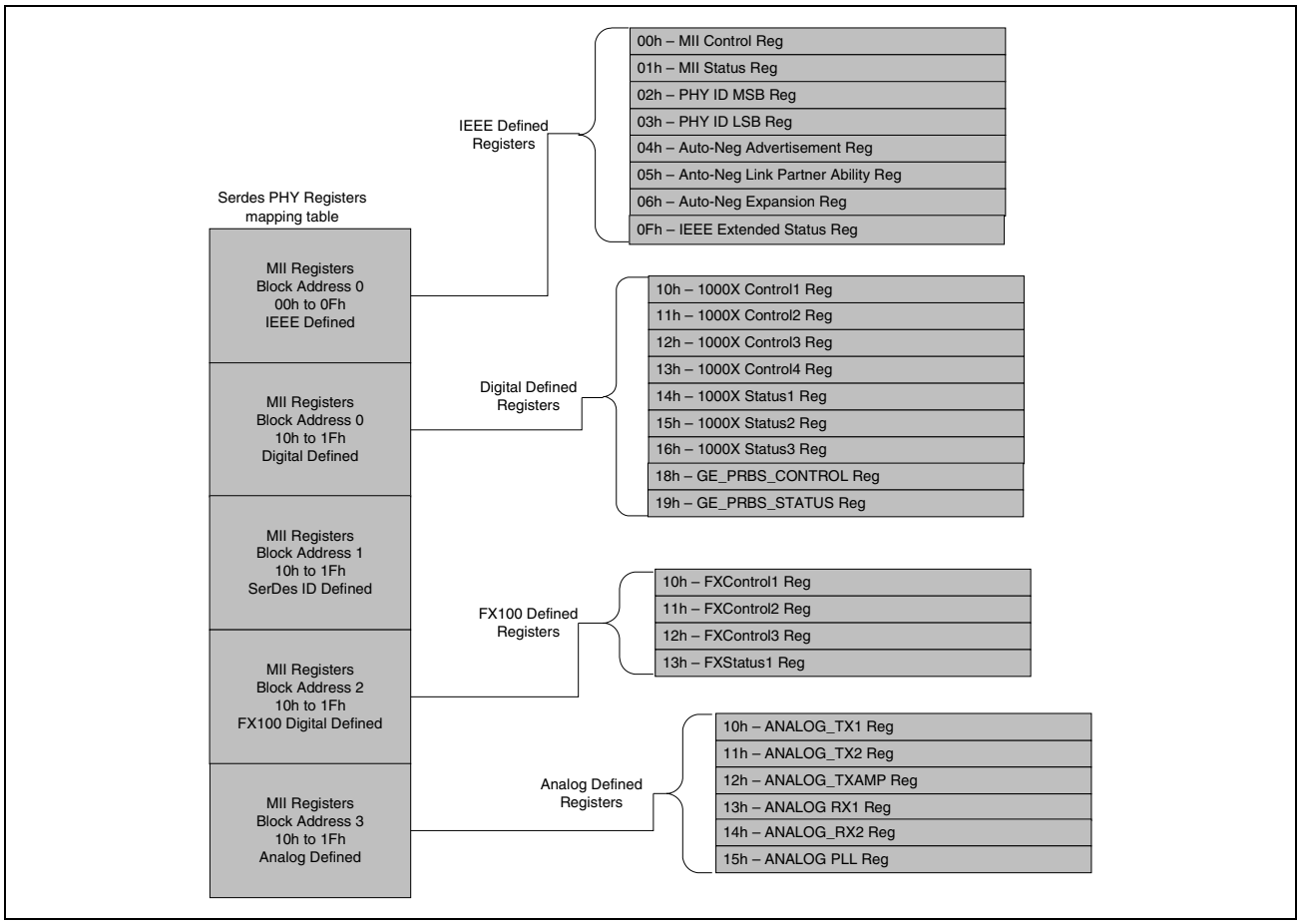
Table 121: GbE Port Internal PHY Register Map

Address	Block	Register Name	Description
00h	0	MII CONTROL	MII Control Register
01h	0	MII STATUS	MII Status Register
02h	0	PHY ID MSB	PHY_Identifier_MSB_Register
03h	0	PHY ID LSB	PHY_Identifier_LSB_Register
04h	0	AUTONEGADV	Auto-Negotiation advertisement register
05h	0	AUTONEG LINK PARTNER ABILITY	Auto-Negotiation Link Partner Ability
06h	0	AUTONEGEXPANSION	Auto-Negotiation Expansion register
07h–0Eh	0	–	Reserved
0Fh	0	EXTENDEDSTATUS	Extended Status register
10h	0	1000XCONTROL1	1000XControl 1 register
11h	0	1000XCONTROL2	1000XControl 2 register
12h	0	1000XCONTROL3	1000XControl 3 register
14h	0	1000XSTATUS1	1000XStatus 1 register
15h	0	1000XSTATUS2	1000XStatus 2 register
16h	0	1000XSTATUS3	1000XStatus 3 register
17h	0	–	Reserved
18h	0	GE_PRBS_CONTROL	Enable PRBS test mode
19h	0	GE_PRBS_STATUS	PRBS test mode status
1Ah–1Eh	0	–	Reserved

Table 121: GbE Port Internal PHY Register Map (Cont.)

Address	*Block*	*Register Name*	*Description*
10h	2	FXCONTROL1	100FX Enabling Control register
11h	2	FXCONTROL2	100FX Extended Packet Size register
12h	2	FXCONTROL3	100FX Control register
13h	2	FXSTATUS1	100FX Link Status register
14h–1Eh	2	–	Reserved
10h	3	ANALOG_TX1	Analog transmitter output for the SerDes when it is in SGMII mode.
11h	3	ANALOG_TX2	Analog transmitter2 output for the SerDes when it is in SGMII mode.
12h	3	ANALOG_TXAMP	Analog transmitter output for the SerDes when it is in FX (Fiber) mode.
13h	3	ANALOG_RX1	Controlling the SerDes receiver
14h	3	ANALOG_RX2	100FX Test Mode
18h	3	ANALOG_PLL	Controlling the GE PLL circuitry
1Fh	-	BLOCKADDRESS	Block address register

Figure 59: Serdes PHY Register Map



MII Control

Register Description: MII Control Register.

Register Offset: 0x0 at Block 0

Table 122: MII Control

Bits	Name	RW	Description	Default
15	RST_SW	RW	PHY Reset 0 = Normal Operation 1 = Reset	0
14	LOOPBACK	RW	Local loopback, data is looping back at the PHY before going out to the wire. 0 = Normal operation 1 = Loopback enable	0
13	SPD[0]	RW	Bit[0] of manual Speed[1:0] in SGMII mode only. This field is ignored in 1000Base-X operation. 1X = 1000 Mbps 01 = 100 Mbps 00 = 10 Mbps	0x0
12	AN_EN	RW	Auto-negotiation enable 0 = Disable 1 = Enable AN	1
11	PWRDN	RW	Power down GE SerDes 0 = Normal operation 1 = Power-down the SerDes PHY	0
10	RESERVED	RO	Reserved write 0, ignore read	0
9	RESTART_AN	RW	Restart Auto-negotiation process 0 = Normal operation 1 = Restart AN	0
8	FDX	RW	Duplex mode 0 = Half duplex 1 = Full duplex	0
7	COL_TEST_EN	RW	Collision test enable 0 = Normal operation 1 = Collision test	0
6	SPD[1]	RW	Bit[1] of manual Speed[1:0] in SGMII mode only. This field is ignored in 1000Base-X operation. 1X = 1000 Mbps 01 = 100 Mbps 00 = 10 Mbps	1
5:0	RESERVED	RO	Reserved write 0, ignore read.	0x0

MII Status

Register Description: MII Status Register.

Register Offset: 0x1 at Block 0

Table 123: MII Status

Bits	*Name*	*RW*	*Description*	*Default*
15	100BASE_T4	RO	0 = Incapable. 1 = 100Base-T4 capable.	0
14	100BASEX_FDX	RO	0 = Incapable. 1 = 100BASE-X full-duplex capable.	0
13	100BASEX_HDX	RO	0 = Incapable. 1 = 100BASE-X half-duplex capable.	0
12	10BASET_FDX	RO	0 = Incapable. 1 = 10BASE-T full-duplex capable.	0
11	10BASET_HDX	RO	0 = Incapable. 1 = 10BASE-T half-duplex capable.	0
10	10BASET2_FDX	RO	0 = Incapable. 1 = 10BASE-T2 full-duplex capable.	0
9	10BASET2_HDX	RO	0 = Incapable. 1 = 10BASE-T2 half-duplex capable.	0
8	EXT_STATUS	RO	0 = No extended status. 1 = Extended status in register 0x0F.	1
7	RESERVED	RO	Reserved write 0, ignore read.	0
6	MF_PREAMBLE_SUPPRESSION	RO	0 = PHY does not accept management frames with preamble suppressed. 1 = PHY accepts management frames with preamble suppressed.	1
5	AN_COMPLETE	RO	Auto-negotiation complete. 0 = Incomplete. 1 = AN complete.	0
4	RF	RO	Remote fault. 0 = No fault detected. 1 = Remote fault detected.	0
3	AN_ABILITY	RO	Auto-negotiation ability. 0 = Incapable of AN. 1 = AN capable.	1
2	LINK_STATUS	RO	Link status. 0 = Link fail. 1 = Good link.	0

Table 123: MII Status (Cont.)

Bits	Name	RW	Description	Default
1	JABBER_DETECT	RO	Jabber detect 0 = Not detected 1 = Jabber detected	0
0	EXT_CAPABILITY	RO	Extended capability 0 = Supports basic register set only 1 = Extended register capabilities supported	1

Table 124: 02h: PHY_Identifier_MSB_Register

Bits	Name	RW	Description	Default
15:0	OUI_MSB	RO	Bits 3:18 of organizationally unique identifier	0143h

Table 125: 03h: PHY_Identifier_LSB_Register

Bits	Name	RW	Description	Default
15:10	OUI_MSB	RO	Bits 3:18 of organizationally unique identifier	101111
9:4	MODEL	RO	Device Model number	111111
3:0	REVISION	RO	Device revision number	0000

AUTONEGADV

Register Description: Auto-negotiation capability register.

Register Offset: 0x4 at Block 0

Table 126: AUTONEGADV

Bits	Name	RW	Description	Default
15	NEXT_PG	RO	Next page	0
14	RESERVED	RO	Reserved write 0, ignore read	0
13:12	RF	RW	Remote fault 2'b00 = No fault 2'b01 = Link failure 2'b10 = Offline 2'b11 = Auto-negotiation error	0x0
11:9	RESERVED	RO	Reserved write 0, ignore read	0x0

Table 126: AUTONEGADV (Cont.)

Bits	Name	RW	Description	Default
8:7	PAUSE	RW	Pause 2'b00 = No pause 2'b01 = Asymmetric pause 2'b10 = Asymmetric pause towards link partner 2'b11 = Both symmetric and asymmetric pause, towards local device	0x0
6	HDX	RW	Half-duplex 0 = Do not advertise half-duplex 1 = Advertise half-duplex	1
5	FDX	RW	Full-duplex 0 = Do not advertise full-duplex 1 = Advertise full-duplex	1
4:0	RESERVED	RO	Reserved write 0, ignore read.	0x0

AUTONEG Link Partner Ability

Register Description: Auto-negotiation link partner ability register.

Register Offset: 0x5 at Block 0.

Table 127: AUTONEG LINK PARTNER ABILITY

Bits	Name	RW	Description	Default
15	NEXT_PG	RO	Next page	0
14	ACK	RO	0 = Link partner has not received link code word 1 = Link partner has received link code word	0
13:12	RF	RO	Remote fault 2'b00 = No fault 2'b01 = Link failure 2'b10 = Offline 2'b11 = Auto-negotiation error	0x0
11:9	RESERVED	RO	Reserved write 0, ignore read	0x0
8:7	PAUSE	RO	Pause 2'b00 = No pause 2'b01 = Symmetric pause 2'b10 = Asymmetric pause towards link partner 2'b11 = Both symmetric and asymmetric pause, towards local device	0x0
6	HDX	RO	Half-duplex 0 = Do not advertise half-duplex 1 = Advertise half-duplex	0

Table 127: AUTONEG LINK PARTNER ABILITY (Cont.)

Bits	Name	RW	Description	Default
5	FDX	RO	Full-duplex 0 = Do not advertise full-duplex 1 = Advertise full-duplex	0
4:1	RESERVED	RO	Reserved write 0, ignore read.	0x0
0	SGMII	RO	SGMII mode 0 = Fiber mode 1 = SGMII mode	0

When the link partner is in SGMII mode (bit-0 = 1), then:
Bit[15] = Link
Bit[14] = Acknowledge
Bit[13] = Duplex
Bit[11:10] = Speed
The other bits are reserved and should be zero.

AUTONEGEXPANSION

Register Description: Auto-negotiation expansion register.

Register Offset: 0x6 at Block 0

Table 128: AUTONEGEXPANSION

Bits	Name	RW	Description	Default
15:3	RESERVED	RO	Reserved	0x000
2	NP_ABILITY	RO	Next page ability 0 = Local device is not next page capable. 1 = Local device is next page capable.	0
1	PG_REC	RO	Page received 0 = New link code word has not been received. 1 = Received new link code word.	0
0	RESERVED	RO	Reserved write 0, ignore read.	0

EXTENDEDSTATUS

Register Description: Extended status register.

Register Offset: 0xF at Block 0

Table 129: EXTENDEDSTATUS

Bits	Name	RW	Description	Default
15	1000BASEX_FDX	RO	0 = 1000Base-X full duplex not capable. 1 = 1000Base-X full duplex capable.	1
14	1000BASEX_HDX	RO	0 = 1000Base-X half duplex not capable. 1 = 1000Base-X half duplex capable.	1
13	1000BASET_FDX	RO	0 = 1000Base-T full duplex not capable. 1 = 1000Base-T full duplex capable.	0
12	1000BASET_HDX	RO	0 = 1000Base-T half duplex not capable. 1 = 1000Base-T half duplex capable.	0
11:0	RESERVED	RO	Reserved write 0, ignore read.	0x000

1000XCONTROL1

Register Description: 1000X Control1 Register.

Register Offset: 0x10 at Block 0

Table 130: 1000XCONTROL1

Bits	Name	RW	Description	Default
15	RESERVED	RO	Reserved write 0, ignore read	0
14	DIS_SD_FILTER	RW	0 = Filter signal detect from pin before using for synchronization. 1 = Disable filter for signal detect.	0
13	MSTR_MDIO_PHY_SEL	RW	0 = Normal operation. 1 = All MDIO write accesses to PHY address "00000" will write this PHY in addition to its own PHY address.	0
12	SERDES_TX_AMPL_OVERRIDE	RW	0 = If SGMII mode, use analog txCntrl Reg. (reg. 3*10h and reg. 3*11h), if fiber mode, use analog txAmp Reg. (reg. 3*12h). 1 = Use analog txCntrl Reg. (reg. 3*10h and reg. 3*11h).	0
11	SEL_RX_PKTS_FOR_CNTS	RW	0 = Select CRC errors for 0*17h counter. 1 = Select received packets for 0*17h counter.	0
10	REMOTE_LOOPBACK	RW	0 = Normal operation. 1 = Enable remote loopback (operates in 10/100/1000) speed.	0
9	ZERO_COMMA_DET_PHASE	RW	0 = Normal operation. 1 = Force comma detector phase to zero.	0
8	COMMA_DET_EN	RW	0 = Disable comma detection. 1 = Enable comma detection.	1
7	CRC_CHECKER_DIS	RW	0 = Enable CRC checker. 1 = Disable CRC checker by gating the clock to save power.	1

Table 130: 1000XCONTROL1 (Cont.)

Bits	Name	RW	Description	Default
6	DISABLE_PLL_PWRDWN	RW	0 = PLL will be powered down when register 0.11 is set. 1 = PLL will never be powered down. (use this when the mac/switch uses the pll_clk125 output).	0
5	SGMII_MSTR_MODE	RW	0 = Normal operation. 1 = SGMII mode operates in "PHY mode". If auto-neg is enabled, then the local device will send out the following auto-neg code word: [15] = 1 [14] = ACK [13] = 0 [12] = Register 0.8 [11] = Register 0.6 [10] = Register 0.13 [9:0] = "0000000001" To disable the link, set register 0.11 = 1. To enable the link, set register 0.11 = 0.	1
4	AUTODET_EN	RW	0 = Disable auto detection (fiber or SGMII mode is set according to bit 0 of this register.) 1 = Enable auto-detection (fiber and SGMII mode will switch each time a auto-negotiation page is received with the wrong selector field in bit 0.)	0
3	INVERT_SIG_DET	RW	0 = Use signal detect from pin. 1 = Invert signal detect from pin.	1
2	SIGNAL_DETECT_EN	RW	0 = Ignore signal detect from pin. 1 = Signal detect from pin must be set in order to achieve synchronization. In SGMII, the signal detect is always ignored regardless of the setting of this bit.	1
1	TBI_INTERFACE	RW	0 = GMII interface 1 = Ten bit interface.	1
0	FIBER_MODE_1000X	RW	0 = SGMII mode 1 = Fiber mode (1000X)	1

1000XCONTROL2

Register Description: 1000X Control2 Register.

Register Offset: 0x11 at Block 0

Table 131: 1000XCONTROL2

Bits	Name	RW	Description	Default
15	DIS_EXTEND_FDX	RW	0 = Normal operation 1 = In full duplex mode, disable carrier extension in pcs receive when bit[7] of this register is set and disable TRRR generation in pcs transmit when bit[8] of this register is set.	0
14	CLR_BER_CNTR	RW	0 = Normal operation 1 = Clear bit-error-rate counter (register 0*17h bits[15:8])	0
13	TX_IDLE_JAM_SEQ_TEST	RW	Register 0*1dh bits[9:0] will override k28.5 for stage 5 (17Ch). Register 0*1eh bits[9:0] will override D16.2 for stage 6 (289h). 0 = Normal operation. 1 = Enable 16-stage 10-bit idle transmit test sequence to SerDes transmitter.	0
12	TX_PKT_SEQ_TEST	RW	Stage 1-4, 13-16 = idle. Stage 5-12 = data packet. 0 = Normal operation. 1 = Enable 16-stage 10-bit idle transmit test sequence to SerDes transmitter.	0
11	TEST_CNTR	RW	0 = Normal operation. 1 = Increment bits[7:0] of register 0*17h counter for each clock cycle.	0
10	BYPASS_PCS_TX	RW	0 = Normal operation 1 = Bypass pcs transmit operation	0
9	BYPASS_PCS_RX	RW	0 = Normal operation 1 = Bypass pcs receive operation	0
8	DISABLE_TRRR_GEN	RW	0 = Normal operation 1 = Disable TRRR generation in pcs transmit	0
7	DISABLE_CARRIER_EXTEND	RW	0 = Normal operation 1 = Disable carrier extension in pcs receive	0
6	AUTONEG_FAST_TIMERS	RW	0 = Normal operation 1 = Speed up timers during auto-negotiation for testing	0
5	FORCE_XMIT_DATA_ON_TXSIDE	RW	0 = Normal operation 1 = Allow packets to be transmitted regardless of the condition of the link or synchronization	0
4	DIS_REMOTE_FAULT_SENSING	RW	0 = Automatically detect remote faults and send remote fault status to link partner via auto-negotiation when fiber mode is selected. (SGMII does not support remote faults) 1 = Disable automatic sensing of remote faults, such as auto-negotiation error	0

Table 131: 1000XCONTROL2 (Cont.)

Bits	Name	RW	Description	Default
3	ENABLE_AUTONEG_ERR_TIMER	RW	0 = Normal operation 1 = Enable auto-negotiation error timer. Error occurs when timer expires in ability-detect, ack-detect, or idle-detect. When the error occurs, config words of all zeros are sent until an ability match occurs, then the autoneg-enable state is entered.	0
2	FILTER_FORCE_LINK	RW	0 = Normal operation 1 = Sync-status must be set for a solid 10ms before a valid link will be established when auto-negotiation is disabled. (This is useful in fiber applications where the user does not have the signal detect pin connected to the fiber module and auto-negotiation is turned off.)	1
1	DISABLE_FALSE_LINK	RW	0 = Normal operation 1 = Do not allow link to be established when auto-negotiation is disabled and receiving auto-negotiation code words. The link will only be established in this case after idles are received. (This bit does not need to be set, if bit 0 below is set.)	1
0	ENABLE_PARALLEL_DETECTION	RW	0 = Disable parallel detection 1 = Enable parallel detection. (This will turn auto-negotiation on and off as needed to properly link up with the link partner. The idles and auto-negotiation code words received from the link partner are used to make this decision)	1

1000XCONTROL3

Register Description: 1000X Control3 Register.

Register Offset: 0x12 at Block 0

Table 132: 1000XCONTROL3

Bits	Name	RW	Description	Default
15	DISABLE_PKT_ALIGNMENT	RW	0 = Normal operation. 1 = Disable packet misalignment by carrier extend and removing preamble.	0
14	RXFIFO_GMII_RST	RW	0 = Normal operation. 1 = Reset receive FIFO and data_out_1000. FIFO will remain in reset until this bit is cleared with a software write.	0
13	DISABLE_TX_CRS	RW	0 = Normal operation. 1 = Disable generating crs from transmitting in half-duplex mode. Only receiving will generate crs.	1
12	INVERT_EXT_PHY_CRS	RW	0 = Use "receive crs from PHY" pin. 1 = Invert "receive crs from PHY" pin	1

Table 132: 1000XCONTROL3 (Cont.)

Bits	Name	RW	Description	Default
11	EXT_PHY_CRS_MODE	RW	0 = Normal operation. 1 = Use external pin for the PHYs "receive only" crs output. (Useful in SGMII 10/100 half-duplex applications in order to reduce the collision domain latency. Requires a PHY, which generates a "receive only" crs output to a pin.)	1
10	JAM_FALSE_CARRIER_MODE	RW	0 = Normal operation. 1 = Change false carriers received into packets with preamble only. (Not necessary if MAC uses crs to determine collision)	0
9	BLOCK_TXEN_MODE	RW	0 = Normal operation. 1 = Block txen when necessary to guarantee an ipg of at least 6.5 bytes in 10/100 mode, 7 bytes in 1000 mode.	0
8	FORCE_TXFIFO_ON	RW	0 = Normal operation. 1 = Force transmit FIFO to free-run in gigabit mode (Requires clk_in and pll_clk125 to be frequency locked.)	0
7	BYPASS_TXFIFO1000	RW	0 = Normal operation. 1 = Bypass transmit FIFO in gigabit mode. (Useful for fiber or gigabit only applications where the MAC is using the pll_clk125 as the clk_in port. User must meet timing to the pll_clk125 domain)	0
6	FREQ_LOCK_ELASTICITY_TX	RW	0 = Normal operation. 1 = Minimum FIFO latency to properly handle a clock which is frequency locked, but out of phase. (overrides bits [2:1] of this register). ***Note:*** pll_clk125 and clk_in must be using the same crystal.	1
5	FREQ_LOCK_ELASTICITY_RX	RW	0 = Normal operation 1 = Minimum FIFO latency to properly handle a clock which is frequency locked, but out of phase. (Not necessary if MAC uses crs to determine collision; overrides bits [2:1] of this register). ***Note:*** MAC and PHY must be using the same crystal for this mode to be enabled.	0
4	EARLY_PREAMBLE_RX	RW	0 = Normal operation 1 = Send extra bytes of preamble to avoid FIFO latency. (Not necessary if MAC uses crs to determine collision)	0
3	EARLY_PREAMBLE_TX	RW	0 = Normal operation 1 = Send extra bytes of preamble to avoid FIFO latency. (Used in half-duplex applications to reduce collision domain latency. MAC must send 5 bytes of preamble or less to avoid non-compliant behavior.)	0

Table 132: 1000XCONTROL3 (Cont.)

Bits	Name	RW	Description	Default
2:1	FIFO_ELASTICITY_TX_RX	RW	00 = Supports packets up to 5 KB 01 = Supports packets up to 10 KB 1X = Supports packets up to 13.5 KB	0
0	TX_FIFO_RST	RW	0 = Normal operation 1 = Reset transmit FIFO. FIFO will remain in reset until this bit is cleared with a software write.	0

1000XSTATUS1

Register Description: 1000X Status1 Register.

Register Offset: 0x14 at Block 0

Table 133: 1000XSTATUS1

Bits	Name	RW	Description	Default
15	TXFIFO_ERR_DETECTED	RO	1 = Transmit FIFO error detected since last read. 0 = No transmit FIFO error detected since last read.	0
14	RXFIFO_ERR_DETECTED	RO	1 = Receive FIFO error detected since last read. 0 = No receive FIFO error detected since last read.	0
13	FALSE_CARRIER_DETECTED	RO	1 = False carrier detected since last read. 0 = No false carrier detected since last read.	0
12	CRC_ERR_DETECTED	RO	1 = CRC error detected since last read. 0 = No CRC error detected since last read or detection is disabled via register 0*10h bit [7].	0
11	TX_ERR_DETECTED	RO	1 = Transmit error code detected since last read (rx_data_error state in pcs receive fsm). 0 = No transmit error code detected since last read.	0
10	RX_ERR_DETECTED	RO	1 = Receive error since last read (early_end state in pcs receive fsm) 0 = No receive error since last read	0
9	CARRIER_EXT_ERR_DETECTED	RO	1 = Carrier extend error since last read (extend_err in pcs receive fsm) 0 = No carrier extend error since last read	0
8	EARLY_END_EXT_DETECTED	RO	1 = Early end extension since last read (early_end_ext in pcs receive fsm) 0 = No early end extension since last read	0
7	LINK_STATUS_CHG	RO	1 = Link status has changed since last read 0 = Link status has not changed since last read	0
6	PAUSE_RESOLUTION_RXSIDE	RO	1 = Enable pause receive 0 = Disable pause receive	0

Table 133: 1000XSTATUS1 (Cont.)

Bits	Name	RW	Description	Default
5	PAUSE_RESOLUTION_TXSIDE	RO	1 = Enable pause transmit 0 = Disable pause transmit	0
4:3	SPEED_STATUS	RO	1X = Gigabit 01 = 100 mbps 00 = 10 mbps	0
2	DUPLEX_STATUS	RO	1 = Full-duplex 0 = Half-duplex ***Note:*** When the ten bit interface is selected with fiber mode (1000-X), then half-duplex will always be reported.	0
1	LINK_STATUS	RO	1 = Link is up. 0 = Link is down. ***Note:*** When the ten bit interface is selected with fiber mode (1000-X), then link will always be down	0
0	SGMII_MODE	RO	1 = SGMII mode 0 = Fiber mode (1000-X)	0

1000XSTATUS2

Register Description: 1000X Status2 Register.

Register Offset: 0x15 at Block 0

Table 134: 1000XSTATUS2

Bits	Name	RW	Description	Default
15	SGMII_MODE_CHG	RO	1 = SGMII mode has changed since last read (SGMII mode enabled or disabled). ***Note:*** This bit is useful when the auto-detection is enabled in register 0*10h bit [4]. 0 = SGMII mode has not changed since last read (fixed in SGMII or fiber mode).	0
14	CONSISTENCY_MISMATCH	RO	1 = Consistency mismatch detected since last read. 0 = Consistency mismatch has not been detected since last read.	0
13	AUTONEG_RES_ERR	RO	1 = Auto-negotiation HCD error detected since last read (HCD is none in fiber mode). 0 = Auto-negotiation HCD error has not been detected since last read.	0
12	SGMII_SELECTOR_MISMATCH	RO	1 = SGMII selector mismatch detected since last read (auto-negotiation page received from link partner with bit [0] = 0 while local device is in SGMII mode). 0 = SGMII selector mismatch not detected since last read.	0

Table 134: 1000XSTATUS2 (Cont.)

Bits	Name	RW	Description	Default
11	SYNC_STATUS_FAIL	RO	1 = Sync_status has failed since last read (synchronization has been lost). 0 = Sync_status has not failed since last read.	0
10	SYNC_STATUS_OK	RO	1 = Sync_status ok detected since last read (synchronization has been achieved). 0 = Sync_status ok has not been detected since last read.	0
9:7	RESERVED	RO	Reserved.	0
6	LINK_DOWN_SYNC_LOSS	RO	1 = 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
5:1	RESERVED	RO	Reserved.	0
0	ANEG_ENABLE_STATE	RO	1 = An_enable state in auto-negotiation fsm entered since last read. 0 = An_enable state has not been entered since last read.	0

1000XSTATUS3

Register Description: 1000X Status3 Register.

Register Offset: 0x16 at Block 0

Table 135: 1000XSTATUS3

Bits	Name	RW	Description	Default
15:11	RESERVED	RO	Reserved	0x00
10	LATCH_LINKDOWN	RO	1 = Link has been down since register 0*13h bit [9] has been written a '1' 0 = Link has not been down since register 0*13h bit [9] has been written a '1'	0
9	SD_FILTER	RO	1 = Output of signal detect filter is set 0 = Output of signal detect is not set ***Note:*** This signal is used for the PCS synchronization. When register 0*10h bit [2] is 0, then the output of the filter will be forced high. This status signal is still valid when register 0*10h bit [14] is 1. Noise pulses less than 16 ns wide are still removed when even the filter is disabled.	0
8	SD_MUX	RO	1 = Output of signal detect filter is set 0 = Output of signal detect is not set ***Note:*** This is the only SD status bit that will be valid when the SerDes is powered down from register 0.11. This status signal is the "signal detect" input port when register 0*10h bit [3] is 0, otherwise it is the "signal detect" input port inverted.	0

Table 135: 1000XSTATUS3 (Cont.)

Bits	Name	RW	Description	Default
7	SD_FILTER_CHG	RO	1 = Signal detect has changed since last read 0 = Signal detect has not changed since last read ***Note:*** The signal detect change is based on a change in bit [9] of this register	0
6	SIGNAL_DETECT	RO	Signal detect directly from pin	0
5	ANA_SIGNAL_DET	RO	Analog signal detect status bit. This status signal is the analog signal detect status if register 0*13h bit [0] is set, otherwise it is the value based on register 0*13h bit [1].	0
4	ANA_SIGDET_CHG	RO	1 = Analog signal detect has changed since last read 0 = Analog signal detect has not changed since last read ***Note:*** The analog signal detect change is based on a change in bit [5] of this register.	0
3:0	RESERVED	RO	Reserved	0

FXCONTROL1

Register Description: 100FX enabling control register

Register Offset: 0x10 at Block 2

Table 136: FXCONTROL1

Bits	Name	RW	Description	Default
15	RESERVED	RW	Reserved	0
14	FIBER_AUTOPWRDWN_WAKEUP	RW	1 = Wake up for 250ms before powering down 0 = Wake up for 42ms before powering down	0
13	FIBER_AUTOPWRDWN_SLEEP	RW	1 = Power-down for 3 seconds before waking up 0 = Power-down for 5 seconds before waking up	0
12	FIBER_AUTOPWRDWN_ENABLE	RW	1 = Power-down fiber when signal detect is inactive (wake up for 42 ms every 5 seconds to transmit code words; see register 2*10h[13:12] for different time options). 0 = Normal operation.	0
11	FIBER_AUTOPWRDWN_DISABLE	RW	1 = Disable 1000-X power down from fiber auto-power down (register 0[11] powerdown not affected). 0 = Normal operation	0
10	FX100_AUTODET_TIMER_SEL	RW	1 = 125-166 ms (do not use if fiber auto-power down is enabled; register 2*10h[12]). 0 = 31-42 ms	0
9:6	FX100_RXDATA_SEL	RW	Selects the sample bit out of 10 bits for FX100 RX data	0x9

Table 136: FXCONTROL1 (Cont.)

Bits	Name	RW	Description	Default
5	FX100_DISABLE_RX_QUAL	RW	1 = Always use sample bit without filtering 0 = Normal operation	0
4	FX100_FORCE_RX_QUAL	RW	1 = Always compare 2 surrounding bits with sample to filter noise 0 = Normal operation	0
3	FX100_FAREND_FAULT_EN	RW	1 = Enable far-end fault 0 = Disable far-end fault	1
2	FX100_AUTODET_EN	RW	1 = Auto-detect between 100FX mode and 1000-X mode 0 = Disable auto-detection	0
1	FX100_FULL_DUPLEX	RW	1 = 100-FX SerDes full-duplex 0 = 100-FX SerDes half-duplex	1
0	FX100_ENABLE	RW	1 = Select 100-FX mode 0 = Select 1000-X mode	0

FXCONTROL2

Register Description: 100FX extended packet size enabled

Register Offset: 0x11 at Block 2

Table 137: FXCONTROL2

Bits	Name	RW	Description	Default
15:1	RESERVED	RW	Reserved	0x000
0	EXTEND_PKT_SIZE	RW	1 = Allow reception of extended length packets 0 = Allow normal length Ethernet packets only	0

FXCONTROL3

Register Description: 100FX control register

Register Offset: 0x12 at Block 2

Table 138: FXCONTROL3

Bits	Name	RW	Description	Default
15:7	RESERVED	RW	Reserved	0x000
6	FX100_BYPASS_NRZ	RW	1 = Bypass NRZ encoder in 100FX mode 0 = Normal operation	0
5	FX100_BYPASS_ENCODER	RW	1 = Bypass 4B5B encoder in 100FX mode 0 = Normal operation	0

Table 138: FXCONTROL3 (Cont.)

Bits	Name	RW	Description	Default
4	FX100_BYPASS_ALIGNMENT	RW	1 = Bypass 5B code group alignment in 100FX mode 0 = Normal operation	0
3	FX100_FORCE_LINK	RW	1 = Force link in 100FX mode 0 = Normal operation	0
2	FX100_FORCE_LOCK	RW	1 = Force lock in 100FX mode 0 = Normal operation	0
1	FX100_FAST_UNLOCK_TIMER	RW	1 = Speed up unlock timer in 100FX mode 0 = Normal operation	0
0	FX100_FAST_TIMER	RW	1 = Speed up timer to acquire lock and link (test vectors and simulation) in 100FX mode 0 = Normal operation	0

FXSTATUS1

Register Description: 100FX link status register

Register Offset: 0x13 at Block 2

Table 139: FXSTATUS1

Bits	Name	RW	Description	Default
15:10	RESERVED	RW	Reserved	0x00
9	FX100_LINK_STATUS_CHG	RO/LH	1 = 100-FX mode link status change since last read 0 = 100-FX mode link status has not changed since last read	0
8	FX100_BAD_ESD_DETECTED	RO/LH	1 = 100-FX mode bad ESD error detected since last read (premature end) 0 = No 100-FX mode bad ESD error detected since last read	0
7	FX100_FALSE_CARRIER_DETECTED	RO/LH	1 = 100-FX mode false carrier detected since last read 0 = No 100-FX mode false carrier detected since last read	0
6	FX100_TX_ERR_DETECTED	RO/LH	1 = 100-FX mode received packet with txer code detected since last read 0 =No 100-FX mode received packet with txer code detected since last read	0
5	FX100_RX_ERR_DETECTED	RO/LH	1 = 100-FX mode receive coding error detected since last read 0 = No 100-FX mode receive coding error detected since last read	0

Table 139: FXSTATUS1 (Cont.)

Bits	Name	RW	Description	Default
4	FX100_LOCK_TIMER_EXPIRED	RO/LH	1 = Unable to lock within 730us since last read 0 = Condition not detected since last read	0
3	FX100_LOST_LOCK	RO/LH	1 = Lost lock since last read 0 = Lock has not been lost since last read	0
2	FX100_FAULTING	RO/LH	1 = Far-end fault detected since last read 0 = No far-end fault detected since last read	0
1	FX100_LOCKED	RO	1 = Enough idles are properly detected to lock 0 = Not locked	0
0	FX100_LINK	RO	1 = 100-FX mode link is up 0 = 100-FX mode link is down	0

ANALOG_TX1

Register Description: Analog transmitter output for the SerDes when it is in SGMII mode.

Register Offset: 0x10 at Block 3

Table 140: ANALOG_TX1

Bits	Name	RW	Description	Default
15:12	DRIVER_CURRENT	RW	Setting the output amplitude of the SerDes ranging from 700 mV to 1280 mV.	0x7
11	RESERVED	RW	Reserved	0
10:7	PREEMPH_COEF	RW	Setting the pre-emphasis level for the SerDes driver, ranging from 0 to 60%.	0
6	RX_CLKP	RW	Reserved for factory use only. 1 = Select RX clock 0 = Do not select RX clock	0
5	REG_EDGE_SEL	RW	Reserved for factory use only. 1 = Capture on rising edge 0 = Capture on falling edge	0
4	BOOST_MODE	RW	Reserved for factory use only 1 = Enable boost output current for preamp driver 0 = Normal mode	0
3	DRIVER_IDLE	RW	1 = Enable transmit driver idle 0 = Disable transmit driver idle	0

Table 140: ANALOG_TX1 (Cont.)

Bits	Name	RW	Description	Default
2	LOOPBACK	RW	1 = Enable remote loopback. The far end sends the data to the device and the data is looped back to the wire at the analog block prior of reaching to the de-serializer. The data will not reach the device MAC block. In order to use the remote loopback, both loopback bits must be set in the ANALOG_TX and ANALOG_RX registers. 0 = Normal operation	0
1	RESET	RW	1 = SerDes is in reset 0 = SerDes is not in reset	0
0	IDDQ	RW	1 = Power down the driver 0 = Normal operation	0

ANALOG_TX2

Register Description: Analog transmitter2 output for the SerDes when it is in SGMII mode.

Register Offset: 0x11 at Block 3

Table 141: ANALOG_TX2

Bits	Name	RW	Description	Default
15:4	RESERVED	RW	Reserved	0x000
3:0	PREDRIVER_CURRENT	RW	This is to control the output driving amplitude. This is set in conjunction with the DRIVER_CURRENT.	0x7

ANALOG_TXAMP

Register Description: Analog transmitter output for the SerDes when it is in FX (Fiber) mode. This register is not used when device is in SGMII mode.

Register Offset: 0x12 at Block 3

Table 142: ANALOG_TXAMP

Bits	Name	RW	Description	Default
15	DRIVER_FULL_RANGE	RW	Reserved for factory use only. 1 = Enable TX driver full output range 0 = Disable TX driver full output range	0
14	PREDRIVER_SWING_BOOST	RW	Reserved for factory use only. 1 = Enable final stage predriver swing boost 0 = Disable final stage predriver swing boost	0
13:10	RESERVED	RW	Reserved.	0x0

Table 142: ANALOG_TXAMP (Cont.)

Bits	Name	RW	Description	Default
9	BMODE	RW	Reserved for factory use only. 1 = Enable transmit boost mode 0 = Normal operation	0
8:5	PREDRIVER_CURRENT	RW	This is to control the output driving amplitude. This is set in conjunction with the DRIVER_CURRENT.	0x7
4:1	DRIVER_CURRENT	RW	Setting the output amplitude of the SerDes ranging from 700 mV to 1280 mV.	0x7
0	RESERVED	RW	Reserved.	0

ANALOG_RX1

Register Description: Controlling the SerDes receiver.

Register Offset: 0x13 at Block 3

Table 143: ANALOG_RX1

Bits	*Name*	*RW*	*Description*	*Default*
15:13	RESERVED	RW	Reserved	0x00
12:10	SD_THRESHOLD	RW	Controls when signal detect should be asserted, based on energy level. 000 = 10 mV 001 = 20 mV 011 = 30 mV 010 = 40 mV 110 = 50 mV 111 = 60 mV 101 = 70 mV 100 = 80 mV	0x2
9	SIGDET_EN	RW	1 = Enable signal detect 0 = Disable signal detect	0
8	LOOPBACK	RW	Set to enable remote loopback. This must be set in conjunction with the loopback bit under ANALOG_TX1 register. 1 = Enable loopback 0 = Normal operation	0
7	REG_EDGE_SELECT	RW	Reserved for factory use only. 1 = Use rising edge of rx_wclk 0 = Use falling edge of rx_wclk	0
6:1	RESERVED	RW	Reserved	0x00
0	IDDQ	RW	Set to power down analog receiver block 1 = Power down RX 0 = Normal operation	0

ANALOG_RX2

Register Description: 100FX Test Mode

Register Offset: 0x14 at Block 3

Table 144: ANALOG_RX2

Bits	Name	RW	Description	Default
15:4	RESERVED	RW	Reserved	0x000
3	100FX_ENABLE	RW	Set to enable SerDes in 100-FX mode. 1 = FX100 mode 0 = Normal operation	0
2:0	RESERVED	RW	Reserved	0x0

ANALOG_PLL

Register Description: Controlling the GE PLL circuitry.

Register Offset: 0x18 at Block 3

Table 145: ANALOG_PLL

Bits	Name	RW	Description	Default
15:1	RESERVED	RW	Reserved	0x4040
0	PLL_POWER_DOWN	RW	1 = PLL power down 0 = Normal operation	0

GE_PRBS_CONTROL

Register Description: PRBS Control Register. This is to use in PRBS test mode only.

Register Offset: 0x18 (Block 0)

Bit Number															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												prbs_order		inv_prbs_order	prbs_en

Table 146: GE_prbs_status

Bit	Name	RW	Description	Default
15:4	Reserved	RO	Reserved	0x0
3:2	Prbs_order	RW	0 = 7th order 1 = 15th order 2 = 23rd order 3 = 31st order	0
1	Inv_prbs_order	RW	Set to invert the polynomial order	0
0	Prbs_en	RW	Set to enable PRBS testing	0

GE_PRBS_STATUS

Register Description: PRBS Status Register. This is to use in PRBS test mode only.

Register Offset: 0x19 (Block 0)

Bit Number															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			Unlock	Lock	Prbs_errors										

Table 147: GE_prbs_status

Bit	Name	RW	Description	Default
15:13	Reserved	RO	Reserved	0x0
12	Unlock	RO	In PRBS mode, this bit indicates signal is no longer available or PRBS pattern is loss.	0
11	Lock	RO	In PRBS mode, this indicates PRBS pattern is locked.	0
10:0	Prbs_errors	RO	In PRBS mode, this indicates number of errors detected.	0x0000

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 bits15:14 = 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

```
-----------------------------------------------
Clause 45 register Dev3 Reg803Eh read Procedure
-----------------------------------------------
int value;

phy_write(0xd, 0x7); //Set function field to address and device to 0x7

phy_write(0xe, 0x803e); //Write 803Eh to to select the EEE Resolution Status Register

phy_write(0xd, 0x4007); //Set function field to data and device to 0x7

value = phy_read(0xe); //read the selected Clause 45 register from 0xe



-------------------------------------------
Clause 45 register Dev3 Reg3Ch write Procedure
-------------------------------------------
phy_write(0xd, 0x7); //Set function field to address and device to 0x7

phy_write(0xe, 0x3C); //Write 3Ch to to select the EEE advertise Register

phy_write(0xd, 0x4007); //Set function field to data and device to 0x7
```

```
phy_write(0xe, 0x6); //Set bit 2:1 to advertise that EEE is supported for 1000BASE-T and 100BASE-TX
```

Table 148: Clause 45 Register Dev 3 Reg14h: EEE Capability Register

Bit	Name	RW	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

1000BASE-T EEE

(empty section)

100BASE-TX EEE

(empty section)

Clause 45 Register Dev 7 Reg3ch (60d): EEE Advertisement Register

Table 149: Clause 45 Register Dev 7 Reg3Ch: EEE Advertisement Register

Bit	Name	RW	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 150: Clause 45 Register Dev 7 Reg803Eh: EEE Resolution Status

Bit	Name	RW	Description	Default
15:3	Reserved	RW	Write as 0, ignore on read	0
2	EEE 1000BASE-T Resolution	RW	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	RW	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	RW	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 151: Clause 45 Register Dev 7 Reg803dh: EEE Control Register

Bit	Name	RW	Description	Default
15	LPI Feature Enable	RW*	1 = Enable LPI Feature 0 = Disable LPI Feature	0

Table 151: Clause 45 Register Dev 7 Reg803dh: EEE Control Register (Cont.)

Bit	Name	RW	Description	Default
14	Enables EEE capability using SGMII auto-negotiation	RW	1 = Enable EEE capability using SGMII auto-negotiation 0 = Do not enable EEE capability using SGMII auto-negotiation	0
13	Reserved	RW	Write as 0, ignore on read.	0
12	Reserved	RW	Write as 0, ignore on read.	0
11:0	Reserved	RW	Write as 0, ignore on read.	0000h

LPI Feature Enable

Setting this bit high enables LPI

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:

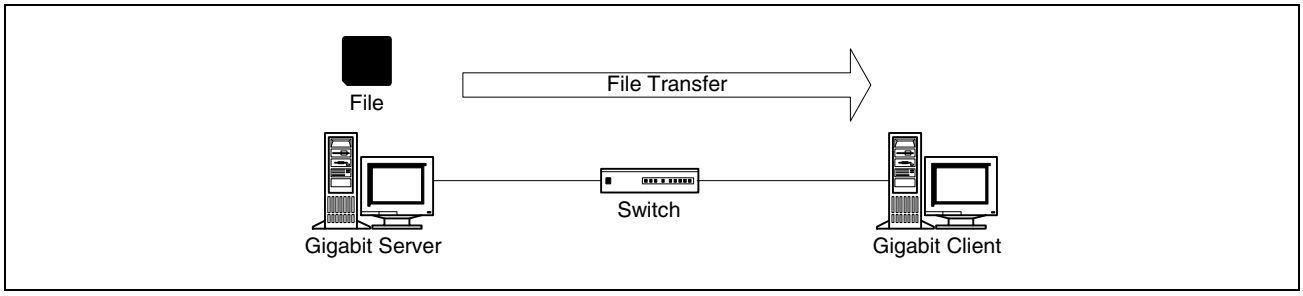
- Client
  - Full-duplex connection at Gigabit speed.
  - Implements flow control.
  - Flow control enabled.
- Switch
  - Does not drop packets.
  - Full-duplex connection to Client.
- Server
  - Gigabit connection.

– 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 60). The file size is very large and will take several minutes for a complete transfer, even at gigabit wire speed.

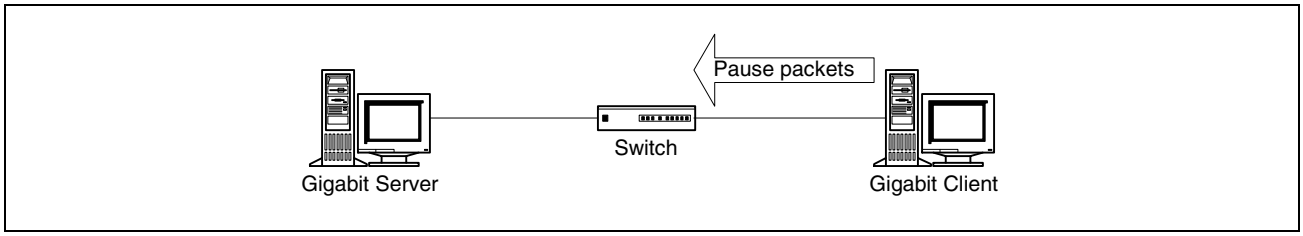
Figure 60: File Transfer Scenario: FTP Session Begins



Speed Mismatch

The Client sends pause frame(s) to the switch (see Figure 61). 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 61: File Transfer Scenario: Speed Mismatch

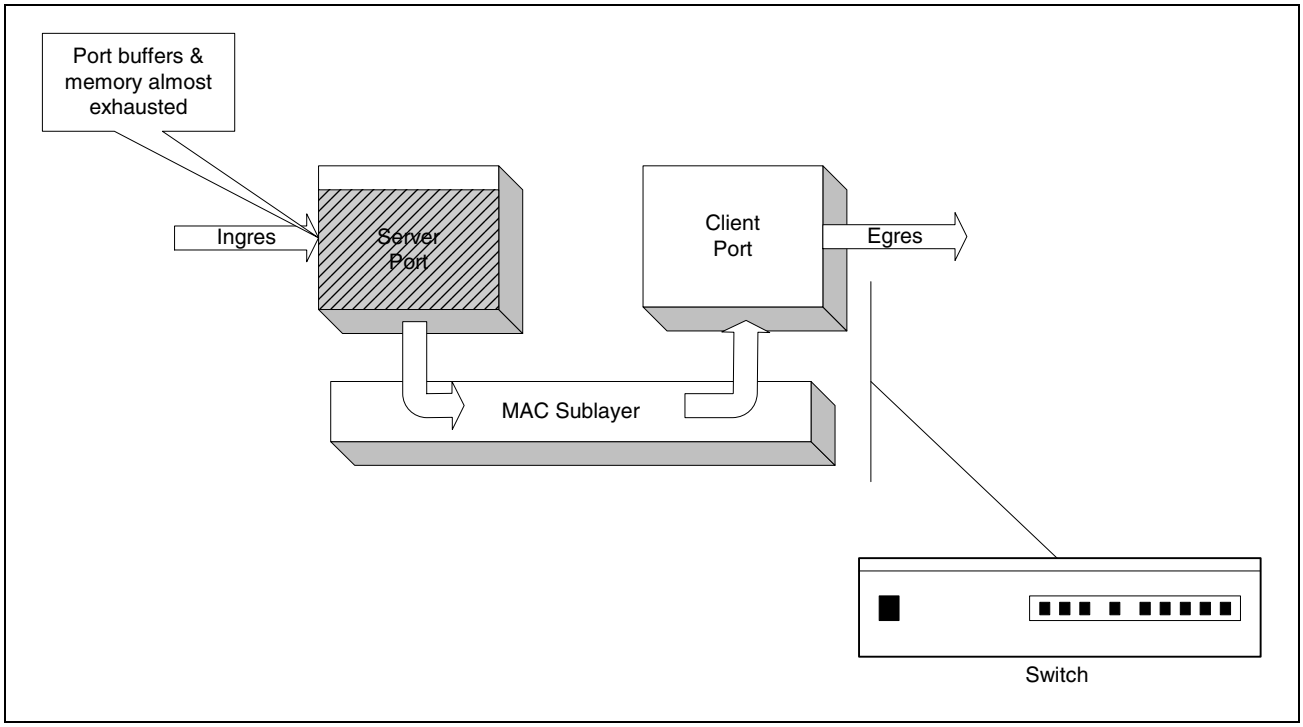


Switch Buffers Run Low

The switch must wait/inhibit transmission to the Client (see Figure 62). During the pause interval, the server is still sending packets to the switch. The switch buffers some packets, but eventually hits 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 dictates how the switch slows traffic. There are two options:

- Jamming (half-duplex)
- Pause frames (full-duplex)

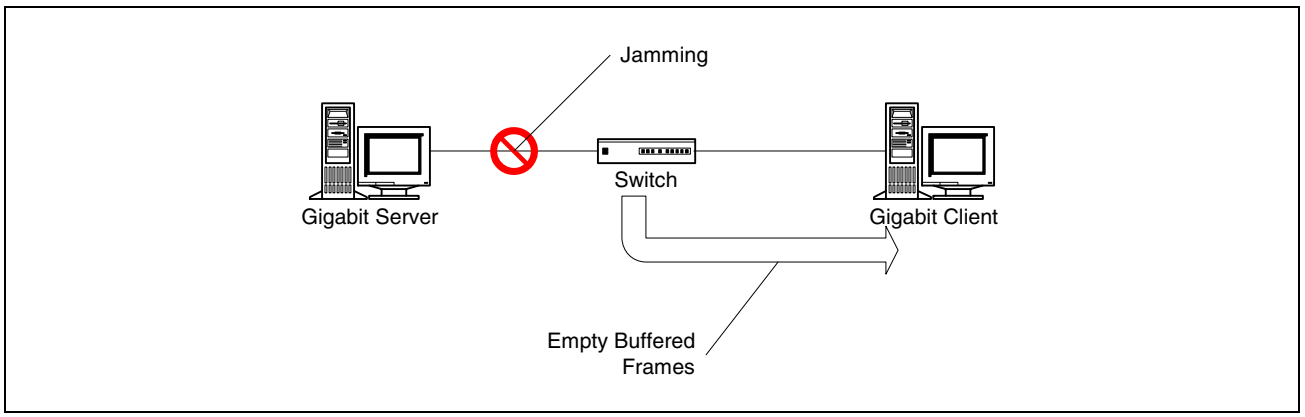
Figure 62: File Transfer Scenario: Speed Buffers Run Low



Switch Backpressure

The Switch will jam ports configured with half-duplex link to slow frame transmission (see Figure 63). In this case, the Server connection must be half-duplex, and then the switch may apply back pressure 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 back pressure 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.

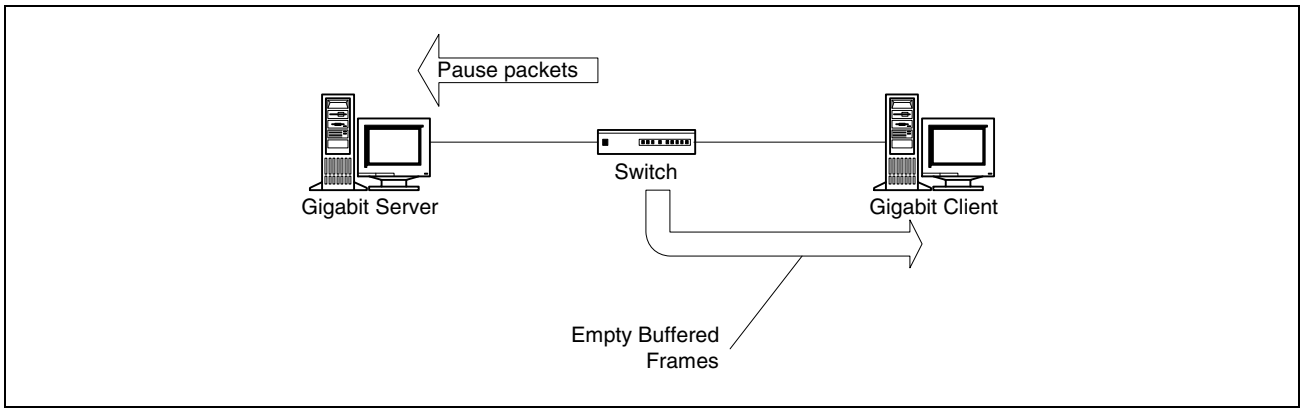
Figure 63: File Transfer Scenario: Switch Backpressure



Switch Flow Control

The Switch can only use IEEE 802.3x flow control when the link is configured for full-duplex operation (see Figure 64). 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.

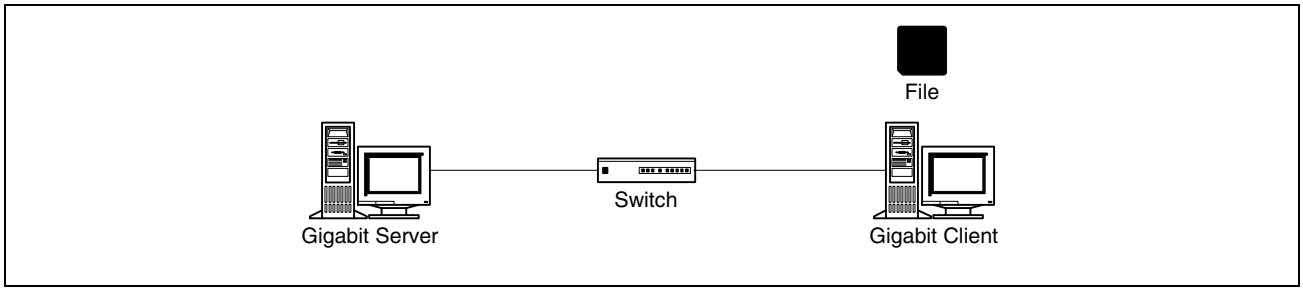
Figure 64: File Transfer Scenario: Switch Flow Control



File Transfer Complete

The Client has caught up with the transmission flow of the Server (see Figure 65). 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 65: File Transfer Scenario: File Transfer Complete

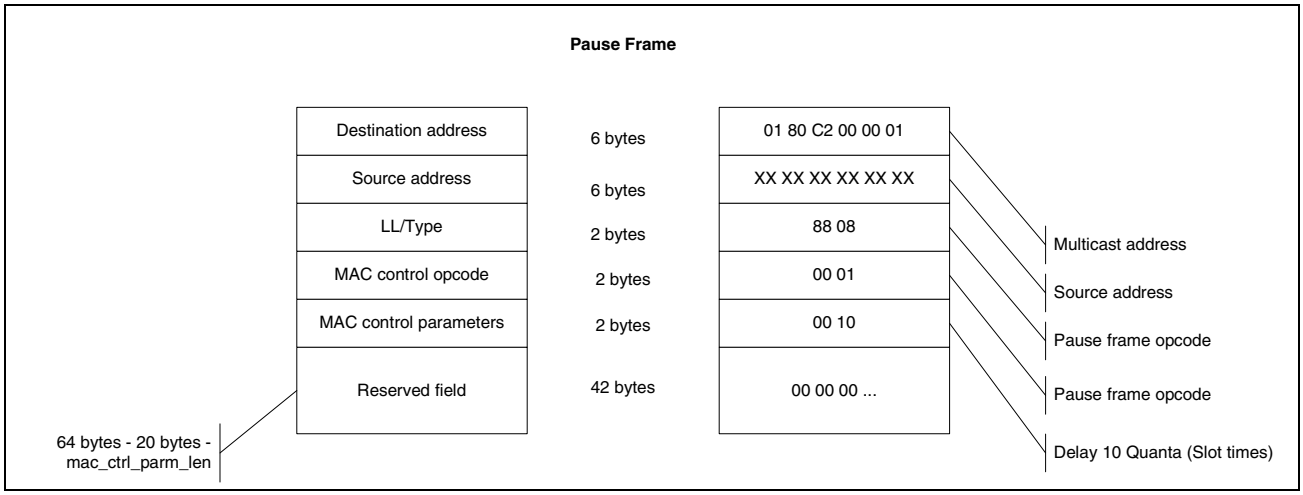


Pause Control Frame

The minimum size frame is 512 bits or 64 bytes (see Figure 66). 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 66: Pause Control Frame



Appendix B: Terminology

Table 152: Terminology

Term	*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.

Appendix C: Device Register and Memory Map

BCM5717 / BCM5718 Memory Map

Table 153: BCM5717 / BCM5718 Memory Map

Region	*Size*	*NIC CPU View*	*Host Flat View*	*Host Standard View (Offset)*	*Host UNDI View*
Unmapped	256B	0x00000000-0x000000FF	0x01000000-0x010000FF	0x00000000-0x000000FF	0x00000000-0x000000FF
Send Ring [1] RCB	16B	0x00000100-0x0000010F	0x01000100-0x0100010F	0x00000100-0x0000010F	0x00000100-0x0000010F
Send Ring [2 – 16] RCBs (Relocated or expanded in BCM5717/ BCM5718)	240B	0x00000110-0x000001FF	0x01000110-0x010001FF	0x00000110-0x000001FF	0x00000110-0x000001FF
Receive Return Ring0 RCB	16B	0x00000200-0x0000020F	0x01000200-0x0100020F	0x00000200-0x0000020F	0x00000200-0x0000020F
Receive Return Ring1 RCB	16B	0x00000210-0x0000021F	0x01000210-0x0100021F	0x00000210-0x0000021F	0x00000210-0x0000021F
Receive Return Ring2 RCB	16B	0x00000220-0x0000022F	0x01000220-0x0100022F	0x00000220-0x0000022F	0x00000220-0x0000022F
Receive Return Ring3 RCB	16B	0x00000230-0x0000023F	0x01000230-0x0100023F	0x00000230-0x0000023F	0x00000230-0x0000023F
VRQ Receive Return Ring RCBs - [4 – 16] (Relocated or expanded in BCM5717/ BCM5718)	208B	0x00000240-0x0000030F	0x01000240-0x0100030F	0x00000240-0x0000030F	0x00000240-0x0000030F
Unmapped	2.3KB	0x00000420-0x00000B4F	0x01000420-0x01000B4F	0x00000420-0x00000B4F	0x00000420-0x00000B4F
Software Gencomm	1KB	0x00000B50-0x00000F4F	0x01000B50-0x01000F4F	0x00000B50-0x00000F4F	0x00000B50-0x00000F4F
Unmapped	12KB–176B	0x00000F50-0x00003FFF	0x01000F50-0x01003FFF	0x00000F50-0x00003FFF	0x00000F50-0x00003FFF
Unmapped	32KB	0x01004000-0x0100FFFF	0x01004000-0x0100FFFF	0x00004000-0x0000FFFF	0x00004000-0x0000FFFF
RX MBUF (Relocated or expanded in BCM5717/ BCM5718)	32KB	0x00010000-0x00019FFF	0x01010000-0x01019FFF	0x00010000-0x00019FFF	0x00010000-0x00019FFF

Table 153: BCM5717 / BCM5718 Memory Map (Cont.)

Region	*Size*	*NIC CPU View*	*Host Flat View*	*Host Standard View (Offset)*	*Host UNDI View*
Unmapped	32KB	0x0001A000-0x0001FFFF	0x0101A000-0x0101FFFF	0x0001A000-0x0001FFFF	0x0001A000-0x0001FFFF
SBD Cache (Relocated or expanded in BCM5717/ BCM5718)	16KB	0x00020000-0x00023FFF	0x00020000-0x00023FFF	0x00020000-0x00023FFF	0x00020000-0x00023FFF
Std RBD Cache (Relocated or expanded in BCM5717/ BCM5718)	17KB	0x00024000-0x000283FF	0x01024000-0x010283FF	0x00024000-0x000283FF	0x00024000-0x000283FF
Jumbo RBD Cache (Relocated or expanded in BCM5717/ BCM5718)	17KB	0x00028400-0x0002C7FF	0x01028400-0x0102C7FF	0x00028400-0x0002C7FF	0x00028400-0x0002C7FF
TX MBUF (Relocated or expanded in BCM5717/ BCM5718)	29KB	0x0002C800-0x00033BFF	0x0102C800-0x01033BFF	0x0002C800-0x00033BFF	0x0002C800-0x00033BFF
Unmapped	15M +817KB	0x00033C00-0x00FFFFFF	0x01033C00-0x01FFFFFF	0x00033C00-0x00FFFFFF	0x00033C00-0x00FFFFFF
RXCPU ROM[C]	1.2KB	0x40000000-0x400004FF	–	–	–
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	–	–
Scratch Pad (Relocated or expanded in BCM5717/ BCM5718)	28KB	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF
Unmapped	4KB	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF
RXCPU ROM Slave Access	8KB	0xC0038000-0xC009FFF	0xC0038000-0xC0039FFF	0xC0038000-0xC0039FFF	0xC0038000-0xC0039FFF
Send RCB0 NIC Address	4B	0x00004000-0x0000400F	0x01004000-0x0100400F	0x00004000-0x0000400F	0x00004000-0x0000400F
Receive Standard Producer RCB0 NIC Address	4B	0x00040000-0x0004000F	0x01040000-0x0104000F	0x00040000-0x0004000F	0x00040000-0x0004000F

Table 153: BCM5717 / BCM5718 Memory Map (Cont.)

Region	*Size*	*NIC CPU View*	*Host Flat View*	*Host Standard View (Offset)*	*Host UNDI View*
Receive Jumbo Producer RCB0 NIC Address	4B	0x00044400-0x0004440F	0x01044400-0x0104440F	0x00044400-0x0004440F	0x00044400-0x0004440F

BCM5717 / BCM5718 Register Map

This is the Host Standard View of BAR1 and BAR2.

Table 154: BCM5717 / BCM5718 Register Map

Block Name	*Block Range*	*Sub-Block Range*	*Description*
HP Mail Box	0x0000–0x01FF	0x0000–0x01FF	Unused (Legacy–PCIe Configuration Register Shadow)
	0x0200–0x03FF	0x0200–0x03FF	High Priority Mail Boxes
EMAC	0x0400–0x07FF	0x0400–0x06FF	EMAC
		0x0700–0x7FF	Unused
EMAC STAT	0x0800–0x0BFF	0x0800–0x08FF	EMAC Statistics (Aggregated)
		0x0900–0x09CF	APE Private EMAC Stats (Aggregated)
		0x09D0–0x0BFF	IOV per Queue EMAC Statistics
SDI	0x0C00–0x0FFF	0x0C00–0x0CF7	Send Data Initiator
		0x0CF8–0x0FFF	Unused
SDC	0x1000–0x13FF	0x1000–0x100B	Send data Completion
		0x100C–0x13FF	Unused
SBDS	0x1400–0x17FF	0x1400–0x147F	SBDS Registers
		0x1480–0x17FF	Unused
SBDI	0x1800–0x1BFF	0x1800–0x1847	Send BD Initiator
		0x1848–0x1BFF	Unused
SBDC	0x1C00–0x1FFF	0x1C00–0x1C03	Send BD Completion
		0x1C04–0x1FFF	Unused
RQP	0x2000–0x23FF	0x2000–0x2258	Receive List Placement
		0x2259–0x23FF	Unused
RDI	0x2400–0x27FF	0x2400–0x24C3	Receive Data Initiator
		0x24C4–0x24FF	Unused
		0x2500–0x27FF	VRQ RCB Registers
RDC	0x2800–0x2BFF	0x2800–0x2803	Receive Data Completion
		0x2804–0x2BFF	Unused
RBDI	0x2C00–0x2FFF	0x2C00–0x2C1B	Receive BD Initiator
		0x2C1C–0x2FFF	Unused
RBDC	0x3000–0x33FF	0x3000–0x300F	Receive BD Completion
		0x3010–0x33FF	Unused
CMPU	0x3400–0x37FF	0x3400–0x35FF	Unused
		0x3600–0x3687	Central Power Management Unit

Table 154: BCM5717 / BCM5718 Register Map (Cont.)

Block Name	*Block Range*	*Sub-Block Range*	*Description*
DBU	0x3800–0x3BFF	0x3800–0x3817	Debug Unit (UART)
		0x3900–0x3907	Chip Debug
		0x3908–0x3BFF	Unused
HC	0x3C00–0x3FFF	0x3C00–0x3CC3	Host Coalescing
		0x3CC4–0x3FFF	Multiqueue HC
MA	0x4000–0x43FF	0x4000–0x400F	Memory Arbiter
		0x4010–0x43FF	Unused
BM	0x4400–0x47FF	0x4400–0x445B	Buffer manager
		0x445C–0x47FF	Unused
DMAR	0x4800–0x4BFF	0x4800–0x4910	DMAR LSO channel
		0x4A00–0x4AFF	DMAR BD channel
		0x4B00–0x4BFF	DMAR non-LSO channels
DMAW	0x4C00 -0x4FFF	0x4C00–0x4C07	DMA Write
		0x4C08–0x4FFF	Unused
RX-CPU	0x5000–0x53FF	0x5000–0x5037	RX CPU
		0x5038–0x53FF	Unused
VRQ Filter–EMAC	0x5400–0x57FF	0x5400–0x55FF	VRQ Filter Set
		0x5600–0x568F	VRQ Mapper
		0x5690–0x572F	VRQ Perfect Match Registers
		0x5730–0x57FF	Unused
MB	0x5800–0x5BFF	0x5800–0x5903	Low Priority Mail Box
		0x5903–0x5BFF	Unused
FTQ	0x5C00–0x5FFF	0x5C00–0x5CFF	Flow Through Queue
		0x5D00–0x5FFF	Unused
MSI	0x6000–0x63FF	0x6000–0x6007	Message Signaled Interrupt
		0x6008–0x63FF	Unused
CFG Port	0x6400–0x67FF	0x6400–0x67FF	PCIe Core Private Registers Access to Configuration Space
GRC	0x6800–0x6BFF	0x6800–0x681B	Misc Host Control
		0x6834–0x6843	SEEPROM
		0x6890–0x68A8	Misc Control
		0x68B4–0x68EF	E Switch
ASF	0x6C00–0x6FFF	0x6C00–6C37	ASF & SMBUS
		0x6C38–0x6FFF	Unused
NVM	0x7000–0x73FF	0x7000–0x703B	Non Volatile memory (Flash Controller)
		0x703C–0x73FF	Unused

Table 154: BCM5717 / BCM5718 Register Map (Cont.)

Block Name	Block Range	Sub-Block Range	Description
UART	0x7800–0x7BFF	0x7800–0x781F	Debug UART Modem
		0x7820–0x7BFF	Unused
TL-DL-PL Port	0x7C00–0x7FFF	0x7C00–0x7FFF	PCIe Core Private Register Access to TL, DL & PL
Registers below are outside Host Standard View			
APE	0x10000–0x18FFF	–	APE Access
MF	0x19000–0x193FF	–	Management Filters

BCM5719 Memory Map

Table 155: BCM5719 Memory Map

Region	Size	NIC CPU View	Host Flat View	Host Standard View (Offset)	Host UNDI View
Unmapped	256B	0x00000000-0x000000FF	0x01000000-0x010000FF	0x00000000-0x000000FF	0x00000000-0x000000FF
Send Ring [1] RCB	16B	0x00000100-0x0000010F	0x01000100-0x0100010F	0x00000100-0x0000010F	0x00000100-0x0000010F
Send Ring [2–16] RCBs	240B	0x00000110-0x000001FF	0x01000110-0x010001FF	0x00000110 0x000001FF	0x00000110-0x000001FF
Receive Return Ring0 RCB	16B	0x00000200-0x0000020F	0x01000200-0x0100020F	0x00000200-0x0000020F	0x00000200-0x0000020F
Receive Return Ring1 RCB	16B	0x00000210-0x0000021F	0x01000210-0x0100021F	0x00000210-0x0000021F	0x00000210-0x0000021F
Receive Return Ring2 RCB	16B	0x00000220–0x0000022F	0x01000220–0x0100022F	0x00000220–0x0000022F	0x00000220–0x0000022F
Receive Return Ring3 RCB	16B	0x00000230–0x0000023F	0x01000230–0x0100023F	0x00000230–0x0000023F	0x00000230–0x0000023F
VRQ Receive Return Ring RCBs [4–16]	208B	0x00000240–0x0000030F	0x01000240–0x0100030F	0x00000240–0x0000030F	0x00000240–0x0000030F
Unmapped	2.3KB	0x00000420–0x00000B4F	0x01000420–0x01000B4F	0x00000420–0x00000B4F	0x00000420–0x00000B4F
Software Gencomm	1KB	0x00000B50–0x00000F4F	0x01000B50–0x01000F4F	0x00000B50–0x00000F4F	0x00000B50–0x00000F4F
Unmapped	12KB–176B	0x00000F50–0x00003FFF	0x01000F50–0x01003FFF	0x00000F50–0x00003FFF	0x00000F50–0x00003FFF
Unmapped	32KB	0x01004000–0x0100FFFF	0x01004000–0x0100FFFF	0x00004000–0x0000FFFF	0x00004000–0x0000FFFF

Table 155: BCM5719 Memory Map (Cont.)

Region	Size	NIC CPU View	Host Flat View	Host Standard View (Offset)	Host UNDI View
RX MBUF	32KB	0x00010000– 0x00019FFF	0x01010000– 0x01019FFF	0x00010000– 0x00019FFF	0x00010000– 0x00019FFF
Unmapped	32KB	0x0001A000– 0x0001FFFF	0x0101A000– 0x0101FFFF	0x0001A000– 0x0001FFFF	0x0001A000– 0x0001FFFF
SBD Cache	16KB	0x00020000– 0x00023FFF	0x00020000– 0x00023FFF	0x00020000– 0x00023FFF	0x00020000– 0x00023FFF
Std RBD Cache	17KB	0x00024000– 0x000283FF	0x01024000– 0x010283FF	0x00024000– 0x000283FF	0x00024000– 0x000283FF
Jumbo RBD Cache	17KB	0x00028400– 0x0002C7FF	0x01028400– 0x0102C7FF	0x00028400– 0x0002C7FF	0x00028400– 0x0002C7FF
TX MBUF	29KB	0x0002C800– 0x00033BFF	0x0102C800– 0x01033BFF	0x0002C800– 0x00033BFF	0x0002C800– 0x00033BFF
Unmapped	15M +817KB	0x00033C00– 0x00FFFFFF	0x01033C00– 0x01FFFFFF	0x00033C00– 0x00FFFFFF	0x00033C00– 0x00FFFFFF
RXCPU ROM	1.2KB	0x40000000– 0x400004FF	–	–	–
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	–	–
Scratch Pad	28KB	0xC0030000– 0xC0036FFF	0xC0030000– 0xC0036FFF	0xC0030000– 0xC0036FFF	0xC0030000– 0xC0036FFF
Unmapped	4KB	0xC0037000– 0xC0037FFF	0xC0037000– 0xC0037FFF	0xC0037000– 0xC0037FFF	0xC0037000– 0xC0037FFF
RXCPU ROM Slave AccessC	8KB	0xC0038000– 0xC009FFF	0xC0038000– 0xC0039FFF	0xC0038000– 0xC0039FFF	0xC0038000– 0xC0039FFF
Send RCB0 NIC Address	4B	0x00004000- 0x0000400F	0x01004000- 0x0100400F	0x00004000- 0x0000400F	0x00004000- 0x0000400F
Receive Standard Producer RCB0 NIC Address	4B	0x00040000- 0x0004000F	0x01040000- 0x0104000F	0x00040000- 0x0004000F	0x00040000- 0x0004000F
Receive Jumbo Producer RCB0 NIC Address	4B	0x00044400- 0x0004440F	0x01044400- 0x0104440F	0x00044400- 0x0004440F	0x00044400- 0x0004440F

BCM5719 Register Map

This is the Host Standard View of BAR1 & BAR2.

Table 156: BCM5719 Register Map

Block Name	*Block Range*	*Sub-Block Range*	*Description*
HP Mail Box	0x0000–0x01FF	0x0000–0x01FF	Unused (Legacy–PCIe Configuration Register Shadow)
	0x0200–0x03FF	0x0200–0x03FF	High Priority Mail Boxes
EMAC	0x0400–0x07FF	0x0400–0x06FF	EMAC
		0x0700–0x7FF	Unused
EMAC STAT	0x0800–0x0BFF	0x0800–0x08FF	EMAC Statistics (Aggregated)
		0x0900–0x09CF	APE Private EMAC Stats (Aggregated)
		0x09D0–0x0BFF	IOV per Queue EMAC Statistics
SDI	0x0C00–0x0FFF	0x0C00–0x0CF7	Send Data Initiator
		0x0CF8–0x0FFF	Unused
SDC	0x1000–0x13FF	0x1000–0x100B	Send data Completion
		0x100C–0x13FF	Unused
SBDS	0x1400–0x17FF	0x1400–0x147F	SBDS Registers
		0x1480–0x17FF	Unused
SBDI	0x1800–0x1BFF	0x1800–0x1847	Send BD Initiator
		0x1848–0x1BFF	Unused
SBDC	0x1C00–0x1FFF	0x1C00–0x1C03	Send BD Completion
		0x1C04–0x1FFF	Unused
RQP	0x2000–0x23FF	0x2000–0x2258	Receive List Placement
		0x2259–0x23FF	Unused
RDI	0x2400–0x27FF	0x2400–0x24C3	Receive Data Initiator
		0x24C4–0x24FF	Unused
		0x2500–0x27FF	VRQ RCB Registers
RDC	0x2800–0x2BFF	0x2800–0x2803	Receive Data Completion
		0x2804–0x2BFF	Unused
RBDI	0x2C00–0x2FFF	0x2C00–0x2C1B	Receive BD Initiator
		0x2C1C–0x2FFF	Unused
RBDC	0x3000–0x33FF	0x3000–0x300F	Receive BD Completion
		0x3010–0x33FF	Unused
CMPU	0x3400–0x37FF	0x3400–0x35FF	Unused
		0x3600–0x3687	Central Power Management Unit

Table 156: BCM5719 Register Map (Cont.)

Block Name	Block Range	Sub-Block Range	Description
DBU	0x3800–0x3BFF	0x3800–0x3817	Debug Unit (UART)
		0x3900–0x3907	Chip Debug
		0x3908–0x3BFF	Unused
HC	0x3C00–0x3FFF	0x3C00–0x3CC3	Host Coalescing
		0x3CC4–0x3FFF	Multiqueue HC
MA	0x4000–0x43FF	0x4000–0x400F	Memory Arbiter
		0x4010–0x43FF	Unused
BM	0x4400–0x47FF	0x4400–0x445B	Buffer manager
		0x445C–0x47FF	Unused
DMAR	0x4800–0x4BFF	0x4800–0x4910	DMAR LSO channel
		0x4A00–0x4AFF	DMAR BD channel
		0x4B00–0x4BFF	DMAR non-LSO channels
DMAW	0x4C00–0x4FFF	0x4C00–0x4C07	DMA Write
		0x4C08–0x4FFF	Unused
RX-CPU	0x5000–0x53FF	0x5000–0x5037	RX CPU
		0x5038–0x53FF	Unused
VRQ Filter-EMAC	0x5400–0x57FF	0x5400–0x55FF	VRQ Filter Set
		0x5600–0x568F	VRQ Mapper
		0x5690–0x572F	VRQ Perfect Match Registers
		0x5730–0x57FF	Unused
MB	0x5800–0x5BFF	0x5800–0x5903	Low Priority Mail Box
		0x5903–0x5BFF	Unused
FTQ	0x5C00–0x5FFF	0x5C00–0x5CFF	Flow Through Queue
		0x5D00–0x5FFF	Unused
MSI	0x6000–0x63FF	0x6000–0x6007	Message Signaled Interrupt
		0x6008–0x63FF	Unused
CFG Port	0x6400–0x67FF	0x6400–0x67FF	PCIe Core Private Registers Access to Configuration Space
GRC	0x6800–0x6BFF	0x6800–0x681B	Misc Host Control
		0x6834–0x6843	SEEPROM
		0x6890–0x68A8	Misc Control
		0x68B4–0x68EF	E Switch
ASF	0x6C00–0x6FFF	0x6C00–6C37	ASF & SMBUS
		0x6C38–0x6FFF	Unused
NVM	0x7000–0x73FF	0x7000–0x703B	Non Volatile memory (Flash Controller)
		0x703C–0x73FF	Unused

Table 156: BCM5719 Register Map (Cont.)

Block Name	Block Range	Sub-Block Range	Description
UART	0x7800–0x7BFF	0x7800–0x781F	Debug UART Modem
		0x7820–0x7BFF	Unused
TL-DL-PL Port	0x7C00–0x7FFF	0x7C00–0x7FFF	PCIe Core Private Register Access to TL, DL & PL
Registers below are outside Host Standard View			
APE	0x10000–0x18FFF	–	APE Access
MF	0x19000–0x193FF	–	Management Filters

BCM5720 Memory Map

The BCM5720 internal memory map is shown in Table 157.

Table 157: BCM5720 Memory Map

Region	Size	NIC CPU View	Host Flat View	Host Standard View (Offset)	Host UNDI View
Unmapped	256B	0x00000000-0x000000FF	0x01000000-0x010000FF	0x00000000-0x000000FF	0x00000000-0x000000FF
Send Ring [1] RCB	16B	0x00000100-0x0000010F	0x01000100-0x0100010F	0x00000100-0x0000010F	0x00000100-0x0000010F
Send Ring [2 – 16] RCBs	240B	0x00000110-0x000001FF	0x01000110-0x010001FF	0x00000110-0x000001FF	0x00000110-0x000001FF
Receive Return Ring0 RCB	16B	0x00000200-0x0000020F	0x01000200-0x0100020F	0x00000200-0x0000020F	0x00000200-0x0000020F
Receive Return Ring1 RCB	16B	0x00000210-0x0000021F	0x01000210-0x0100021F	0x00000210-0x0000021F	0x00000210-0x0000021F
Receive Return Ring2 RCB	16B	0x00000220-0x0000022F	0x01000220-0x0100022F	0x00000220-0x0000022F	0x00000220-0x0000022F
Receive Return Ring3 RCB	16B	0x00000230-0x0000023F	0x01000230-0x0100023F	0x00000230-0x0000023F	0x00000230-0x0000023F
VRQ Receive Return Ring RCBs - [4 – 16]	208B	0x00000240-0x0000030F	0x01000240-0x0100030F	0x00000240-0x0000030F	0x00000240-0x0000030F
Unmapped	2.3KB	0x00000420-0x00000B4F	0x01000420-0x01000B4F	0x00000420-0x00000B4F	0x00000420-0x00000B4F
Software Gencomm	1KB	0x00000B50-0x00000F4F	0x01000B50-0x01000F4F	0x00000B50-0x00000F4F	0x00000B50-0x00000F4F
Unmapped	12KB – 176B	0x00000F50-0x00003FFF	0x01000F50-0x01003FFF	0x00000F50-0x00003FFF	0x00000F50-0x00003FFF

Table 157: BCM5720 Memory Map (Cont.)

Region	Size	NIC CPU View	Host Flat View	Host Standard View (Offset)	Host UNDI View
Unmapped	32KB	0x01004000-0x0100FFFF	0x01004000-0x0100FFFF	0x00004000-0x0000FFFF	0x00004000-0x0000FFFF
RX MBUF	32KB	0x00010000-0x00019FFF	0x01010000-0x01019FFF	0x00010000-0x00019FFF	0x00010000-0x00019FFF
Unmapped	32KB	0x0001A000-0x0001FFFF	0x0101A000-0x0101FFFF	0x0001A000-0x0001FFFF	0x0001A000-0x0001FFFF
SBD Cache	16KB	0x00020000-0x00023FFF	0x00020000-0x00023FFF	0x00020000-0x00023FFF	0x00020000-0x00023FFF
Std RBD Cache	17KB	0x00024000-0x000283FF	0x01024000-0x010283FF	0x00024000-0x000283FF	0x00024000-0x000283FF
Jumbo RBD Cache	17KB	0x00028400-0x0002C7FF	0x01028400-0x0102C7FF	0x00028400-0x0002C7FF	0x00028400-0x0002C7FF
TX MBUF	29KB	0x0002C800-0x00033BFF	0x0102C800-0x01033BFF	0x0002C800-0x00033BFF	0x0002C800-0x00033BFF
Unmapped	15M +817KB	0x00033C00-0x00FFFFFF	0x01033C00-0x01FFFFFF	0x00033C00-0x00FFFFFF	0x00033C00-0x00FFFFFF
RXCPU ROM	1.2KB	0x40000000-0x400004FF	–	–	–
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	–	–
Scratch Pad	28KB	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF	0xC0030000-0xC0036FFF
Unmapped	4KB	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF	0xC0037000-0xC0037FFF
RXCPU ROM Slave Access	8KB	0xC0038000-0xC009FFF	0xC0038000-0xC0039FFF	0xC0038000-0xC0039FFF	0xC0038000-0xC0039FFF
Send RCB0 NIC Address	4B	0x00004000-0x0000400F	0x01004000-0x0100400F	0x00004000-0x0000400F	0x00004000-0x0000400F
Receive Standard Producer RCB0 NIC Address	4B	0x00040000-0x0004000F	0x01040000-0x0104000F	0x00040000-0x0004000F	0x00040000-0x0004000F

Table 157: BCM5720 Memory Map (Cont.)

Region	Size	NIC CPU View	Host Flat View	Host Standard View (Offset)	Host UNDI View
Receive Jumbo Producer RCB0 NIC Address	4B	0x00044400-0x0004440F	0x01044400-0x0104440F	0x00044400-0x0004440F	0x00044400-0x0004440F

BCM5720 Register Map

The BCM5720 internal register map of each MAC unit is shown in Table 158. This is the Host Standard View of BAR1 & BAR2.

Table 158: BCM5720 Register Map

Block Name	Block Range	Sub-Block Range	Description
HP Mail Box	0x0000 – 0x01FF	0x0000 – 0x01FF	Unused {Legacy - PCIe Configuration Register Shadow}
	0x0200 – 0x03FF	0x0200 – 0x03FF	High Priority Mail Boxes
EMAC	0x0400 – 0x07FF	0x0400 – 0x06FF	EMAC
		0x0700 – 0x7FF	Unused
EMAC STAT	0x0800 – 0x0BFF	0x0800 – 0x08FF	EMAC Statistics (Aggregated)
		0x0900 – 0x09CF	APE Private EMAC Stats (Aggregated)
		0x09D0 – 0x0BFF	IOV per Queue EMAC Statistics
SDI	0x0C00 – 0x0FFF	0x0C00 – 0x0CF7	Send Data Initiator
		0x0CF8 – 0x0FFF	Unused
SDC	0x1000 – 0x13FF	0x1000 – 0x100B	Send data Completion
		0x100C – 0x13FF	Unused
SBDS	0x1400 – 0x17FF	0x1400 – 0x147F	SBDS Registers
		0x1480 – 0x17FF	Unused
SBDI	0x1800 – 0x1BFF	0x1800 – 0x1847	Send BD Initiator
		0x1848 – 0x1BFF	Unused
SBDC	0x1C00 – 0x1FFF	0x1C00 – 0x1C03	Send BD Completion
		0x1C04 – 0x1FFF	Unused
RQP	0x2000 – 0x23FF	0x2000 – 0x2258	Receive List Placement
		0x2259 – 0x23FF	Unused
RDI	0x2400 – 0x27FF	0x2400 – 0x24C3	Receive Data Initiator
		0x24C4 – 0x24FF	Unused
		0x2500 – 0x27FF	VRQ RCB Registers
RDC	0x2800 – 0x2BFF	0x2800 – 0x2803	Receive Data Completion
		0x2804 – 0x2BFF	Unused

Table 158: BCM5720 Register Map (Cont.)

Block Name	Block Range	Sub-Block Range	Description
RBDI	0x2C00 – 0x2FFF	0x2C00 – 0x2C1B	Receive BD Initiator
		0x2C1C – 0x2FFF	Unused
RBDC	0x3000 – 0x33FF	0x3000 – 0x300F	Receive BD Completion
		0x3010 – 0x33FF	Unused
CMPU	0x3400 – 0x37FF	0x3400 – 0x35FF	Unused
		0x3600 – 0x3687	Central Power Management Unit
DBU	0x3800 – 0x3BFF	0x3800 – 0x3817	Debug Unit (UART)
		0x3900 – 0x3907	Chip Debug
		0x3908 – 0x3BFF	Unused
HC	0x3C00 – 0x3FFF	0x3C00 – 0x3CC3	Host Coalescing
		0x3CC4 – 0x3FFF	MultiQueue HC
MA	0x4000 – 0x43FF	0x4000 – 0x400F	Memory Arbiter
		0x4010 – 0x43FF	Unused
BM	0x4400 – 0x47FF	0x4400 – 0x445B	Buffer manager
		0x445C – 0x47FF	Unused
DMAR	0x4800 – 0x4BFF	0x4800 – 0x4910	DMAR LSO channel
		0x4A00 – 0x4AFF	DMAR BD channel
		0x4B00 – 0x4BFF	DMAR non-LSO channels
DMAW	0x4C00 -0x4FFF	0x4C00 – 0x4C07	DMA Write
		0x4C08 – 0x4FFF	Unused
RX-CPU	0x5000 – 0x53FF	0x5000 – 0x5037	RX CPU
		0x5038 – 0x53FF	Unused
VRQ Filter-EMAC	0x5400 – 0x57FF	0x5400 – 0x55FF	VRQ Filter Set
		0x5600 – 0x568F	VRQ Mapper
		0x5690 – 0x572F	VRQ Perfect Match Registers
		0x5730 – 0x57FF	Unused
MB	0x5800 – 0x5BFF	0x5800 – 0x5903	Low Priority Mail Box
		0x5903 – 0x5BFF	Unused
FTQ	0x5C00 – 0x5FFF	0x5C00 – 0x5CFF	Flow Through Queue
		0x5D00 – 0x5FFF	Unused
MSI	0x6000 – 0x63FF	0x6000 – 0x6007	Message Signaled Interrupt
		0x6008 – 0x63FF	Unused
CFG Port	0x6400 – 0x67FF	0x6400 – 0x67FF	PCIe Core Private Registers Access to Configuration Space

Table 158: BCM5720 Register Map (Cont.)

Block Name	*Block Range*	*Sub-Block Range*	*Description*
GRC	0x6800 – 0x6BFF	0x6800 – 0x681B	Misc Host Control
		0x6834 – 0x6843	SEEPROM
		0x6890 – 0x68A8	Misc Control
		0x68B4 – 0x68EF	E Switch
ASF	0x6C00 – 0x6FFF	0x6C00 – 6C37	ASF & SMBUS
		0x6C38 – 0x6FFF	Unused
NVM	0x7000 – 0x73FF	0x7000 – 0x703B	Non Volatile memory (Flash Controller)
		0x703C – 0x73FF	Unused
UART	0x7800 – 0x7BFF	0x7800 – 0x781F	Debug UART Modem
		0x7820 – 0x7BFF	Unused
TL-DL-PL Port	0x7C00 – 0x7FFF	0x7C00 – 0x7FFF	PCIe Core Private Register Access to TL, DL & PL
The registers below are outside Host Standard View.			
APE	0x10000 – 0x18FFF	–	APE Access
MF	0x19000 – 0x193FF	–	Management Filters

Connecting everything®

BROADCOM CORPORATION
5300 California Avenue
Irvine, CA 92617

Phone: 949-926-5000
Fax: 949-926-5203
E-mail: info@broadcom.com
Web: www.broadcom.com

5718-PG107-R July 17, 2013