Intel® Ethernet Switch FM10000

Datasheet

Ethernet Networking Division (ND)

Revision 2.1
April 2018
333497-002

Revision History

Revision	Date	Comments
2.1	April 3, 2018	Updates include the following: • Updated Section 7.4.1, “DFE Tuning & Emphasis”. • Added Section 12.4.6, “FM10840/FM10420 Voltage Initial Conditions for Fusebox Read”.
2.0	May 12, 2016	Initial release (Intel public).

Contents

NOTE: This page intentionally left blank.

1.0 Introduction

1.1 Scope

This specification defines the FM10000 architecture as well as registers definition and electrical, mechanical and thermal specifications for the product.

1.2 Product Applicability

The FM10000 is available in two SKUs, listed in Table 1-1.

Table 1-1 FM10000 Product SKUs

Product	Max Ethernet Port Usage					PCIe Data Ports	Max Bandwidth (Gb/s)				Switch Frequency (MHz)
	100G	40G	25G	10G	SGMII		Ethernet	PCIe[1]	TE	Max[2]	
FM10840	4	9	24	36	36	8x4, 4x8	400	200	200	600	980
FM10420	2	2	8	8	8	4x4, 2x8	200	100	200	400	600[3]

Product	EPL Port Location	PEP Port Location	PEP BW Selection	Estimated WC[4] Power (W)	Use Cases
FM10840	Port[0..35], up to 400G bandwidth.	PCIe x8: PEPs [0..8] are all available PCIe x4: PEPs [0..8] are all available	PCIe x8: 50/25/10/2.5 per port PCIe x4: 25/10/2.5 per port	44.8	High-performance compute fabric.
FM10420	Port[0..35], up to 200G bandwidth.	PCIe X8: PEP [0,2] PCIe X4: PEP [0,1,2,3], or PEP [0,2,4,6]		28.3	High Performance 10/40/100 GbE NIC.

1. Unused PCIe bandwidth can be used for Ethernet.
2. The total bandwidth allowed is the sum of Ethernet, PCIe, and Tunneling. The total may not exceed the Maximum for the SKU in question.
3. The default switch frequency of FM10420 at is set to 600 MHz in case 100 GbE ports are used. 600 MHz is the minimum frequency to support 100 GbE at baseline (`Total Bandwidth x 1.5 + 25 MHz`) requirement. The switch frequency can still be reduced to 500, 400, 300MHz if 100 GbE is not used; see PLL_FABRIC_LOCK.*FeatureCode* in Section 11.26.2.5.
4. WC = Worse Case refers to Max condition and can vary based on application (see Section 12.6.2).

The information about SKUs and default options are available from following register locations:

- DEVICE_CFG.*FeatureCode*

  00b = FULL — All PEPs available (0..8)
  01b = HALF — PEPs 0,1,2,3,4,6,8 can be used. Hosts 0,1,2,3 can operate in either mode.
  Hosts 4 and 6 can only be used in 4-lane mode.
  10b = BASIC — Only one PEP interface available (0 or 8)
  11b = Reserved

- DEVICE_CFG.*Eth100GDisabled*

  0b = Disabled
  1b = Enabled

- PLL_FABRIC_LOCK.*FeatureCode*

  0000b = No speed restrictions
  0001b = Support 600,500,400,300 MHz
  0010b = Support 500,400,300 MHz
  0011b = Support 400,300 MHz
  0100b = Support 300 MHz
  All other values are reserved.

- FUSE_DATA_0[15:11] SKU number

  00000b = FM10840
  00001b = FM10420

1.3 Document Organization

This document is organized as follows:

- Section 1.0, "Introduction"
- Section 2.0, "Architecture Overview"
- Section 3.0, "Pin Descriptions"
- Section 4.0, "Reset, Boot and Clocking"
- Section 5.0, "Ethernet Switch"
- Section 6.0, "PCIe Host Interface"
- Section 7.0, "Ethernet Port Logic (EPL)"
- Section 8.0, "Tunneling Engine"
- Section 9.0, "Peripherals"
- Section 10.0, "Reliability, Diagnostics and Testability"
- Section 11.0, "Register Definitions"
- Section 12.0, "Electrical Specification"
- Section 13.0, "Mechanical Specification"

1.4 Definitions

Table 1-2 Terminology Definitions

Term	Definition
{A,B}	Denotes a bit concatenation of variable A or B where A is most significant bit field and B is least significant bit field. ***Note:*** {0,A} means that 0s are added to the left (most significant bits) to pad to the desired size while {A,0} means padded to the right.
Bit Numbering	Bit 0 is the least significant bit throughout the architecture (even if Ethernet standards and specifications suggest otherwise).
Byte	8 bits.
Double Word	64 bits.
EBI	External Bus Interface — Intel's term for a legacy address/data external bus interface to processor.
F32/F64/F96	Intel-proprietary inter-switch link tag, which is used to pass relevant management and control information from one Intel Ethernet and Switch Family device to another in a network. Not supported in the FM2000 series.
FM2000	The first generation of the Intel® Ethernet and Switch Family. The FM2000 is a Layer 2 (L2) 10 GbE switch chip platform that forms the basis for many L2 switch product variants, including the FM2224, FM2212, FM2208, FM2112, FM2104, and FM2103.
FM4000	The second generation of the Intel® Ethernet and Switch Family. The FM4000 is an enhanced multi-layer 10 GbE switch chip platform that forms the basis for new switch product variants, all of which are pin-compatible with their original FM2000 series counterparts. The FM4xxx devices are full-featured L3 routing devices, and contain other enhancements, such as ACLs, congestion management, increased frame memory and more. The FM3xxx devices are similar to the FM4xxx, except that L3 routing is not included.
FM5000/FM6000	The third generation of the Intel® Ethernet and Switch Family. The FM5000/FM6000 enhances functionality and bandwidth while providing package options that are pin compatible with the FM4000 series. The FM5000/FM6000 L2/L3/L4 devices provide advanced CEE/DCB features for the Data Center.
FM10000	The fourth generation of the Intel® Ethernet and Switch Family. The Intel® FM10000 family combines integrated Ethernet controllers with an advanced Ethernet switch architecture to provide a multi-host Ethernet controller for low latency rack scale server platforms and high-performance communications infrastructure applications. The FM10000 series supports enhanced features critical for today’s need for high performance environments: low latency, scalability, L3 routing, Priority Flow Control, Enhanced Transmission Selection, as well as support for VXLAN, NVGRE, EVB, NSH and software defined networking. The FM10000 series supports up to 36 10 GbE ports and the ability to group four ports as 40 GbE or 100 GbE links. Four integrated Ethernet controllers can be used as standard 50 Gb/s host interfaces, while providing low latency interfaces into the integrated Ethernet switch.
GENEVE	IETF Geneve (GENEVE) is one of the three supported tunnels in the FM10000 (along with NVGRE and VXLAN). GENEVE is also known as NGE.
GloRT	Intel-proprietary Global Resource Tag, which is used to pass global identification information from one device to another in a network. GloRT is the proper pronunciation, although other uses might appear in the document and have the same meaning. For example, glort, Glort, GloRT or GLORT.
Half Word	16 bits.
Logging	Logging refers to a copy of the frame sent to a local CPU for monitoring purposes.
Mirroring	Mirroring refers to a copy of the frame sent to another port for monitoring purposes.

Table 1-2 Terminology Definitions [Continued]

Term	Definition
Packet or Frame	**Packet** — On a typical computer network, data is transmitted in the form of structured and modest-sized packets. Instead of transmitting arbitrary-length strings of data, structured packets Packet or Frame allow error checking and other relevant processing to occur on smaller easier-to-retransmit data. Packetized data also helps to alleviate traffic jams on the network when multiple nodes are contending for a shared network resource. **Frame** — While a packet is a small block of data, a Frame is the definition of how packets of data are defined and transported on a specific network. When sending data over a network, both sides of the connection must agree on a common frame format (e.g., when a frame starts, when a frame ends, padding, etc.) Combining terms, an Ethernet packet is sent onto an Ethernet interface using an Ethernet frame format. This document uses both terms interchangeably.
PCI Express* (PCIe*)	PCIe Interface to a processor. Compliant to PCIe Gen3/2/1.
Register Type	Registers are split into fields of the following types: RW Read/Write SRW Read/Write (Service reset) RO Read-Only (This register cannot be programmed by software. Typically reports a status.) CW Clear-on-Write (Any value written clears the register.) CW1 Clear-on-Write-1 (Writing 1b to any bit clears that bit. Writing 0b has no effect.) SW1 Set-on-Write-1 (Writing 1b to any bit sets that bit. Writing 0b has no effect.) CR Clear-on-Read (Reading the register clears the register.) RV Reserved (For upward compatibility. Always write as zero, ignore on read.) WO Write-Only (The value written cannot be read back.)
Segment	A portion of a packet corresponding to one of the common architecturally-significant storage and processing chunks that has been defined in the architecture. For FM10000, a segment in 192 bytes.
Trapping	Trapping refers to special frames that are captured by the switch and redirected to a local CPU for processing.
Word	32 bits.
X[0..N]	Denotes an array with indexes that go from 0 through N, inclusively.
X[N:0] or X[0:N]	Denotes a bit range within a variable. The bit range [0:N] indicates that bit 0 is the most significant, while bit range [N:0] indicates that the bit 0 is the least significant.

2.0 Architecture Overview

2.1 Overview

The FM10000 is a 48-port (36 Ethernet, 9 PCIe, 2 Tunneling Engines and one FBIM) Ethernet switch based on a shared memory fabric architecture supporting Ethernet ports and PCI Express (PCIe) host interfaces. Figure 2-1 shows the main elements of the device.

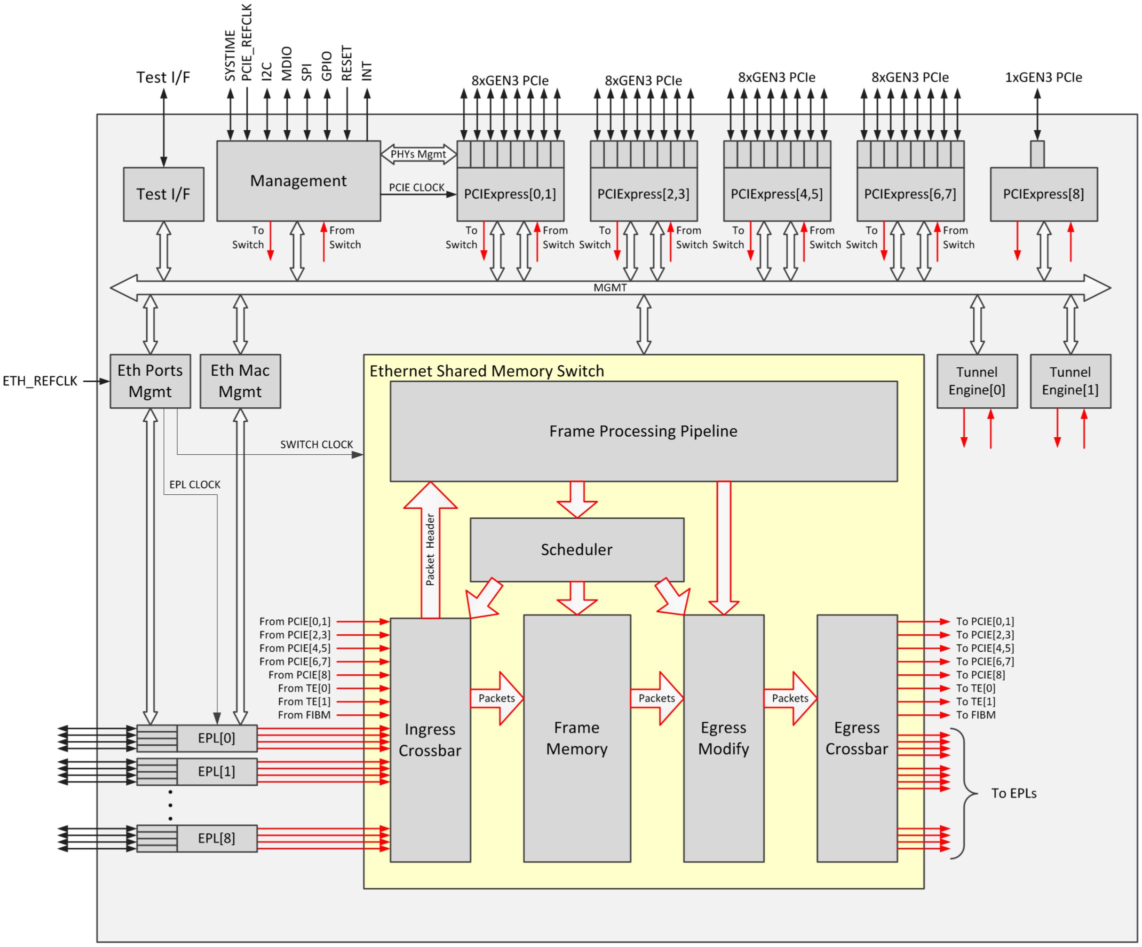


Figure 2-1 FM10000 Block Diagram

The Ethernet Port Logic (EPL) are the actual Ethernet interfaces. The FM10000 supports nine EPLs which have 4 SerDes each for a total of 36 external ports. Each SerDes is capable of supporting rates from 1.25 GbE to 25.78125 GbE and could be used to support either four independent ports each running at their own speed (1 GbE, 2.5 GbE, 10 GbE or 25 GbE) or a combination of four SerDes in one logical port (40 GbE, 100 GbE). The EPL contains the PCS and MAC layers to serialize or deserialize data frames.

The Ports Mgmt is a block acting as a bridge to manage the SerDes on the Ethernet ports. This block also includes the PLLs to generate the clocks for EPL and the Ethernet Shared Memory Switch and clock observations outputs.

The Ethernet Shared Memory Switch is the core component of the device and is responsible to switch frames between ports; all ports are equivalent from a switch perspective, external (Ethernet) or internals (PCIe, FIBM). The fabric includes the following components:

- Ingress crossbar
- Main memory
- Egress modification
- Egress crossbar
- Scheduler
- Frame processing pipeline

Data frames incoming into the switch are fragmented into segments of up to 192 bytes which are stored in memory by the ingress crossbar. The scheduler responsibilities are to manage the segment pool, send commands to the ingress and egress crossbars to get segments into memory or out of memory, reports arrival of segments or frame processing pipeline and schedule frames transmission according to priorities and shaping profiles programmed. The scheduler also instructs the ingress crossbar to send the first segment of the frame to the frame processing pipeline for a switching decision.

The frame processing pipeline responsibilities are to parse the frame headers and do necessary table lookups to determine the set of ports the frame is sent to. The forwarding decision is forwarded to the scheduler, and new data generated for this new frame is forwarded to the MODIFY unit. The frame processing pipeline also keeps track of segments consumed on ingress or freed on egress for accounting purposes.

The MODIFY unit is instructed by the scheduler of segment transmission, and applies the necessary transforms on the first segment of the frame using information stored for that frame which was supplied earlier by the frame processing pipeline. Modifications could include, for example, modifying a VLAN tag or routing the frame.

The PCIe Host Interfaces are the network interfaces between hosts and the switch. They include the necessary DMA engines to efficiently transfer frames from host to switch or from switch to host. The FM10000 supports five Host Interfaces. The first four contain two PCIe End Points (PEPs) and could operate either in 2x4 lane mode (both PEPs active) or in 1x8 lane mode (one PEP active, one PEP down). The last interface contains one PCIe End Point and only one lane.

The FM10000 could be controlled through various management interfaces:

- PCIe End Points (PEP)
- FTAG In-Band Management (FIBM)
- Low-speed Management (LSM)
- Test Interface

The primary management interfaces are the PCIe End Points (PEPs), which can be used to manage the switch and also can be used to transmit frames from the attached hosts to the fabric or to receive frames from the fabric. From a fabric perspective, the PEP are not different than an Ethernet port and thus frames from/to PCIe are switched the same way as any other frame. Each of the PCIe host interface supports 8 lanes and is v3.0 compliant (a.k.a. GEN3).

A secondary management interface is the in-band-management which is a unit designed to interpret and respond to specialized formatted Ethernet frames to read or write internal registers in the switch remotely. This management interface is connected to the fabric allowing receiving and sending FIBM frames from any port.

Finally, the FM10000 also supports a set of slower speed interfaces grouped into two blocks; low-speed management and test interface. The test interface includes a 33 MHz address/data bus and a scan infrastructure used by production. The low-speed management block includes an I^{2}C interface, a JTAG interface, an internal monitoring interface, SPI interface, an MDIO interface, a LED interface and a GPIO interface. The low-speed management is also responsible to handle reference clocks, generate internal high speed clocks and implement reset control, watchdogs and boot.

2.2 Management Overview

The management infrastructure is shown in Figure 2-2.

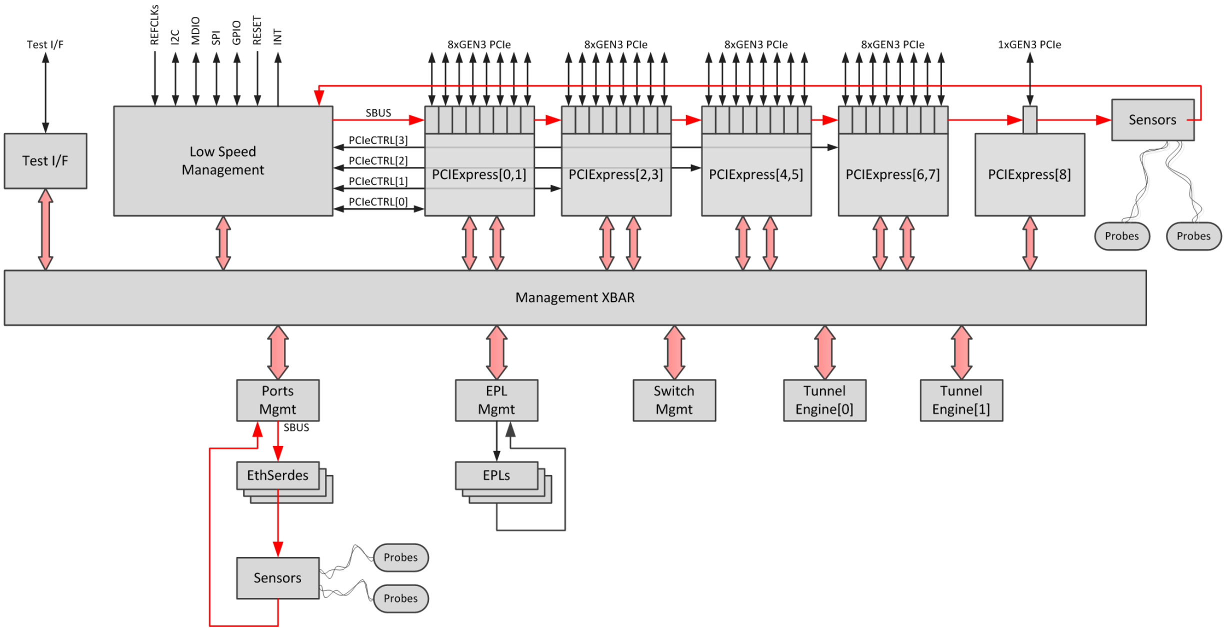


Figure 2-2 Management Overview Block Diagram

The Low-Speed-Management (LSM) is the first unit to come out of reset and is responsible for booting the device. The LSM normally retrieves a configuration from a serial FLASH memory to setup the device initially. This configuration could be minimal or exhaustive depending of the desired mode of operation. As a example, a minimum configuration could be to configure the internal PLLS to provide necessary clocks to the system, configure a set of PCIe SerDes and take one PCIe interface out of reset, thus allowing a host connected on PCIe to take over. The details on LSM are covered in the Low-Speed Management chapter. The LSM also extends itself to provide a serial bus (SBUS) interface that daisy chains all PCIe SerDes and also access to sensors which are attached to probes inside the chip. The LSM include also the FIBM which provides an Ethernet frame-based method to manage the device.

The Test Interface (TEST) is also immediately available after reset and provides an interface for testing purpose. The Test Interface is not intended to be used in normal applications.

The PCIe interfaces are the usual interfaces to manage the device.

All these management interfaces (9xPCIe, 1xLSM, 1xTEST) could initiate read/write requests to any register in the switch. The central management crossbar is the unit that arbitrates the access from each initiator to each target. There are 14 targets:

- LSM/FIBM
- EPLs/Ports Management
- 9 x PCIe
- 2 x Tunnel Engines
- Switch

The Switch, Tunnels, and EPLs and Ports management interfaces are targets only and are used to access registers and tables in the switch and tunnel engines, registers in EPLs and registers in Ports management as well as Ethernet SerDes (Ethernet SerDes accessible via an SBUS).

The switch, EPLs, tunnels could receive transactions from any master and up to the 11 possible masters. The master interfaces however have more limited capabilities as targets; the TEST is never a target, the LSM supports only up to 3 requests at any time and the PCIe supports up to 8.

The transactions on the crossbar are simple:

- Read request
- Write request
- Read response

The ordering of read/write requests, and of read responses, between a master and a target is always maintained. All transactions on the crossbar are in 32-bit data words; no byte or half-word accesses are supported.

2.2.1 Access to 64-bit or 128-bit Registers

Some registers in the system are larger than 32 bits (either 64 or 128 bits), and the read/write of those registers is done atomically. It is expected that a given initiator writes or reads all the words of the register in increasing address order (from least to most significant 32-bit word), without interleaving other management access. If the host follows that pattern, the mechanism below guarantees that the hardware register access is atomic.

To ensure atomic access during write accesses, each target keeps a scratch register per initiator, to store all but the most significant word written from that initiator for registers larger than 32 bits. The target performs the actual atomic write at the designated register address when the most significant word of the register is written, using the scratch register for that initiator to provide the lower significant words. The write address of the most significant word is used to determine the address of the overall atomic write.

Similarly, on read, the target keeps local scratch read registers per initiator, holding address and data of larger-than-32-bit register. When the initiator first reads any word of a register, the target reads the register atomically, stores its address and data into this scratch register and returns the content of the word addressed to the initiator. Successive word reads on the same register returns the content of the scratch register as long as it is not the first word of the register. Reads of the least significant word or reads on a different register cause the scratch register content to be refreshed by the target redoing an atomic read.

2.2.2 Global Register Space

The global register space is defined as follows (in 32-bit words):

```
 23                       0

 00000000_xxxxxxxx_xxxxxxxx 64KW => LSM/FIBM
 00000001_xxxxxxxx_xxxxxxxx \
 ...                         > 13*64KW => Do not access
 00001101_xxxxxxxx_xxxxxxxx /            (TEST is at 0x0DXXXX)
 00001110_0xxxxxxx_xxxxxxxx 32KW => EPLs
 00001110_1xxxxxxx_xxxxxxxx 32KW => PORTS MGMT
 00001111_xxxxxxxx_xxxxxxxx 64KW => Do not access
|0000port|
         |
         +---> The four bits [19:16] is the management crossbar port if
               top four bits are zero.

 0001xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[0]
 0010xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[1]
 0011xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[2]
 0100xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[3]
 0101xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[4]
 0110xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[5]
 0111xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[6]
 0100xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[7]
 0101xxxx_xxxxxxxx_xxxxxxxx   1MW => GLOBAL PCI EXPRESS[8]
 0110xxxx_xxxxxxxx_xxxxxxxx   1MW => TUNNEL ENGINE[0]
 0111xxxx_xxxxxxxx_xxxxxxxx   1MW => TUNNEL ENGINE[1]
|port|
     |
     +----> The top four bits is the management crossbar port
            except for 0,12,13,14,15.

 11xxxxxx_xxxxxxxx_xxxxxxxx   4MW => Ethernet Shared Memory Switch
|port|
     |
     +----> Mapped to crossbar port #15.
```

2.2.3 Inactive or Non-existent Registers

Management read or write requests should not be made to the address ranges 0x010000-0x0DFFFF or 0x0F0000-0x0FFFFF, marked as "Do not access" in the table above. Also, when a PCIe host interface is disabled, as indicated by the DEVICE_CFG.*PCIeEnable*[i] configuration bit, management read or write requests must not be made to any address in its address space, shown as the GLOBAL PCI EXPRESS[i] space in the definition presented in Section 2.2.2. Violating either of these rules can lead to deadlock of the management infrastructure.

Aside from the above restrictions, a master can read at any address. The read request completes successfully, even if there are no registers or tables defined for this address. A read from the address space of a unit in reset always returns zero. The returned value for non-existing registers is undefined, but usually zero.

A master always writes to existing registers or tables. The behavior of the hardware is not defined if software attempts to write to an address for which no registers or tables exist. Writing to the address space of a unit in reset has no effect.

2.3 Switch Aggregation and Frame Tagging

The FM10000 supports the addition of a Fabric Tag (FTAG) to carry special information from Switch to Switch, between Switch and PCIe Host Interface or between Switch and Tunneling Engines. This tag is essential for a set of switches to behave like one switch (switch aggregation). Figure 2-3 shows where FTAG is normally used.

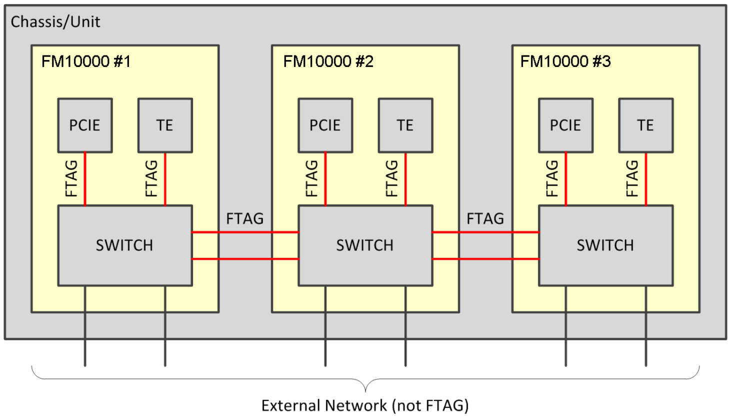


Figure 2-3 FTAG Usage

The FTAG is placed at the beginning of the frame. The Ethernet frame preamble is re-used to carry this tag for Ethernet ports avoiding using extra bandwidth. It is simply added at the beginning of the frame for PCIe Host Interfaces and Tunneling Engines. Figure 2-4 shows the different cases.

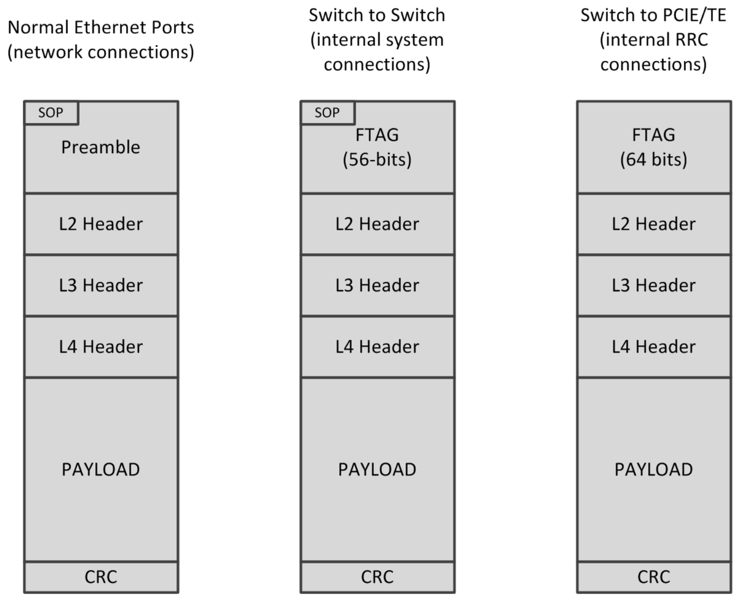


Figure 2-4 Frame Tagging

There are two formats for the FTAG. The first format is on Ethernet ports for Switch to Switch, the second format is used for frames between the PCI Express Host Interfaces or Tunneling Engines and the Switch function. The only difference is the USER field, which is not available on Ethernet ports.

	31 … 24	23		… 16	15	… 8 7 … 0
WORD 0	USER(PCIE/TE)	FTYPE	00	SWPRI	VLAN	
	SOP(Ethernet)				VPRI	VID
WORD 1	SGLORT				DGLORT	

Figure 2-5 FTAG Format

The presence of FTAG on Ethernet port is configured on a per port basis, and are only set on secure links for switch to switch intercommunication. If enabled, all frames on this port gets an FTAG.

The FTAG contains the following fields:

- USER (8 bits) — General purpose 8-bit field.
- FTYPE (2 bits) — Frame type.
- SWPRI (4 bits) — Assigned frame switch priority.
- DGLORT (16 bits) — Destination global resource tag.
- SGLORT (16 bits) — Source global resource tag.
- VPRI (4 bits) — VLAN priority.
- VID (12 bits) — VLAN ID, must be non-null.

2.3.1 GLORT

The Global Resource Tag (GLORT) is a 16-bit number that identifies a logical port or multicast group or link aggregation group or load-balancing group. When a frame is received from an external Ethernet port, it is first associated with a default Source GLORT (SGLORT) and a null Destination GLORT (DGLORT). Then, one of the following stages, FFU, NextHop, L2Lookup, replaces the null DGLORT with a non-null one. The destination mask generation stage transforms the DGLORT into a destination mask (48 bits) and the scheduler forwards the frame to those ports identified in the destination mask. Therefore, the DGLORT could represent by example a single port, or a set of ports (multicasting).

The SGLORT and DGLORT are transported over to the next Switch using the FTAG, allowing a set of switches to operate logically as one. The next switch uses the supplied GLORTs in the FTAG to decide how to switch the frame and identify the original source port for this frame.

2.3.2 USER Field

The USER field is defined as follows for PCIe.

- USER[0] — Set by switch to indicate ingress VID VID = egress VID when frame is L3 multicasted (this multicast copy was switched).
- USER[1] — Set by PCI Express to indicate request for time stamping. Bit is used to initialize CAPTURE_TIME flag.

The USER field is general purpose for the Tunneling Engines. Refer to Section 8.0, "Tunneling Engine" for its usage.

The USER field is not available on Ethernet ports and is assumed to be 0 on reception.

2.3.3 FTYPE

The FTYPE indicates the frame type, there are two types supported in the FM10000:

00b = NORMAL
10b = SPECIAL

Normal are those switched or routed normally.

Mirrored or trapped frames are tagged SPECIAL. Furthermore, switches are expected to deliver the special frames using the DGLORT only bypassing normal L2 switching or L3 routing. These follow-on switches do not modify the content of the frames nor change the frame type further. The Triggers unit is capable to alter the direction if needed but not normally expected to do so.

2.3.4 VID/VPRI

The VID and VPRI are the VLAN Id and VLAN Priority for this frame. Each frame has a VID/VPRI associate when the frame is traveling across FTAG-ed links.

2.3.5 SWPRI

The SWPRI identified the switch priority for this frame. Rules are available on each port to define if the switch priority is used as supplied in the FTAG or derived from another priority field.

2.4 Frame Sizes and Frame Rate

The FM10000 switch supports valid frames in sizes from 15 bytes to 15 KB. The switch supports handling invalid frames shorter than 15 bytes. In this case, the switch might either drop the packet or forward it depending on cut-through mode of operation. If the switch has to forward the invalid tiny frame, the frame exits with a bad CRC, but might be modified in an unpredictable way while being transmitted.

For Ethernet ports, the minimum frame size is 64 bytes, and the maximum frame size is programmable per port with a default of 15 KB. Any frame larger than 15 KB at reception is truncated to around 15 KB, and is flagged as erroneous, while the remainder of the frame is discarded. Any frame smaller than 64 bytes is flagged as erroneous regardless of whether the CRC is good or bad (runt frame). The next frame following a runt frame might be discarded as well if it follows the runt frame too closely.

The FM10000 is designed to support line rate operation for 64 bytes, provided the average inter-frame gap is 12 bytes or more. This is 14.88 Mp/s for 10 GbE, and 148.8 Mp/s for 100 GbE. The FM10000 does not guarantee line rate operation for frames smaller than 64 bytes.

For PCIe interfaces, the minimum frame size is 15 bytes, and the maximum is 15 KB. The FM10000 switch relies on the PCIe interface to filter out frames smaller than this minimum size. The switch is not line rated for tiny packets from PCIe interfaces, but the packet interface between the switch and the PCIe is a full handshake interface, and no packets are lost regardless of packet rate. The FM10000 supports up to 50 GbE of bandwidth per host interface in 8-lane PCIe Gen3 mode, for frames of at least 256 bytes.

NOTE: This page intentionally left blank.

3.0 Pin Descriptions

The pins interfaces are shown in Figure 3-1.

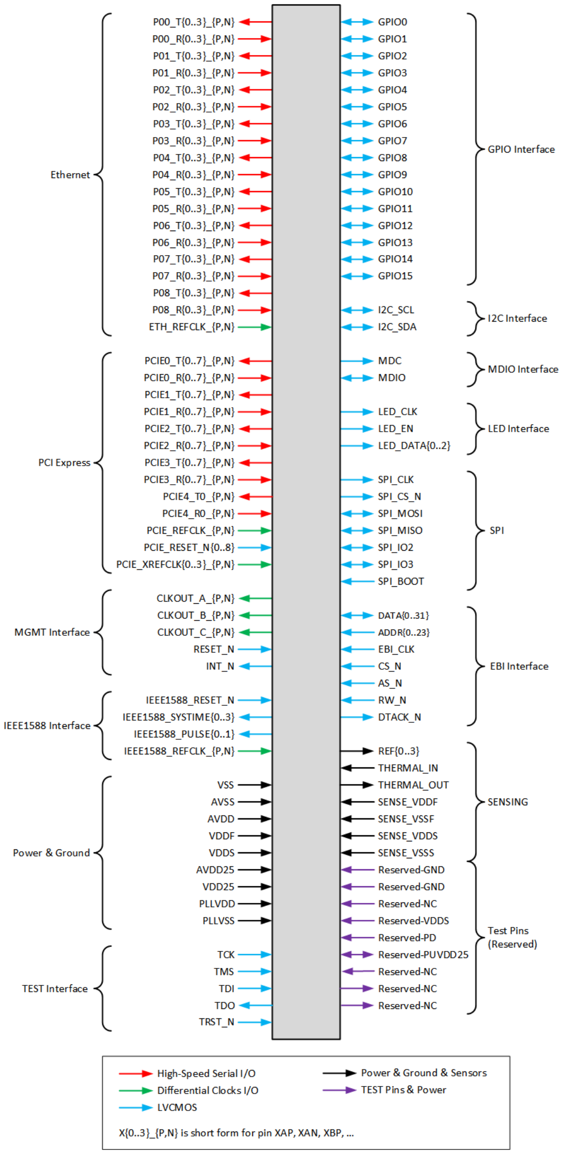


Figure 3-1 Pins Overview

3.1 Signal Name Convention

The following signal name convention is used:

- **Signal Mnemonic** — The signal mnemonic is a generic name used as a prefix to identify the function of this pin.
- **Negative Logic** — Signals with inverted logic (0b=asserted, 1b=de-asserted) are designated by using the signal mnemonic followed by "_N" suffix.
- **Differential Pairs** — Differential signals are designated by using the signal mnemonic followed by "_P" for the positive pin and "_N" for the negative pin.
- **Set** — A set of signals is referenced globally using name{M...N} or name{A,B,C,D}. The actual pin designation is formed by using the signal mnemonic and concatenating one of the values of the set. As an example, the pin set PCIE_REFCLK_{P,N} represents 2 pins designated PCIE_REFCLK_P and PCIE_REFCLK_N, and DATA{0..31} represent 32 pins designated DATA0, DATA1, ..., DATA31.
- **Direction** — The following pin direction types are used:
  - IN: Input to the chip
  - OUT: Output from the chip
  - IN/OUT: May operate as input or output
  - OD: Open drain output
  - OC: Open collector output
  - Power: A pin used for power
  - Ground: A pin used for ground
  - Sense: A pin used for sensing (no input/output concept)
- **Standard** — The following pin input/output buffer types are used:
  - CML
  - LVCMOS

3.2 Detailed Pin Descriptions

3.2.1 Ethernet Port Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
P{0..8}_R{0..3}_{P,N}	In	CML	SerDes receive ports organized in a group of four SerDes per EPL. Unused pins may be left unconnected.	Terminated	Terminated
P{0..8}_T{0..3}_{P,N}	Out	CML	SerDes transmit ports, organized in a group of four pairs per port. Each set {A,B,C,D} of four SerDes per EPL. Unused pins may be left unconnected.	Terminated	Terminated
ETH_REFCLK_{P,N)	In	LVPECL	Ethernet reference clock. 156.25 MHz. Must be supplied.	See LED_DATA[0]	See LED_DATA[0]

3.2.2 PCIe Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1

PCIE{0..3}_R{0..7}_{P,N}
PCIE4_R0_{P,N}
	In	CML	PCIe Receive Lanes.
			There are five PCI Express host interfaces, The first four have 8 lanes, while last one has one lane. Unused pins may be left unconnected.
				Terminated	Terminated

PCIE{0..3]_T{0..7}_{P,N}
PCIE4_T0_{P,N}
	Out	CML	PCIe Transmit Lanes.
			There are five PCI Express host interfaces. The first four have 8 lanes, while last one has one lane. Unused pins may be left unconnected.
				Terminated	Terminated

PCIE_REFCLK_{P,N}	In	LVPECL	PCIe reference clock. 100 MHz. Must be supplied.	See LED_DATA[1]	See LED_DATA[1]
PCIE_XREFCLK{0..3}_{P,N}	In	LVPECL	PCI Express extra reference clocks, 100 MHz. Optional. Connect to ground if not used.	Terminated	Termination controlled by software
PCIE_RESET_N{0..8}	In	LVCMOS	PCIe Reset Pin. This pin is per host and provides same function as pin PERST# defined in PCIe Specification. Must be driven. Connect to a pull down if corresponding host connection is never used. For example, if only x8 mode is used, connect the odd-numbered PCIE_RESET_N corresponding to that x8 to a pull-down to ground.	Floating	Floating

3.2.3 General Purpose Input/Output Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
GPIO{0..15}	In/Out	LVCMOS	General purpose I/O pin. Connect to a pull-up or pull-down as desired per system board requirements.	Floating	Floating

3.2.4 Serial Peripheral Interface

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
SPI_CLK	Out	LVCMOS	SPI Clock.	Floating	Floating unless enabled
SPI_CS_N	Out	LVCMOS	SPI Chip Select.	Floating	Floating unless enabled
SPI_MOSI	Out	LVCMOS	SPI Serial Data Out.	Floating	Floating unless enabled
SPI_MISO	In	LVCMOS	SPI Serial Data In. Connect to a pull-up to VDD25.	Floating	Floating
SPI_IO2	In/Out	LVCMOS	Used as IO[2] for QuadSPI memory. Connect to a pull-up to VDD25.	Floating	Floating
SPI_IO3	In/Out	LVCMOS	Used as IO[3] for QuadSPI memory. Connect to a pull-up to VDD25.	Floating	Floating
SPI_BOOT	Input	LVCMOS	Defines boot option. Must be supplied Connecting to a pull-up to VDD25 indicates that the device recovers its initial configuration from the attached SPI memory (required for PCIe operation). If pull-down to GND, indicates that the device wait for an external CPU to initialize the chip via I^{2}C or EBI.	Floating	Floating

3.2.5 I^{2}C Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
I2C_SCL	Open-drain	LVCMOS	I^{2}C Clock. Connect to a pull-up to VDD25.	Floating	Floating
I2C_SDA	Open-drain	LVCMOS	I^{2}C data. Connect to a pull-up to VDD25.	Floating	Floating

3.2.6 MDIO Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
MDC	Out	LVCMOS	MDIO clock.	Floating	Active
MDIO	Open-drain	LVCMOS	MDIO data. Connect to a pull-up to VDD25.	Floating	Floating

3.2.7 LED Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
LED_CLK	Out	LVCMOS	Serial LED clock.	Floating	Active
LED_EN	Out	LVCMOS	Serial LED enable. Asserted only on the first bit of an 80-bit frame.	Floating	Active
LED_DATA{0..2}	Out	LVCMOS	Serial LED data.	Weak pull-down	Floating until enabled

The following pins are latched at power-up to control internal termination on the 2.5 V LVPECL reference clocks inputs. Using a pull-down or leaving them to float enables the termination., while using a pull-up disables the termination.

- LED_DATA[0] — Controls internal termination for ETH_REFCLK input
- LED_DATA[1] — Controls internal termination for PCIE_REFCLK input
- LED_DATA[2] — Controls internal termination for IEEE1588_REFCLK inputs

3.2.8 JTAG Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
TCK	In	LVCMOS	JTAG clock. Leave unconnected if not used.	Internal pull-up	Internal pull-up
TMS	In	LVCMOS	JTAG control. Leave unconnected if not used.	Internal pull-up	Internal pull-up
TDI	In	LVCMOS	JTAG data input. Leave unconnected if not used.	Internal pull-up	Internal pull-up
TDO	Out	LVCMOS	JTAG data output. Leave unconnected if not used.	Floating	Active if ATPG mode
TRST_N	In	LVCMOS	JTAG reset. Pull down to ground if not using JTAG.	Internal pull-up	Internal pull-up

3.2.9 External Bus Interface Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
EBI_CLK	In	LVCMOS	Clock for EBI. Max rate is 33 MHz. Leave unconnected if not used.	Internal pull-down	Internal pull-down
DATA{0..31}	In/Out	LVCMOS	Data for EBI. Leave unconnected if not used.	Internal pull-up	Internal pull-up
ADDR{0..23}	In	LVCMOS	Register address to access. Leave unconnected if not used.	Internal pull-up	Internal pull-up
CS_N	In	LVCMOS	Chip select for register access. Leave unconnected if not used.	Internal pull-up	Internal pull-up
AS_N	In	LVCMOS	Address valid for EBI Leave unconnected if not used.	Internal pull-up	Internal pull-up
RW_N	In	LVCMOS	Indicates type of access (READ if high, WRITE if low). Leave unconnected if not used.	Internal pull-up	Internal pull-up
DTACK_N	Open-drain	LVCMOS	Indicates when data is valid (read) or when data has been captured (write). Connect to pull-up to VDD25. Leave unconnected if not used.	Internal pull-up	Internal pull-up

3.2.10 Miscellaneous Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
RESET_N	In	LVCMOS	General chip reset. Must be driven.	Internal pull-up	Internal pull-up
CLKOUT_{A,B,C}_{P,N}	Out	LVPECL	Clock recovery from SerDes. Leave unconnected if not used.	Floating	Active
INT_N	Out	LVCMOS	Interrupt. This pin is active low and asserted whenever an interrupt condition exists in the chip and the INT_N is configured to report it. The pin tri-states when not asserted, and thus requires an external pull-up to VDD25.	Floating	Floating/ Asserted

3.2.11 IEEE1588 Interface Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
IEEE1588_REFCLK_{P,N}	In	LVPECL	Reference clock for 1588. Optional. Range 66.666-250 MHz. Must be less than 50 ppm to a an integral period of 4-15 ns (for example, less than 50 ppm to a 10 ns period would be a 100.000 MHz clock). Can be left unconnected if not used.	See LED_DATA[2]	See LED_DATA[2]
IEEE1588_RESET_N	In	LVCMOS	Reset 1588 clock to system time 0 when asserted. Must be driven.	Floating	Floating
IEEE1588_SYSTIME{0..3}	Out	LVCMOS	System time tracker: • IEEE1588_SYSTEM[0] = SYSTEM.CLOCK0 • IEEE1588_SYSTEM[1] = SYSTEM.CLOCK1 • IEEE1588_SYSTEM[2] = SYSTEM.PLUS1 • IEEE1588_SYSTEM[3] = SYSTEM.MINUS1	Floating	Active
IEEE1588_PULSE{0..1}	Out	LVCMOS	Programmable periodic pulses out.	Floating	Active

3.2.12 Power Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
VDDF	N/A	Power	Core power supplies for Ethernet Switch function.	N/A	N/A
VDDS	N/A	Power	Core power supplies for interface blocks.	N/A	N/A
VDD25	N/A	Power	I/O power supply.	N/A	N/A
AVDD25	N/A	Power	LVPECL Reference Clocks power supply.	N/A	N/A
AVDD	N/A	Power	SerDes analog power supply.	N/A	N/A
PLLVDD{0,1}	N/A	Power	PLL analog power supply.	N/A	N/A
VSS	N/A	Ground	Ground.	N/A	N/A
AVSS	N/A	Analog Ground	Ground.	N/A	N/A
PLLVSS{0,1}	N/A	Analog Ground	PLL analog ground.	N/A	N/A

3.2.13 Reserved Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
Reserved-GND	Reserved	Ground	Reserved. Tie to ground for normal operation. Permanent damage will occur if this pin is tied to a positive potential.	Floating	Floating
Reserved-PD	In	LVCMOS	Reserved. Pull down to ground.	Floating	Floating
Reserved-PUVDD25	In	LVCMOS	Reserved. Pull up to VDD25.	Floating	Floating
Reserved-NC	In	LVCMOS	Reserved. Pull down to ground or leave unconnected.	Internal pull-down	Internal pull-down
Reserved-NC	Reserved	N/A	Reserved. Leave unconnected for normal usage.	N/A	N/A
Reserved-VDDS	N/A	Power	Reserved. Connect to VDDS.	N/A	N/A
Reserved-NC	Out	HSTL	Reserved. Leave unconnected for normal usage.	Floating	Active
Reserved-GND	Reserved	Ground	Reserved. Connect to ground for normal usage.	Floating	Floating

3.2.14 Sensing/Reference Pins

Pin Name	Direction	Type	Usage	Pin Default State	
				RESET_N=0	RESET_N=1
SENSE_VDDF SENSE_VSSF	N/A	Sense	VDDF power differential sensing line.	N/A	N/A
SENSE_VDDS SENSE_VSSS	N/A	Sense	VDDS power differential sensing line.	N/A	N/A
THERMAL_IN THERMAL_OUT	N/A	Sense	Thermal diode differential sensing line.	N/A	N/A
REF{0..3}	N/A	Reference	Connect to a 402 ohms (+/- 1%) pull down resistor to ground.	N/A	N/A

4.0 Reset, Boot and Clocking

4.1 Reset and Boot

This section discusses the detailed sequencing of how power and clocking signals must be applied to bring the FM10000 out of reset and how to boot the device.

The FM10000 manifests as up to 9 PCIe endpoints. The reset sequence is compatible with the *PCI Express® Base Specification, Revision 3.0* and the *PCI Express® Card Electromechanical Specification, Revision 3.0*.

4.1.1 Types of Reset

The FM10000 has several types of reset:

- Cold Reset
- PCIe Fundamental Reset
- PCIe Hot reset
- PCIe Targeted function reset(s)
- Switch and port reset(s)

Each of these have differing effects and implementations as described below.

4.1.1.1 Cold Reset

The initial boot when the power supplies are energized is facilitated by the Powergood mechanism. The voltage sources from all platform power supplies are routed to a system component which tracks them as they ramp-up, asserting platform powergood (PWROK) a fixed interval after the last voltage reference has stabilized.

When the FM10000 is installed on a motherboard (or a custom mezzanine card), the platform PWROK is connected to the RESET_N pin of the FM10000.

When the FM10000 is integrated on a PCIe add-in card, the PWROK signal must be generated locally on the card and then connected to the RESET_N pin of the FM10000.

4.1.1.2 PCIe Fundamental Reset

The FM10000 supports up to 9 PCIe endpoints with a dedicated reset pin for each. A given processor subsystem could initiate a fundamental reset using its PERST# pin (also called PLTRST#) at various S-state wake events (including normal startup) or other reset signal assertions. This PERST# pin is expected to connect externally to the corresponding PCIE_RESET_N[8:0] of the FM10000 for this end point.

The PCIe Fundamental reset only affects the targeted end-point when asserted. The switch, ports and other end-points are designed to continue to operate normally even when a particular PCIe end point is in reset.

De-assertion of any PCIE_RESET_N is an indication to the device that the PCIE_REFCLK is valid. The device does wait for both a RESET_N de-assertion and at least one PCIE_RESET_N[8:0] de-assertion before starting any boot sequence. Once started, the device considers PCIE_REFCLK valid even if all PCIE_RESET_N[8:0] get asserted in the future.

Note: The *PCI Express® Base Specification, Revision 3.0* defines "cold reset" to be a Fundamental Reset following the application of power while a "warm reset" is a Fundamental Reset without cycling the supplied power. Note that the PCIe Base spec defined terms are different than that defined for the FM10000.

4.1.1.3 PCIe Hot Reset

A PCIe hot reset may be detected via a PCIe in-band message on any of the 9 PCIe endpoints. This is an endpoint reset and therefore resets the entire endpoint, but effect is limited to that endpoint. This has the same effect as a PCIe fundamental reset. A temporary link down has the same effect.

4.1.1.4 PCIe Function Level Reset (FLR)

PCIe Function Level Resets or device disables are initiated by software via CSR write. In this case, an individual function is reset, not the entire PCIe endpoint. The endpoint recognizes the request for a reset, performs any necessary housekeeping, and then requests a function level reset from the endpoint's associated Virtual Function or Physical Function.

The processing of Function Level Reset is detailed in Section 6.5.

4.1.1.5 Switch Reset

The Switch reset could be initiated by software via a CSR write into the SOFT_RESET register located in the device management address space. This signal is internal to the device and is named SWITCH_RESET.

The switch can be placed in reset at any time, the signal SWITCH_RESET is sent to all modules connected to the switch (PCIe, EPLs). Those devices terminate any packets in flight from switch with a framing error code, and then drains any packets going to the switch. Except for packet draining, both the PCIe and Ethernet ports remains active while the switch is reset.

The SWITCH_RESET signal is asserted after a Cold Reset and remains asserted until the boot sequence de-asserts the signal.

When de-asserted, SWITCH_RESET must remain de-asserted for a minimum of 100 ns.

4.1.1.6 Ethernet Ports Logic Reset

The Ethernet ports logic could be reset by software via a CSR write into the SOFT_RESET register located in the device management address space. This single signal, named EPL_RESET, is internal to the device and distributed to all Ethernet ports.

The Ethernet ports can be placed in reset at any time provided the switch is in reset while the EPLs are reset. Note that all ports are affected. This signal is automatically asserted after a Cold Reset and remains asserted until the boot sequence de-asserts the signal, normally after having downloaded the Ethernet SerDes firmware.

When de-asserted, EPL_RESET must remain de-asserted for a minimum of 100 ns.

Note: The EPLs have registers sets to control power and state of each port. Therefore, re-assertion of a global EPL_RESET is not expected to ever be required once the initial boot sequence is completed.

4.1.2 Reset Distribution

The RESET_N input is a power good indicator that is intended to directly or indirectly reset most circuitry and flip-flop based registers, sticky or not, to their default values. SRAM and TCAM values are undefined after reset. The RESET_N is first sent to the watchdog circuit along with a CRM_FATAL_ERROR and a ORed of all PCIE_RESET_N.

The watchdog circuit asserts a MASTER_RESET, which gets into the Management Reset Domain logic to be distributed to all other blocks. Reset logic requires the ETH_REFCLK to be present. Therefore, RESET_N is not de-asserted until the clock is stable.

The Management Reset Domain Logic generates the following internal resets:

- 1 x Internal Management Reset (MASTER_RESET).
- 1 x Internal Cold Reset (COLD_RESET).
- 1 x Internal Ethernet Port Logic Reset (EPL_RESET).
- 1 x Internal Ethernet Switch Reset (SWITCH_RESET).
- 1 x Internal Ethernet Switch Ready (SWITCH_READY).
- 9 x Internal PCIe End Points Reset (PCIE_WARM_RESET).

The distribution is shown in Figure 4-1.

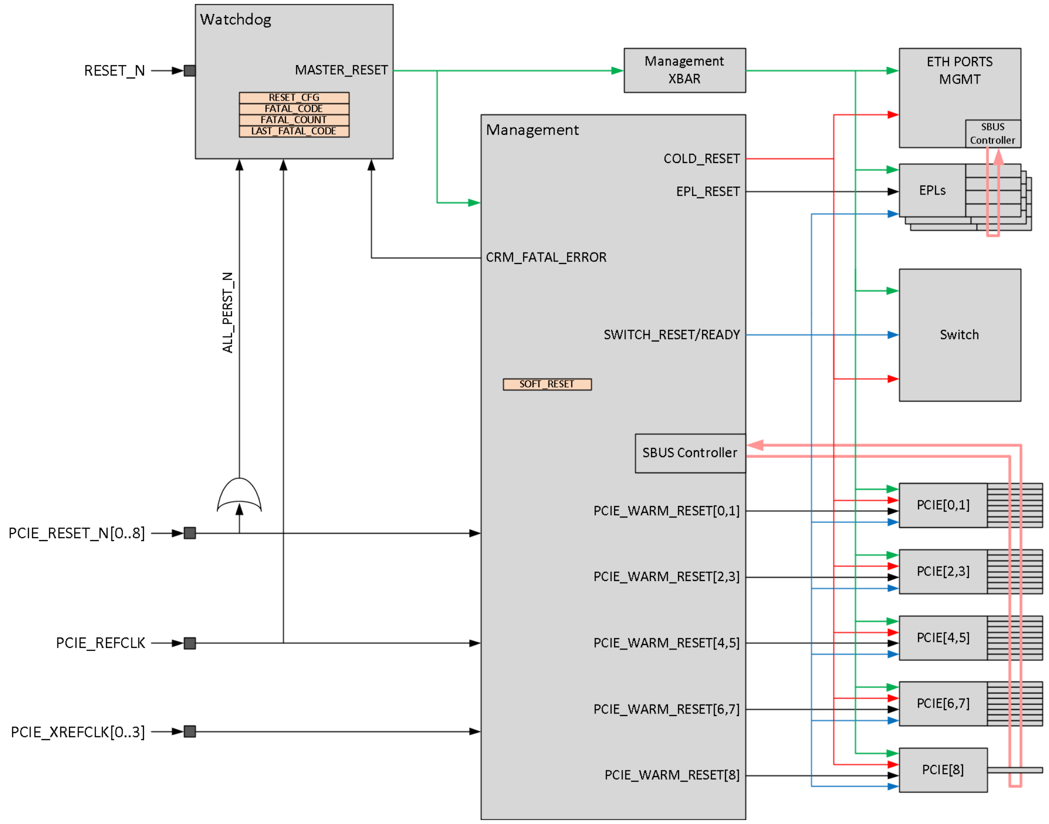


Figure 4-1 Reset Distribution

The signals details are listed in Table 4-1.

Table 4-1 Reset Signal Details

Reset	Color	Description
MASTER_RESET	Green	Asserted to indicate a full device reset. Includes reset of management crossbar and ripples to all modules through the management interface channels.
COLD_RESET	Red	Asserted jointly with MASTER_RESET, de-asserted when internal clocks are valid.
EPL_RESET	Black	Asserted to reset Ethernet ports.
SWITCH_RESET	Blue	Asserted to indicate a switch logic reset.
SWITCH_READY	Blue	Asserted to indicate switch ready for sending/receiving frames.
PCIE_WARM_RESET	Black	Asserted to indicate an individual end-point reset
CRM_FATAL_ERROR	Black	Indication of a fatal event detected by CRM
PCIE_XREFCLK	Black	Extra reference clocks (1 per host). Those clocks can be monitored and cause a PCIE_WARM_RESET to be asserted.

Note: The Tunneling Engines are part of the switch reset domain.

4.1.3 Power Sequencing and Reset Sequence

4.1.3.1 Power Sequence

The FM10000 supports the following power domains:

- VDDS (power for fabric)
- VDDF (power for PEP, EPLs, tunnels and management)
- AVDD (analog power for SerDes)
- VDDPLL (quiet power for PLLs)
- AVDD25 (analog power for LVCMOS I/Os)
- VDD25 (digital power for LVCMOS I/Os)

The power sequencing constraints are detailed in Section 12.0, "Electrical Specification".

The RESET_N is de-asserted when all power domains required are stable.

4.1.3.2 Reset Timing Diagram

The FM10000 waits only for a RESET_N de-assertion before starting the boot sequence. The state of PCIE_RESET_N[8:0] pins is ignored for the purpose of booting and running the device. The PCIE_RESET_N[8:0] are still used to control the state of the warm reset on the respective PEPs.

The de-assertion of RESET_N is assumed to indicate that both ETH_REFCLK and PCIE_REFCLK are valid. It is required for both clocks to be valid at least 100 µs before RESET_N is de-asserted, and must remain valid until RESET_N is asserted again. The FM10000 supports four additional clock references (PCIE_XREFCLK[0..3]), which can be optionally used by a host (or pair of hosts in x4 mode) to supply its own reference clock in either CC (Common Clock topology) mode with SSC enabled, or IR (Independent Reference topology) mode with SSC OFF.

Figure 4-2 shows the initial boot, up to and including PCIe end point initialization (init PCIe), which is then followed by the end point being taken out of reset following the PCIE_RESET_N[i]. The diagram also shows the timing if an additional PCIe reference clock is used for that host.

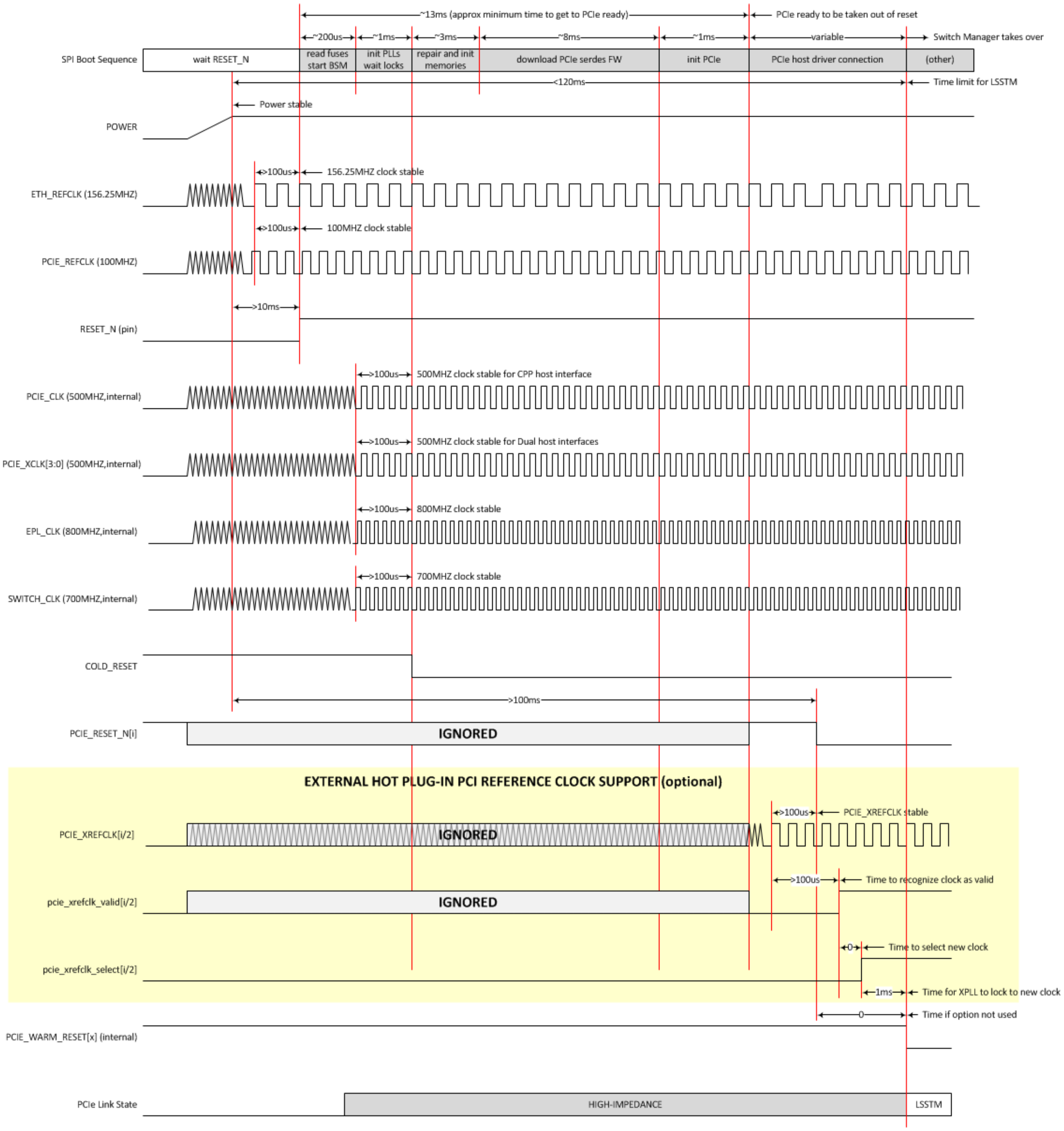


Figure 4-2 Management Reset Timing

The boot image in NVM can be used to wait and select an external clock coming from an attached host, or use the default (assumed always valid) PCIE_REFCLK.

The constraints are listed in Table 4-2.

Table 4-2 Management Reset Constraints

Reference	Constraint	Description
1	>10 ms	Power stable to RESET_N de-assertion.
2	>100 µs	ETH_REFCLK (156.25 MHz) stable to RESET_N de-assertion.
3	>100 µs	PCIE_REFCLK (100 MHz) stable to RESET_N de-assertion.
4	>100 µs	PCIE_XREFCLK (100 MHz) stable prior to PCIE_RESET_N de-assertion (if PCIE_XREFCLK used).
5	<20 ms	Time allowed from PCIE_RESET_N to LTSSM state.
6	>100 µs	Internal clocks stable to COLD_RESET de-assertion.
7	<120 ms	Time allowed from power good to LTSSM state for network interface card.

Note: The FM10000 has a minimum time of 200 µs from RESET_N to the first read of SPI Flash.

4.1.3.3 RESET_N Pin

The RESET_N pin places the watchdog in reset when asserted. The watchdog then automatically asserts the internal MASTER_RESET which cause entire management domain to be reset placing all modules into reset.

The watchdog module is taken out of reset immediately when RESET_N is de-asserted (power is good) and any of the nine PCIE_RESET_N is de-asserted (assume that PCIE_REFCLK is valid). The watchdog then releases the internal MASTER_RESET after few cycles. It also initializes the FATAL_COUNT and FATAL_CODE registers to 0.

4.1.3.4 FATAL_CODE Register

This is an internal Register in Watchdog. There are two methods to write into this register:

- via timeout of CRM access.
- via a direct write from any bus master to this register through the management infrastructure.

The CRM module can be programmed to create an alarm to the watchdog if a read or write access by the CRM module to a register exceed maximum time allowed. The amount of time is programmable in the CRM_CTRL register. This function could be disabled (and is disabled by default). For details, see CRM_CTRL.*TimeoutPrescale* (Table 9-1 or Section 11.25.2.58).

When the register is written, the watchdog copies FATAL_CODE into LAST_FATAL_CODE, clears FATAL_CODE, increment FATAL_COUNT by one and then waits N cycles before asserting MASTER_RESET for M cycles (both N & M are configurable in RESET_CFG register). The error code is:

- 1 = CRM FATAL ERROR
- 2..255 = Reserved for software

4.1.3.5 MASTER_RESET Signal

The MASTER_RESET is an output of the watchdog circuit. It causes the management module and management crossbar to be placed in reset when asserted. The management in turn asserts a COLD_RESET, which is distributed to all modules. The management also contains the SOFT_RESET register which controls the reset status of individual cores: PCIe interfaces, Switch, and EPL modules. The SOFT_RESET asserts a soft reset on all cores by default when MASTER_RESET is asserted.

When MASTER_RESET gets de-asserted, the management module boots the system by checking the status of the SPI_BOOT pin. If this pin is asserted, an initial boot sequence is recovered from the device from an attached SPI memory. Usage of SPI_BOOT is required if switch manager is located on a PCIe interface and recommended on all systems.

4.1.3.6 COLD_RESET Signal

The COLD_RESET signal is an internal signal generated by management to all other modules of the device. This internal signal is asserted whenever MASTER_RESET is asserted and remains asserted until PLL are configured and locked. The boot sequencer is responsible to program the PLL, wait for a lock, wait a period (see Timing (6) in Table 4-2), and then de-assert this signal to indicate to all cores that they now have a valid clock. The de-assertion is through the SOFT_RESET.*ColdReset* register.

4.1.3.7 SOFT_RESET Register

The SOFT_RESET register controls the global cold reset and the reset for different modules: EPLs, PCIe interfaces and Switch. Its default value is to assert reset on all modules and assert the global COLD_RESET. The COLD_RESET must be cleared first, then each module must be taken out of reset before it can be accessed.

4.1.3.8 PCIE_RESET_N[0..8] Pins

The PCIE_RESET_N[0..8] pins are used as per PCI-Express specification to command a warm reset of the corresponding PCIe module. It does not cause the device to reset or the switch function to reset; only the PCIe end-point module internal local (including physical function) is reset.

The device assumes the PCIE_REFCLK becomes valid when any of the PCIE_RESET_N[0..8] is de-asserted.

4.1.3.9 PCIE_WARM_RESET[0..8] Signal

The PCIE_WARM_RESET signals are internal signals generated by management, one per PCIe end point, to perform a PCIe fundamental reset on each specific PCIe unit. These signals are controlled from the following states, in priority order:

State	Priority	Description
DEVICE_CFG.*FeatureCode*	1	Defines number of PCIe enabled in this device. Disabled PCIe end points are maintained in warm reset at all times, regardless of the next conditions. For further details, see DEVICE_CFG.*FeatureCode* in Section 11.25.2.5.
SOFT_RESET.*PCIeReset*[0..8]	2	Setting any bit to 1b causes the corresponding PCIe end point to be placed in warm reset. For further details, see SOFT_RESET.*PCIeReset*[0..8] in Section 11.25.2.4.
SOFT_RESET.*PCIeActive*[0..8]	3	Setting any bit to 1b will causes the PCIe to be placed out of reset, overriding the PCIE_RESET_N pin state. For further details, see SOFT_RESET.*PCIeActive*[0..8] in Section 11.25.2.4.
PCIE_RESET_N[0..8] pins	4	The PCIe warm reset follows this pin if all other previous conditions are not active.

4.1.3.10 SWITCH_RESET and SWITCH_READY Signals

The SWITCH_RESET and SWITCH_READY are signals generated to reset the Ethernet switch in an orderly manner. The SWITCH_RESET is immediately asserted whenever the MASTER_RESET is asserted. The SWITCH_RESET and SWITCH_READY can be asserted or de-asserted using the SOFT_RESET register; the SOFT_RESET contains one bit to control each of those signals. However, software must proceed in order, and only the following sequences are valid:

To place the switch in reset by software:

1. SOFT_RESET.*SwitchReady* = 0b
2. Wait 100 µs
3. SOFT_RESET.*SwitchReset* = 1b

To return the switch to normal state:

1. SOFT_RESET.*SwitchReset* = 0b
2. Program the switch (level of setting could vary)
3. SOFT_RESET.*SwitchReady* = 1b

The SWITCH_RESET and SWITCH_READY signals are distributed to PCIe, EPLs and tunneling engines as these modules must be aware when the switch is reset.

4.1.3.11 Initialization Sequence

The sequence of events is enumerated in Table 4-3.

Table 4-3 Initialization Sequence of Events

Step	Action	Time Allowed	Details
System control:			
1	System asserts external resets.	N/A	System asserts RESET_N and PCIE_RESET_N{0..8}. Also, it must pull the TRST_N and TEST_MODE pins low and the TEST_CTRL{0..2} pins high.
2	System applies power.	N/A	Device asserts MASTER_RESET, COLD_RESET, PCIE_WARM_RESET[0..8], EPL_RESET, SWITCH_RESET and clears SWITCH_READY.
3	System provides refclks.	100 µs	ETH_REFCLK (156.25 MHz) and PCIE_REFCLK (100 MHz) must be provided.
4	System de-asserts external resets.	N/A	System de-asserts RESET_N when power and PCIE_REFCLK/ETH_REFCLK are stable for at least 100 µs. System de-asserts one of PCIE_RESET_N[x] when PCIE_REFCLK is stable for at least 100 µs, and at least one host is ready to start. De-assertion of RESET_N and PCIE_RESET_N can come in either order.
Automatic:			
5	The FM10000 de-asserts MASTER_RESET.	10 µs	
6	The FM10000 reads fusebox.	200 µs	The boot sequencer recovers the fusebox content for memory repair and SKU selection.
7	The FM10000 starts boot sequence.	10 µs	The FM10000 reads the SPI_BOOT pin to detect if booting from SPI-attached memory is enabled, and if yes, starts to execute a boot sequence stored in SPI memory (recommended method). If SPI_BOOT is not asserted, the self-boot stops and the device waits for an access from an external processor through the EBI bus to do the boot sequence.
Boot commands read from SPI-attached memory:			
8	Start clocks.	1 ms	1. Select the desired fabric PLL frequency, via PLL_FABRIC_CTRL or PLL_FABRIC_LOCK.*FreqSel*. For lower-power application, consider slowing down the clock temporarily: • Set PLL_PCIE_CTRL.OutDiv = 40 • Toggle to apply the new divider: – PLL_PCIE_CTRL.*MiscCtrl*[4] = 1 – PLL_PCIE_CTRL.*MiscCtrl*[4] = 0 2. Toggle PLL_FABRIC_CTRL.*Nreset* (set *Nreset* = 0b; wait 500 ns; set *Nreset* = 1b). 3. Wait 1.0 ms for all PLLs to lock. After the delay, all the PLL_XXX_STAT.*PllLocked* are expected be set (but do not base the delay on polling these bits). • The FM10000 has three mandatory PLLs (FABRIC, PCIE, and ETH). There are four additional PLLs (locks are in PCIE_CLK_STAT.*PllLocked*). The last four can be disabled if they are not used. 4. Set PLL_XXX_CTRL.*OutMuxSel* = 1. • Also, set PCIE_CLK_CTRL.*OutMuxSel*[interface] = 1 for the interfaces receiving their own clocks. 5. Set DEVICE_CFG.*SystimeClockSource*.
9	Configure PCIe mode.	10 µs	Set DEVICE_CFG.PCIeMode and PCIeEnable. For lower-power applications, disable all PEPs temporarily by setting DEVICE_CFG.*PCIenable*[0..7] = 0.

Table 4-3 Initialization Sequence of Events [Continued]

Step	Action	Time Allowed	Details
10	De-assert COLD_RESET.	10 µs	Set SOFT_RESET.*ColdReset* = 0. If the clock was slowed down in step 8.1, return to the normal clock rate: • Set PLL_PCIE_CTRL.*OutDiv* = 10 • Toggle to apply the new divider: — PLL_PCIE_CTRL.*MiscCtrl*[4] = 1 — PLL_PCIE_CTRL.*MiscCtrl*[4] = 0 • Activate only the required PEPs (DEVICE_CFG.*PCIenable*[0..8] = xxx).
11	Start SBuses.	100 µs	Set SBUS_PCIE_CFG.*SBUS_ControllerReset* = 0 and SBUS_EPL_CFG.*SBUS_ControllerReset* = 0.
12	Repair memories.	1.6 ms	Memory defects are repaired using data from the fusebox. The steps are: 1. Set REI_CTRL.*AutoLoadEnable* = 1b. 2. Set REI_CTRL.*Reset* = 0b. 3. Poll REI_STAT until either Pass or Fail is detected. Failure indicates a device problem.
13	Initialize memories.	0.8 ms	Initialize large internal memories by activating the BIST engine in each memory with a memory fill command. The initialization is done through the following steps: 1. Set BIST_CTRL.*BistMode_XXX* = 1b (all BistMode bits can be set in parallel, to initialize all memories at once). 2. Set BIST_CTRL.*BistRun_XXX* = 1b (this must be sequenced after setting the BistMode bits). 3. Wait 0.8 ms. 4. Set BIST_CTRL.*BistRun_XXX* = 0b. 5. Set BIST_CTRL.*BistMode_XXX* = 0b. ***Note:*** Only initialize the memories for units that are not under reset.
14	Disable PCIe interrupts	<10 µs	Set PCIE_SERDES_CTRL[i,0..7].*Enable* = 1b on all enabled PEPs (those with the corresponding bit DEVICE_CFG.*PCIeEnable*[i] set).
15	Program PCIe SerDes.	8 ms	The boot sequencer downloads the firmware to the PCIe Master SPICO and the PCIe SerDes and takes the SerDes out of reset. The download procedure is detailed in Section 9.3.4, "Firmware Download".
16	Initialize PCIe SerDes.	100 µs	The PCIe SerDes are given their refclk ratio and initial equalization parameters, as detailed in Section 9.3.5, "PCIe SerDes Initialization".
17	Enable PCIe PCS interrupts	10 µs	Set PCIE_SERDES_CTRL[i,0..7].*Enable* = 0b on all enabled PEPs.
18	Initialize PCIe end points.	1 ms	The boot sequencer sets SOFT_RESET.*PCIeReset*[i] = 0b on all enabled PEPs, to allow them to come out of reset. Only those that have PCIE_RESET_N[i] de-asserted are actually taken out of reset; this is determined by checking PINS_STAT.*PCIE_RESET_N*[i]. For these, the boot sequencer initializes the PEP as detailed in Section 6.16, "Initialization After Reset". PCIe primary setup now completed, the Switch Manager and Host Drivers could take over and complete initialization of the device, which includes the Ethernet Ports and the Shared Memory Switch.
Switch Manager control:			
19	De-assert SWITCH_RESET.		Set SOFT_RESET.*SwitchReset* = 0b.
20	Disable switch scan.		Write SCAN_DATA_IN with *UpdateNodes* (bit 27) and *Passthru* (bit 30) set to 1b. All other bits = 0 (value of 0x4800000). This disables the scan switch chain, which is active by default.
21	Disable switch loopbacks.		Disable the parallel loopbacks around the switch (refer to Section 10.2.1.2): 1. Set PCIE_CTRL_EXT[i].*SwitchLoopback* = 0b on all enabled PEPs. 2. Set EPL_CFG_A[i].*Active* = 0xF on all EPLs. 3. Set TE_CFG[i].*SwitchLoopbackDisable* = 1b on both TEs.

Table 4-3 Initialization Sequence of Events [Continued]

Step	Action	Time Allowed	Details
22	Initialize switch functions.		The necessary steps to initialize the desired switch/router functions depend on the application. However, the following registers should be configured even for basic L2 switching functions: 1. Configure CM_GLOBAL_CFG to enable Congestion Management. 2. Set CM_GLOBAL_WM as specified in Section 5.7.10.4, "Memory Usage Tracking and Watermarks Application". 3. Set MA_TCN_IM = 0, FH_TAIL_IM.TCN = 0, and INTERRUPT_MASK_XXX.FH_TAIL = 0b to receive interrupt notification of new source MAC addresses.
23	Initialize scheduler.		This is detailed in Section 5.4.3, "Scheduler Initialization".
24	Start LED controller.		Set LED_CFG.*LEDEnable* = 1b.
25	Initialize IEEE 1588 system time.		The configuration of IEEE 1588 system time is described in Section 9.9.1, "Initialization of System Time".
26	Assert SWITCH_READY.		Set SOFT_RESET.SwitchReady = 1b to allow traffic to flow into the switch.

Note: The recommended method on all systems is to use SPI memory boot to bring up the PCIe host interfaces. The EBI interface should not be used.

Note: Some device models are pre-programmed at the factory with reduced functionality. The controlled options are: clocks rate (hardware fix, software-programmable), 100 GbE/25 GbE operation (enabled/disabled) and numbers of PEPs. For PEPs restrictions the options are; (a) enabled all (0 through 8), (b) enabled up to five (0,1,2,3 and 8), (c) enable no more than one (0 or 8).

The boot sequence executed from SPI-attached memory is flexible. Refer to Section 9.7.5, "Boot State Machine" for details of the command encoding.

After boot is complete, the BSM can also be kept active as a system monitor, or configured to be woken on interrupts directed to it via the INTERRUPT_MASK_BSM register. Interrupts handled in the BSM could include initialization of PCIe endpoints, and VPD requests from those endpoints. For suggested flows, refer to Section 6.0, "PCIe Host Interface".

4.2 Base Clocking

The FM10000 clock distribution system is shown in Figure 4-3.

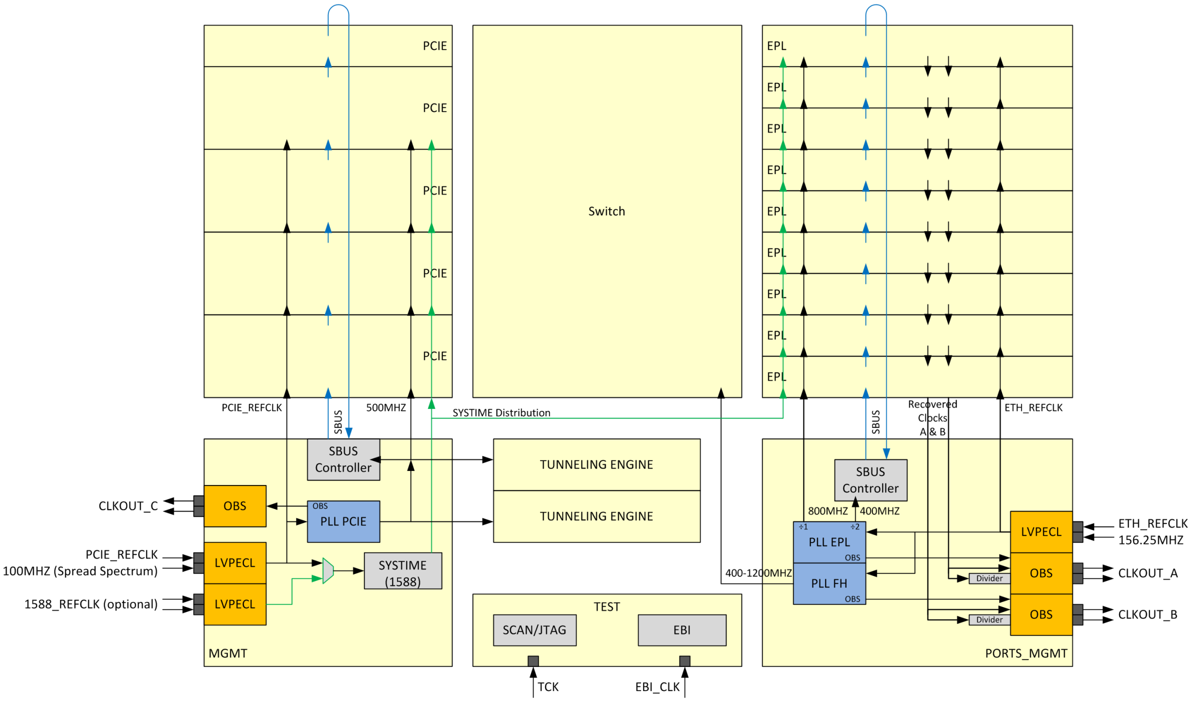


Figure 4-3 Clock Distribution

There are 5 input clocks:

Table 4-4 Input Clock Definitions

Name	Type	Speed (MHz)	Description
ETH_REFCLK	LVPECL	156.25	Ethernet reference clocks
PCIE_REFCLK	LVPECL	100	PCI-Express reference clock
1588_REFCLK	LVPECL	Variable	Reference clock (optional)
EBI_CLK	LVCMOS	33	External Bus Interface clock (optional) — Range = 66.666-250 MHz
TCK	LVCMOS	18 MHz	JTAG Clock (optional)

Both the ETH_REFCLK and PCIE_REFCLK are required.

The EBI clock is strictly used by the Test module and not used anywhere else. If the test module is not used, this clock does not need to be supplied.

The 1588 reference clock is an alternative clock source for the 1588 module. It is intended to be used as an alternative to the PCIE_REFCLK if the PCIE_REFCLK characteristics are inappropriate for a time reference.

The TCK clock is used for JTAG and only required when JTAG is used.

The switch has 3 programmable PLLs:

Table 4-5 Programmable PLLs

Name	Speed Range	Normal Speed	Description
PLL1	325 MHz - 800 MHz	800 MHz	Ethernet Port Logic clock
PLL2	300 MHz - 1.2 GHz	700 MHz	Ethernet Shared Memory Switch clock
PLL3	N/A	500 MHz	PCIe End Point clocks

All PLLs have default configurations suitable to produce the right frequency after reset and do not need to be re-configured. The default configuration for each can be overridden if desired using the PLL_xxx_CTRL registers. The multipliers and dividers for each PLL must be programmed such that the PLL operates within +/- 20% of its core frequency (10 GHz).

Note: The FM10000 is available in different SKUs listed in Section 1.2. The PLL_FABRIC_LOCK.*FeatureCode* indicates the possible fabric clocks for a given SKU, while the register PLL_FABRIC_LOCK.*FreqSel* allows selection of predefined values. The selectable values for the Fabric clocks for restricted SKUs are 600, 500, 400 and 300 MHz. For SKUs that are not fabric clock rate restricted, the software leaves the PLL_FABRIC_LOCK.*FreqSel* to its default value and programs the PLL_FABRIC_CTRL register to the desired fabric clock rate.

The clocks are defined as follows:

```
// xxx is PCIE, FABRIC or EPL

RefClk = (xxx == 'PCIE') ? 100MHz : 156.25MHz;
RefDiv = PLL_xxx_CTRL.RefDiv;
FbDiv4 = PLL_xxx_CTRL.FbDiv4;
FbDiv255 = PLL_xxx_CTRL.FbDiv255;
OutDiv = PLL_xxx_CTRL.OutDiv;

// PLL VCO frequency, must be between 8 and 12GHz.
VCO = RefClk/RefDiv * 4 * FbDiv255 * (1+FbDiv4 )

// Output frequency
OutClock = VCO/OutDiv/2
```

When using the PLL_xxx_CTRL registers to set a clock, the multipliers and dividers for each PLL must be selected such that the PLL operates within +/- 20% of its core frequency (10 GHz).

Table 4-6 shows the normal default values for the PLLs for SKUs that do not have restrictions.

Table 4-6 PLL Default Values

Parameters	ETH	FABRIC (700 MHz)	FABRIC (1 GHz)	PCIe	Comment
RefClk	156.25	156.25	156.25	100	
RefDiv	3	3	4	1	
FbDiv4	0	0	0	0	
FbDiv255	46	47	64	25	
FreqVCO (GHz)	9.583333333	9.791666667	10	10	Must be between 8 and 12.
OutDiv	6	7	5	10	
OutClock (MHz)	798.611111	699.404762	1000.000000	500.000000	

The FM10000 provides a few clock outputs:

- Three clock observation points:
  - CLKOUT_A, choice of:
    - 0 = Ethernet recovered clock "A"
    - 1 = Ethernet recovered clock "A" with divider (M/N/DIV)
    - 2 = EPL PLL/8
  - CLKOUT_B, choice of:
    - 0 = Ethernet recovered clock "B"
    - 1 = Ethernet recovered clock "B" with divider (M/N/DIV)
    - 2 = Fabric PLL/8
  - CLKOUT_C, choice of:
    - 0 = CLKOUT_C held low
    - 1 = PCIe PLL/8
  - For each of the CLKOUT pins, there is a final divider, with the option to divide by 1, 2, 3, or 8.
- Two periodic pulses from System Time (1588) module.

For Ethernet recovered clocks, the ETHCLK_CFG[0..1] and ETHCLK_RATIO[0..1] registers define the dividers and multipliers to apply to the recovered clocks "A" and "B". The selection of which SerDes is observed on "A" or "B" is defined in LANE_CFG[epl][lane] registers (refer to Section 7.0, "Ethernet Port Logic (EPL)").

The final output clock is:

```
// BitRate : SerDes bit rate (example:10.3125G)
M = ETHCLK_RATIO[i].M;
N = ETHCLK_RATIO[i].N;
D = ETHCLK_CFG[i].DIVIDER;
F = LANE_CFG[epl][lane].RefClock?Div;   // 2 or 4
if ( 1G or 2.5G )
    R = 10
else if ( 10G or 40G )
    R = 10
else // 25G or 100G
    R = 40

if ( divided output selected and divider enabled )
    CLKOUT = BitRate/R/F * (M/N) / DIVIDER;
else
    CLKOUT = BitRate/R/F;
```

4.3 Extended PCIe Clocking

4.3.1 Overview

The FM10000 uses a common 100 MHz reference clock for all four x8 host interface blocks. This reference clock must be good prior to RESET_N. Refer to Section 4.1.3.2 for more information.

Spread Spectrum Clocking (SSC) is a known method used across the industry to reduce Electro-Magnetic Interference (EMI) in electronic equipment and reduce overall shielding cost in enclosures. EMI specifications are defined by standard bodies such as the FCC and must be met before equipment can be sold. SSC consists of constantly varying the reference clock by up to [0,-0.5%] from nominal clock at a rate not exceeding [30-33 KHz], and instant variation not exceeding 1250 ppm/µs. However, in some multi-CPU systems, it is not always practical to deploy a single PCIe common reference clock.

The FM10000 adds four additional PCIe clock domains. Those additional clock inputs allow hosts to supply their own clock sources to the FM10000, and any one of those hosts could independently enable SSC even when each host provides a different clock. Each one of the new PCIe reference clocks can only be used for a given PCIe Host Interface, which can operate as a 1x8 or 2x4. Therefore, a pair of 2x4 hosts in one interface must use a single input clock. The FM10000 allows each x8 PCIe Host interface to use the default common PCIE_REFCLK or the corresponding PCIE_XREFCLK, but it cannot use one from some other PCIe Host interface. The last x1 PCIe Host Interface (for management) can only use the common PCIE_REFCLK.

The FM10000 also includes a clock monitoring circuit that could be enabled to provide safeguard against bad clocks or surprise clock disconnection. The next sections provide details on the clock monitoring circuit.

A typical system using multiple SSC domains is shown in Figure 4-4. The host modules supply the PCIe reference clocks to the FM10000, which could be selected by the FM10000 when the corresponding PCIE_RESET_N[i] is de-asserted.

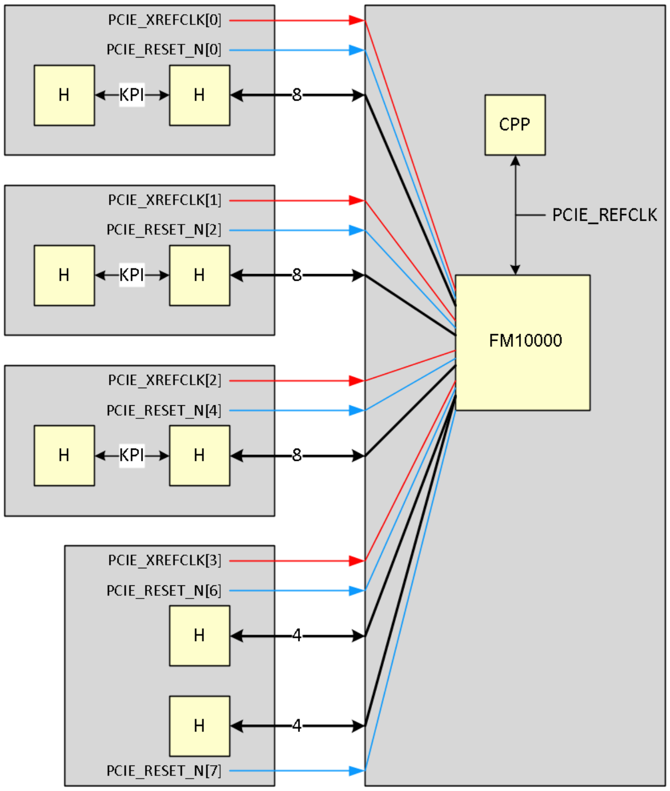


Figure 4-4 Extended Clocking Usage Example

The last module illustrates a case where the FM10000 is configured in 2x4 mode rather than 1x8. In this case, both x4 connections must share the same clock, but they can still be individually placed in and out of reset.

All PCIE reference clocks, the current mandatory PCIE_REFCLK, and the four PCIE_XREFCLK[0..3], can be in different Spread Spectrum domains corresponding to their PCIe end-points.

4.3.2 Clock Signals

Following are the PCIE_XREFCLK[0..3] clock inputs (of LVPECL I/O type) with their mappings:

- PCIE_XREFCLK[0] is for PEP[0] and PEP[1]
- PCIE_XREFCLK[1] is for PEP[2] and PEP[3]
- PCIE_XREFCLK[2] is for PEP[4] and PEP[5]
- PCIE_XREFCLK[3] is for PEP[6] and PEP[7]

Figure 4-5 shows the clock distribution block diagram and PCIE warm reset control.

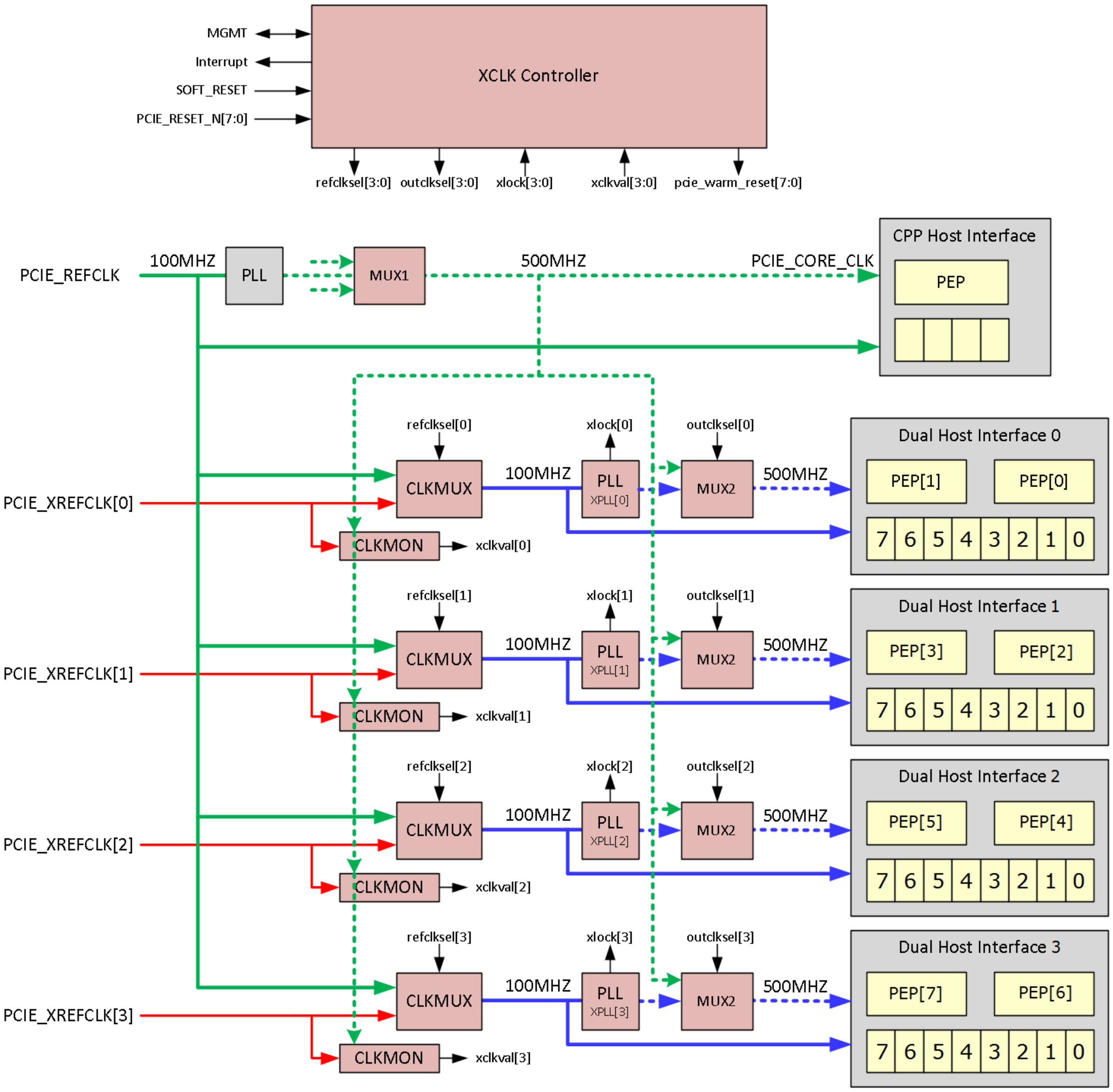


Figure 4-5 Extended PCIe Clocking Distribution

The following circuits are in the design:

- Four 500 MHz PLLs (XPLL[0..3])
  — Dedicated synthesizer per host interface. All four PLLs are enabled by default. The boot sequence can be programmed to disable unused XPLL to save power if it is known that the interface is never enabled, or the attached host never provides a clock.
- Four glitch-free CLKMUX
  — Clock multiplexers providing glitch-free transition from PCIE_XREFCLK and PCIE_REFCLK for each host interface. The circuit is glitch-free provided both clocks are valid, but switches anyway if the clock PCIE_XREFCLK is out of spec.
- Four CLKMON
  — Clock monitoring circuits reporting validity of each clock.

- Four non-glitch-free clock multiplexers on output of PLLs 500 MHz
    - — This multiplexer allows selecting between the per-interface dedicated XPLL output or the output of a higher level test multiplexer (which allows between a test clock, a 100 MHz clock, and the main PCIE PLL output). The multiplexer is controlled via PCIE_CLK_CTRL.*OutMuxSel* while the higher-level test multiplexer is controlled via PLL_PCIE_CTRL.*OutMuxSel*. The default is to use the higher-level test multiplexer output. The NVM image should program the PCIE_CLK_CTRL.*OutMuxSel* register to use the XPLL only if it is enabled and locked.
- Clock Controller
    - — Controls PLLs and warm resets.

4.3.3 Clock Monitor

Each clock monitor (CLKMON) monitors a given PCIE_XREFCLK and indicates if this clock is valid or not. The monitoring function offers two monitoring circuits:

- Check if each clock period is within coarse limits.
- Check if frequency within bound.

The clock is determined as valid if and only if those two circuits report the clock as valid. If any condition is not met, the XCLKVAL is immediately de-asserted.

Both circuits are configurable and operate as follows:

- Circuit “A”: period check
    - — The circuit uses the 500 MHz reference clock from main PLL to measure the clock period for a given PCIE_XREFCLK clock. The circuit has a counter which is restarted at each rising edge of the monitored clock and increments every 2 ns until it saturates (or a rising edge is detected). The circuit declares the clock invalid if the counter exceeds a programmable threshold (PCIE_CLKMON_DEADLINES_CFG.*SilentLimit*, default 10) indicating this clock has period in excess of the limit configured.
        - This circuit reacts quickly to faults (20 ns by default).
        - Setting the *SilentLimit* to 0 disables this check and the clock period is assumed good.
- Circuit “B”: frequency check
    - — The circuit implements a counter which is incremented by N (default 1) on rising edge of 500 MHz and decremented by D (default 5) on rising edge of the monitored clock (100 MHz). This counter is monitored over a programmable period P and is expected to be within bounds (-MIN,MAX) at all time during the monitored period. The clock is indicated as invalid as soon as the counter exceeds its bounds. The counter is reset to 0 at the end of the monitored period. Parameters are:
        - N = PCIE_CLKMON_RATIO_CFG.*Num*
        - D = PCIE_CLKMON_RATIO_CFG.*Denom*
        - P = PCIE_CLKMON_DEADLINES_CFG.*Sustain*
        - MIN = PCIE_CLKMON_TOLERANCE_CFG.*NegTol*
        - MAX = PCIE_CLKMON_TOLERANCE_CFG.*PosTol*

— The default period is 1000 cycles of 500 MHz (2 µs) and the default margins are (-10,+15), thus providing a check of +/- 1% (10/1000). This is slightly higher than the worst expected differences (0.56%), which is SSC variance (0.5%) plus 2x the clock precision (2x300 ppm = 0.06%).

Note: If multi-domain SSC is used, and it is known that the supplied XREFCLK can be off by up to 1% and still accepted, it is recommended to boost voltage by 1% to speed up the device to ensure no timing closure is encountered if the XREFCLK is actually close to the 1% limit without exceeding it.

— The circuit reacts rapidly to coarse clock discrepancies and slower to slight variations off the target.

— Setting the period P to 0 disables this check, and the clock frequency is assumed good.

Note: The clock monitoring circuit is designed to detect basic clock frequency problems. It does not catch all possible failure modes imaginable. For hot insertion of modules, it is preferable to always provide a pre-warning to the FM10000 to indicate that the card is about to be extracted. The recommended method is to assert PCIE_RESET_N early enough during extraction to allow the FM10000 to do a clean transition before the clock disappears or glitches. The clock monitoring circuit is designed to detect clock frequency problems for an input reference clock with a frequency up to 250 MHz, with at least 1 ns high. Behavior is not defined if a malicious host system provides a higher frequency refclk.

4.3.4 XCLK Controller

The XCLK controller is a module accessible by software, which manages the clock multiplexing individually for each host interface and associated PLLs. It also uses SOFT_RESET register settings and PCIE_RESET_N pins and clocking mode to control the warm reset on each PEP. The block diagram in Figure 4-6 shows the logic dependency:

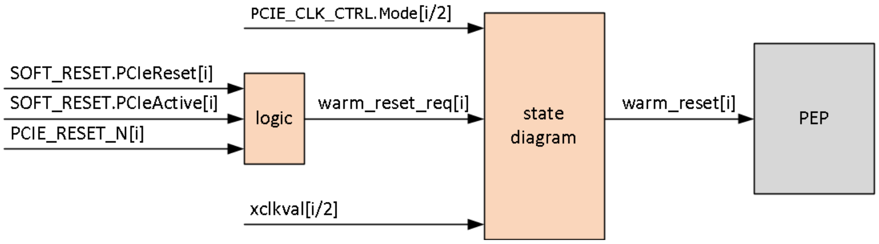


Figure 4-6 PCIe End Point Warm Reset Dependencies

The warm reset request is defined as:

```
warm_reset_request = SOFT_RESET.PCIeReset[i] ||
                     ~(PCIE_RESET_N[i] || SOFT_RESET.PCIeActive[i]);
```

The controller includes the following modes of operation per host interface (defined in PCIE_CLK_CTRL.*Mode*[0..3]).

Mode 0

Mode 0 forces the CLKMUX to always use the main PCIE_REFCLK (default) and set the PEP **warm_reset** equal to the **warm_reset_request**. In this mode, the PCIE_XREFCLK and **xclkval** are ignored.

Mode 1

Mode 1 forces the CLKMUX to always use the alternate PCIE_XREFCLK (ignoring if the clock is valid or the state of PCIE_RESET_N) and set the PEP **warm_reset** equal to the **warm_reset_request**. The PCIE_XREFCLK is always used, and the **xclkval** is ignored. This mode of operation is a test mode used to bypass the clock monitoring circuits and PCIE_RESET_N dependencies.

Mode 2

Mode 2 implements automatic reference clock selection based on PCIE_XREFCLK validity (**xclkval**) and PEP **warm_reset_request**. This is the mode normally selected when PCIE_XREFCLK domains are used in hot plug-in systems.

In this mode, the circuit selects PCIE_XREFCLK if and only if:

- the PCIE_XREFCLK is valid, AND
- one of the PEPs of the target host interface is requested to be taken out of reset (**warm_reset_request** = 0).

The actual warm reset de-assertion to the target PEP is delayed by a programmable amount of time (increments of 100 selected refclk cycles) in PCIE_WARM_RESET_DELAY. The delay must be set to a value higher than the time required for XPLL to lock (the lock signal from the XPLL is not used; only the specified delay is applied). The delay is intended to provide time for the XPLL to lock to the new clock.

There is only one timer per host interface, shared between the two PEPs of a host interface. If the two PEPs of an interface are requested out of reset (**warm_reset_request** = 0) at slightly different times, the timer is started by the first one's out-of-reset request. The earlier PEP requested out of reset is taken out of reset at the end of the programmed delay. If the later PEP's out-of-reset request occurs while the timer is counting down, it is taken out of reset at the end of the programmed delay. Otherwise, it is taken out of reset immediately after the out-of-reset request occurs.

If a PEP of a host interface is requested to be placed in warm reset, the PEP is placed in warm reset immediately (without delay). If both PEPs of an interface are requested to be in reset, the clock selection circuit reverts to PCIE_REFCLK.

Whenever the PCIE_XREFCLK clock is invalid, the clock selection circuit selects PCIE_REFCLK automatically and asserts the warm reset of both PEPs of the host interface.

The warm reset to each PEP remains asserted from any time PCIE_XREFCLK is invalid until **warm_reset_request** transitions from asserted to deasserted while PCIE_XREFCLK is valid, at which time the warm reset is deasserted.

Motivation: The detection of an invalid PCIE_XREFCLK while the PEP is out of reset indicates a surprise lost connection, or corruption of the host-supplied reference clock.The design quarantines the host interface by keeping the PEPs in reset until the PCIE_XREFCLK is valid and a new reset request is detected.

Mode 3

Mode 3 is automatic control of reference clock selection based on PEP reset request alone. The validity of the PCIE_XRECLK (xclkval) is ignored.

In this mode, the circuit selects PCIE_XREFCLK if one of the PEPs of the target host interface is requested to be taken out of reset (**warm_reset_request** = 0). Similar to mode 2, the actual warm reset de-assertion to the target PEP is delayed by a delay programmable in PCIE_WARM_RESET_DELAY. The delay must be set to a value higher than the time required for XPLL to lock (the actual lock signal of the XPLL is not used, only delay is applied). The delay is intended to provide time for the XPLL to lock to the new clock, and there is only one timer per interface. If the two PEPs of a given interface are to be taken out of reset at slightly different times, the timer is started by the first one making a out-of-reset request, and that PEP is taken out of reset at the end of the programmed delay. The second (later) PEP out-of-reset waits for the delay to complete, if it has not done so already, it does not restart the timer.

If a PEP of a host interface is requested to be placed in warm reset, then immediately place the PEP in warm reset without delay. If both PEPs of an interface are requested to be in reset, also revert to PCIE_REFCLK.

For all modes

The mode of operation must not be changed while a PEP using this clock is out of reset (warm reset de-asserted). Changing mode might cause glitches on the clock to a PEP, and might create deadlocks in the device if that PEP is out of reset and a switch manager is currently accessing it. To change the mode, always assert reset by software (SOFT_RESET.*PCIeReset*) on affected PEPs at least 100 μs before changing the mode, and hold reset for at least 100 μs after the mode is changed.

Note: The XCLK controller posts interrupts on changes to **xclkval[3:0]** or **xlock[3:0]**. The interrupt capabilities are always available regardless of the mode configured.

4.3.5 Clock Source Selection in Hot Plug-in System

When a hot plug-in card is inserted, the system must de-assert PCIE_RESET_N[2*i+1] or PCIE_RESET_N[2*i] only when PCIE_XREFCLK[i] is known good (if PCIE_XREFCLK is used).

Also, on disconnect, it is required for the host to assert PCIE_RESET_N before cutting the reference clock to the FM10000. As an example, a short pin can be used for PCIE_RESET_N with a pull-up on this signal on the FM10000 board, thus asserting PCIE_RESET_N automatically when a card is extracted.

The normal expected timing diagram is shown in Figure 4-7. It is expected that the XLOCK remains asserted through the transition.

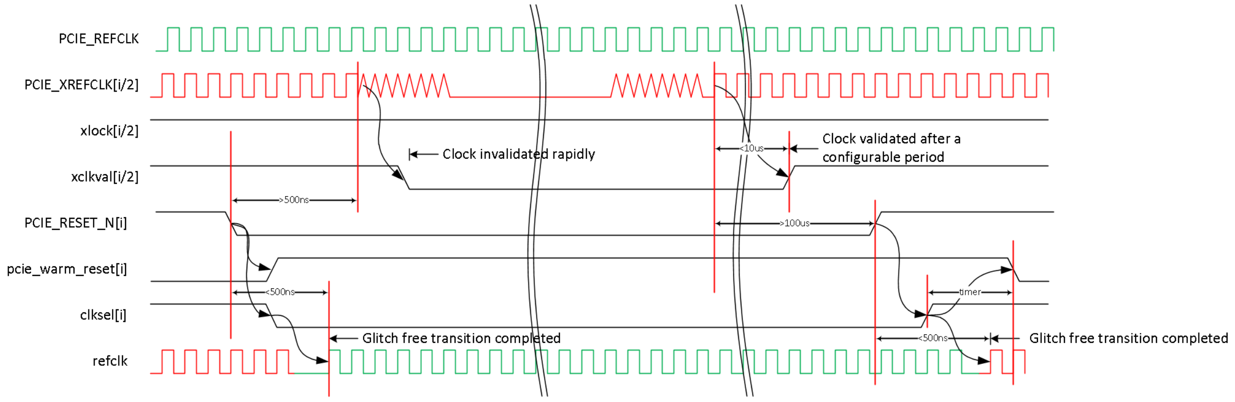

PCIE_REFCLK
PCIE_XREFCLK[i/2]
xlock[i/2]
xclkval[i/2]
PCIE_RESET_N[i]
pcie_warm_reset[i]
clksel[i]
refclk
Clock invalidated rapidly
<10us
Clock validated after a configurable period
>500ns
>100us
<500ns
timer
Glitch free transition completed
<500ns
Glitch free transition completed

Figure 4-7 PCIe Reset Timing with External Clock Transitions

The timing diagram includes the following elements:

- The PCIE_XREFCLK is assumed valid for at least 500 ns after PCIE_RESET_N is asserted. The circuit must switch to the PCIE_REFCLK within that time, and must assert the internal **pcie_warm_reset** within that time as well.
- The PCIE_XREFCLK is assumed to be valid at least 100 µs before PCIE_RESET_N is de-asserted. The clock monitoring circuit should normally be capable of detecting that the clock is valid within few microseconds; an upper bound of 10 µs is specified. Once PCIE_RESET_N is de-asserted, the circuit should change the CLKSEL to use the new XREFCLK, and then start a timer (programmable) that de-asserts **pcie_warm_reset** after it expires. This ensures enough time for the PLL to lock to the new clock. The LTSSM state (not shown) is entered within 10 ms after **pcie_warm_reset** is de-asserted.

The next timing diagram (shown in Figure 4-8) is when the **xclk_controller** is configured in mode 2 and the reset is caused by a the reference clock becoming invalid.

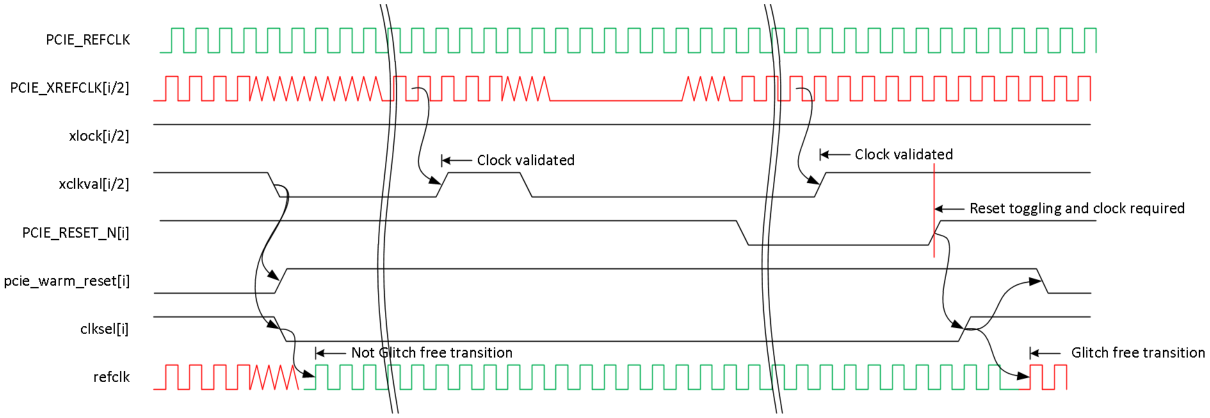


Figure 4-8 PCIe Reset Timing with Invalid External Clock Transitions

In this case, the change of the reference clock happens as soon as the clock is invalidated, and also causes the PCIe warm reset to be asserted on the interfaces affected. However, if the reference clock becomes valid again, it is not considered until the PCIE_RESET_N[i] is asserted (or software control of PCIe warm reset is used) and de-asserted again.

The mux configuration is set via the PCIE_XCLK_CTRL register. PCIE_REFCLK is assumed good. Changes to the mux configuration must be done by the Control Plane Processor or the Boot State Machine (BSM). Mux transitions are glitch-less.

4.3.6 Control and Diagnostics

The registers are:

- PCIE_PLLX_CTRL — Controls common settings for PLLS (**xpll[0..3]**).

  Note: The default management PCIE PLL and the XPLLs configuration can differ in the divider setting and lock time. For the default management PCIE PLL, the PCIE_REFCLK clock is known to be stable and the lock time should be short to allow as much time as possible to chip bring up.

- PCIE_CLK_CTRL — Controls individual setting of each XPLL and XCLK controller.
- PCIE_CLK_CTRL2 — Controls common setting for termination and observation.
- PCIE_CLKMON_CFG — Defines the clock monitors configurations.
- PCIE_CLK_STAT — Reports current XPLL status pins and XCLK controller controls.
- PCIE_CLK_IP/IM — Interrupt registers.

The Clock Observation point C controlled by LSM_CLKOBS_CTRL has the following clock observations. as shown in Table 4-7.

Table 4-7 Clock Observations

LSM_CLKOBS_CTRL.*SelC*	Observations
0	Disabled
1	PCIe PLL output divided by 8
2	Ring Oscillator Tick
3	PCIE_REFCLK pad input
4	LSM core_clock
5	1588_REFCLK pad input
6	SYSTIME clock
7	TESTDFT clock
8	PCIe PLL lock signal
9	PCIe PLL output divided by 1
10	PCIe clock
11	Tunnel clock
12	Fabric mgmt clock
13	PCIe SBus Master Spico Clock

Table 4-7 Clock Observations [Continued]

LSM_CLKOBS_CTRL.*SelC*	Observations
14	See PCIE_CLK_CTRL2.*ClkObs* (Section 11.25.2.85). The clock observations are: 0 = PCIE_XREFCLK[0] 1 = PCIE_XREFCLK[1] 2 = PCIE_XREFCLK[2] 3 = PCIE_XREFCLK[3] 4 = CLKMUX_OUT[0] 5 = CLKMUX_OUT[1] 6 = CLKMUX_OUT[2] 7 = CLKMUX_OUT[3] 8 = XPLL[0] clock output 9 = XPLL[1] clock output 10 = XPLL[2] clock output 11 = XPLL[3] clock output 12 = xlock[0] 13 = xlock[1] 14 = xlock[2] 15 = xlock[3] 16 = xclkval[0] 17 = xclkval[1] 18 = xclkval[2] 19 = xclkval[3]
15	See PCIE_CLK_CTRL2.*ClkObs* (Section 11.25.2.85). The clock observations are: 0 = xclkval[0] 1 = xclkval[1] 2 = xclkval[2] 3 = xclkval[3] 4 = XPLL_CCB_OUT[0] 5 = XPLL_CCB_OUT[1] 6 = XPLL_CCB_OUT[2] 7 = XPLL_CCB_OUT[3]

NOTE: ***This page intentionally left blank.***

5.0 Ethernet Switch

5.1 Overview

The Ethernet Shared Memory Switch (or simply Ethernet Switch) consists of the Frame Processing Pipeline, Ingress and Egress Crossbars, Frame Memory, Scheduler and MODIFY units. Figure 5-1 shows an overview of this unit.

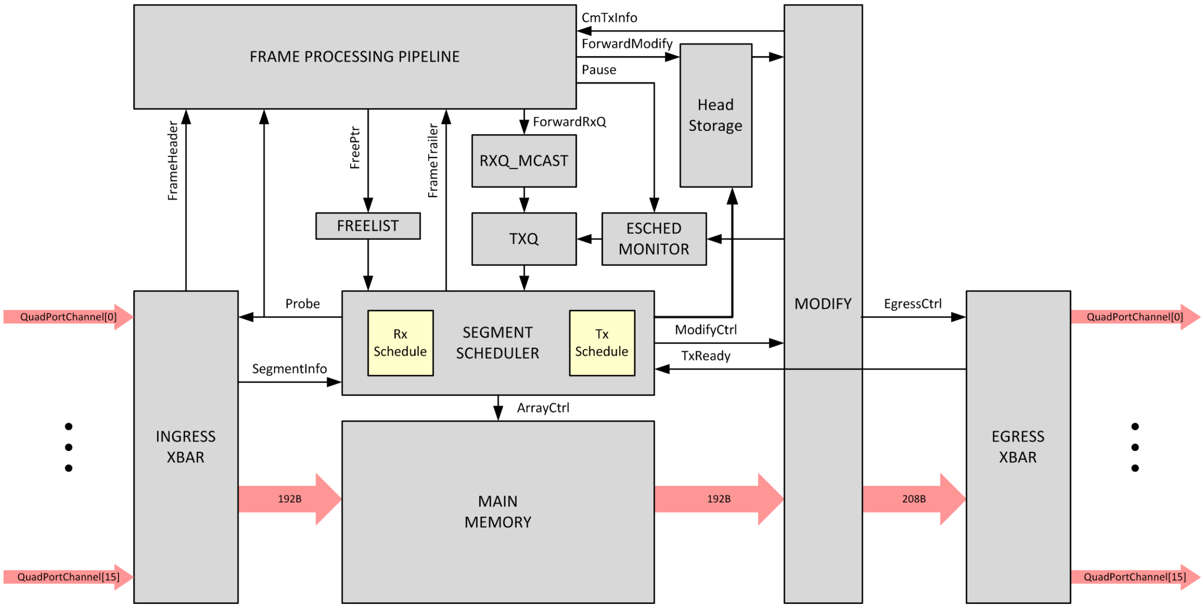


Figure 5-1 Ethernet Switch Block Diagram

The elements of the switch are described in the next sections.

5.2 Quad Port Channels

Frames arrive into the ingress crossbar or exit the egress crossbars using a Quad Port Channel. Each Quad Port Channel can be used either as a single higher speed channel or as four independent narrower and slower channels. Each individual channel can run at up to 25 GbE. Quads are used for 40 GbE or 100 GbE operation, while the narrower channels are used for independent 1/10/25 GbE ports. A PCIe host interface in 1x8 lane mode uses a full Quad Port Channel; in 2x4 lane mode, each PEP uses a single Port Channel (the even-numbered physical ports).

An individual or Quad Port Channel carries a stream of 8-byte symbols, structured as follows:

- DATA (64 bits)
- CONTROL (9 bits)
    - End of segment, end of frame (good, bad fcs, framing error).
    - Number of bytes valid if end of frame.
    - Mode of operation: QUAD or INDEPENDENT.

An individual Port Channel serializes and transmits the 8-byte symbols between end points and crossbars through a narrow 16-bit x 1.6 Gb/s asynchronous channel. If the end point supports both modes of operation, it is expected that the switch and end point are separately configured to the same mode.

The Quad Port Channels are mapped to physical end points as listed in Table 5-1.

Table 5-1 Quad Port Channel Mapping

QPC #	EndPoint	Physical Port	QPC #	EndPoint	Physical Port
0	EPL[0]	0..3	10	PCIE[2..3]	40..43
1	EPL[1]	4..7	11	PCIE[4..5]	44..47
2	EPL[2]	8..11	12	PCIE[6..7]	48..51
3	EPL[3]	12..15	13	TUNNEL[0]	52..55
4	EPL[4]	16..19	14	TUNNEL[1]	56..59
5	EPL[5]	20..23	15.0	FIBM	60
6	EPL[6]	24..27	15.1	Loopback	61
7	EPL[7]	28..31	15.2	Loopback	62
8	EPL[8]	32..35	15.3	PCIE[8]	63
9	PCIE[0..1]	36..39			

5.3 Ingress Crossbar

The ingress crossbar is a 129 Gb/s asynchronous crossbar and is responsible to un-serialize data received through Quad Port Channels into 192-byte parallel segments to memory and Frame Processing Pipeline. The ingress crossbar has memory to store up to 4x192-byte segments per channel before throttling back the asynchronous channels. The scheduler is responsible for probing the ingress crossbar for each channel periodically at a rate high enough to meet the throughput demand of the channel. This probing control is also a permit instruction to transfer the segment received to the main memory and/or the Frame Processing Pipeline. The ingress crossbar reports back to the scheduler the amount of data transferred for the current segment along with end-of-frame indicators if applicable.

5.4 Segment Scheduler

The scheduler is the unit driving the main cadence of the switch. It runs at 675 MHz, cycle time is 1.48 ns. It is responsible for managing the pool of segments, receiving segments, requiring frame header processing, managing transmit queues and transmitting segments.

The memory pool consist of 24576 segments of 192 bytes.

5.4.1 Scheduler Lists and Physical Port Mapping

For both ingress and egress, the scheduler walks through a port list repetitively probing the ingress or egress crossbar ports for segments at a rate of one list entry per cycle.

The RX and TX schedulers use separate lists, identical in principle and with the same constraints. (We use the register names corresponding to the RX list below.) The scheduler supports two RX (or TX) port lists: an active one and a standby one. Each list has up to 512-entries, the top and bottom halves of the SCHED_RX_SCHEDULE[0..1023] array, plus an implicit idle entry at the beginning of each list.

The software is expected to program one list at startup and then activate this list using the SCHED_SCHEDULE_CTRL register:

- SCHED_SCHEDULE_CTRL.*RxPage* — Defines which list is active (SCHED_RX_SCHEDULE[0..511] or SCHED_RX_SCHEDULE[512..1023]).
- SCHED_SCHEDULE_CTRL.*RxMaxIndex* — Defines the index of the last element of the list (modulo 512).
- SCHED_SCHEDULE_CTRL.*RxEnable* — Controls the RX scheduler.

If further changes are needed, software first writes the new list as the standby page (not the active one), and then updates the SCHED_SCHEDULE_CTRL.*RxPage* and *RxMaxIndex* to instruct the scheduler to flip to the new list. The scheduler only changes lists at the end of the current list.

The lists contain:

- physical port (8 bits)
- logical port (6 bits)
- idle (1 bit)
- quad (1 bit)

The "quad" is an indication that the physical port operates in a quad channel mode (required for 40 GbE, 100 GbE or for PCIe x 8 lane mode).

The 'idle' is an indication to not probe for any segments, allowing management access to get through the switch. There is always an implicit idle cycle at the beginning of each list, but this indication allows for additional explicit idle cycles.

The physical port is the Quad Port Channel number x 4 plus the channel number (0..3) in the quad. As an example, the physical port for channel 3 of EPL[5] is 23 (5x4 + 3). For quad-channel mode, always use the channel 0; in this case, channels 1..3 of that Quad Port Channel are not available for other use. Physical port numbers must be less than 64. There are no other references to physical port within the switch, aside from the RX and TX scheduler lists.

The logical port is software choice and is the "port number" used through the switch. Logical port numbers must be less than 48. There must be a consistent mapping between physical ports and logical ports, but this mapping is arbitrary.

The list is programmed to satisfy the bandwidth of the channel or quad-channel probed. The list contains up to 512-entries and thus can take up to 758 ns to get through (shorter time if the list is shorter). The maximum segment rate for a 100 GbE interface is 148.8 MHz (or 6.7 ns period) for 64-byte frames. Therefore, it means the same port must be serviced every 4.52 entry on average (6.7 ns/1.48 ns). The PCIe host interface, when running in 8-lane PCIe Gen3 mode, supports 50 GbE of nominal bandwidth for frames ≥ 256 bytes; this requires servicing once every 21.2 ns, or 14.3 entries.

Also, the list is programmed to provide a minimum spacing of 4 clock cycles between probing of the same logical port, and a minimum of 4 clock cycles between probing of same physical quad-port channel. The 4-cycle logical port spacing is enforced in hardware; if this spacing is violated, the violating list entry is replaced with an idle cycle. The switch can tolerate a temporary violation of the 4-cycle spacing rule when switching between active and standby scheduler lists.

If a port of speed less than 25 GbE is configured for cut-through operation (refer to Section 5.5.8) to other ports of the same speed, it must appear in the scheduler list with minimal jitter. This means that the minimum and maximum spacing for that port, from one entry to the next, must differ by no more than 1 cycle. This includes the spacing from the last entry for that port wrapped around to the first entry, taking into account the implicit idle cycle.

As an example, assume 4 x 100 GbE on physical port 0,4,8,12 (logical 0,1,2,3) and 1 x 10 GbE on port 16 (logical 4) and some space for management (at least one required in the whole list), at a rate of 675 MHz. Then a list could be:

Table 5-2 RX Scheduler List Example

Entry	Physical Port	Logical Port	Idle	Quad
implicit	x	x	1	0
0	0	0	0	1
1	4	1	0	1
2	8	2	0	1
3	12	3	0	1
4	0	0	0	1
5	4	1	0	1
6	8	2	0	1
7	12	3	0	1
8	16	4	0	0
9	0	0	0	1
10	4	1	0	1
11	8	2	0	1
12	12	3	0	1
13	0	0	0	1
14	4	1	0	1
15	8	2	0	1
16	12	3	0	1
17	0	0	0	1
18	4	1	0	1
19	8	2	0	1

Table 5-2 RX Scheduler List Example [Continued]

Entry	Physical Port	Logical Port	Idle	Quad
20	12	3	0	1
21	x	x	1	0

This list has *RxMaxIndex*=20, and a total of 22 entries including the implicit idle. It takes 32.56 ns to go through, and the probing rate for 100 GbE ports is 6.52 ns on average which is smaller than the minimum 6.7 ns required.

Furthermore, there should be a small amount of overspeed provided in the schedule to account for ppm differences in the Ethernet clocks of different devices. The Ethernet clock specification requires support for a minimum frame processing overspeed of 0.02%.

5.4.2 Managing Segment Free List

The scheduler retrieves a segment pointer from the free list if a probe returns that a segment is available to be received on a given port. This pointer is forward to the array to indicate where the incoming segment is to be stored. If the free list is empty and there are no pointers available, any incoming frames are terminated and marked with an error.

5.4.3 Scheduler Initialization

The scheduler initialization follows the following steps:

- Clear memories
- Initialize the RXQ_MCAST list.
- Initialize the TXQ list.
- Initialize the FREE list.
- Initialize the schedule.

The steps are detailed in the next sections.

5.4.3.1 Clear Switch Memories

All scheduler and other switch fabric memories must be cleared each time the switch is initialized, before SWITCH_RESET is de-asserted. This is done by activating the BIST engine in the switch memories with a pattern fill command. (The CM_PAUSE_RCV_STATE memory requires special handling, as shown below.)

All BIST-accessible memories in the FM10000 are cleared as part of the recommended chip initialization (refer to Section 4.1.3.11), so the following step need not be repeated during the first switch initialization after bring-up.

```
// Clear BIST-accessible switch memories (all but CM_PAUSE_RCV_STATE)
BIST_CTRL.BistMode_FABRIC = 1;
BIST_CTRL.BistRun_FABRIC = 1; // this step must be sequenced after BistMode_FABRIC = 1
wait ( 0.8 ms );
BIST_CTRL.BistRun_FABRIC = 0;
BIST_CTRL.BistMode_FABRIC = 0;
```

The CM_PAUSE_RCV_STATE is not BIST-accessible and must be explicitly written to 0. This step is performed after SWITCH_RESET is de-asserted, but before asserting SWITCH_READY.

```
// Clear CM_PAUSE_RCV_STATE
for (i=0;i<48;i++)
{
CM_PAUSE_RCV_STATE[i] = 0;
}
```

5.4.3.2 Initialization of RXQ_MCAST List

The first 8 freelist pointers are written directly to the SCHED_RXQ_STORAGE_POINTERS registers, the rest to SCHED_RXQ_FREELIST_INIT. The maximum total number of freelist pointers allowed is 1 K.

```
// RXQ_MCAST list initialization
 for (i=0;i<8;i++)
 {
     SCHED_RXQ_STORAGE_POINTERS[i].HeadPage = i;
     SCHED_RXQ_STORAGE_POINTERS[i].TailPage = i;
     SCHED_RXQ_STORAGE_POINTERS[i].HeadIdx = 0;
     SCHED_RXQ_STORAGE_POINTERS[i].TailIdx = 0;
     SCHED_RXQ_STORAGE_POINTERS[i].NextPage = 0;
}
 for (i=0;i<(1024-8);i++)
 {
     SCHED_RXQ_FREELIST_INIT.Address = 8+i;
 }
```

5.4.3.3 Initialization of TXQ List

The first 384 pointers are written to the head/tail memories, and the rest to the SCHED_TXQ_FREELIST_INIT. Note that both of the tail SRAMs and the head SRAM should be configured identically. The maximum total number of freelist pointers allowed is 24 K.

```
// TXQ list initialization
 for (i=0;i<384;i++)
 {
     SCHED_TXQ_HEAD_PERQ[i].Head = i;
     SCHED_TXQ_TAIL0_PERQ[i].Tail = i;
     SCHED_TXQ_TAIL1_PERQ[i].Tail = i;
 }
 for (i=0;i<(24576-384);i++)
 {
     SCHED_TXQ_FREELIST_INIT.Address = 384+i;
 }
```

Note: All *XXX*_PERQ registers are indexed [0..383] which is equal to PORT*8+TC.

5.4.3.4 Initialization of FREE Segment List

The first 48 main array segments pointers are written into SCHED_SSCHED_RX_PERPORT and the rest to SCHED_FREELIST_INIT_ADDR. The maximum total number of freelist pointers allowed is 24 K.

```
// FREE segement list initialization
 for (i=0;i<48;i++)
 {
     SCHED_SSCHED_RX_PERPORT[i].Next = i;
 }
 for (i=0;i<(24576-48);i++)
 {
     SCHED_FREELIST_INIT.Address = 48+i;
 }
```

5.4.3.5 Initialization of Scheduler Polling Schedule

The TX and RX schedule must be initialized according to the relative speed of each port. The example below is 45x10 GbE (36x10 GbE for Ethernet, 5x10 GbE for PEPs (bifurcation mode off), 2 for Tunneling Engines, 1xFIBM and 1x10 GbE for idle).

Note: Writing to SCHED_SCHEDULE_CTRL enables the scheduler, this should be done at the end after RXQ, TXQ and FREE lists are initialized.

```
// Logical ports 0..35 are physical ports 0..35 (Ethernet)
 for (i=0;i<36;i++)
 {
     SCHED_RX_SCHEDULE[i].Port = i;
     SCHED_RX_SCHEDULE[i].PhysPort = i;
     SCHED_RX_SCHEDULE[i].Quad = 0;
     SCHED_RX_SCHEDULE[i].Idle = 0;
     // Repeat for TX
     SCHED_TX_SCHEDULE[i].Port = i;
     SCHED_TX_SCHEDULE[i].PhysPort = i;
     SCHED_TX_SCHEDULE[i].Quad = 0;
     SCHED_TX_SCHEDULE[i].Idle = 0;
 }

// Logical ports 36..41,46,47 are 5xPEPs and 2xTE and 1xFIBM (bifurcation ON)
// (last logical port is mapped to PEP[8]
 int xLogPort[8] = { 36, 37, 38, 39, 40, 41, 46, 47 };
 int xPhysPort[8] = { 36, 40, 44, 48, 52, 56, 60, 63 };
 int xQuad[8] = { 1, 1, 1, 1, 1, 1, 0, 0 };
 for (i=0;i<8;i++)
 {
     SCHED_RX_SCHEDULE[i+36].Port = xLogPort[i];
     SCHED_RX_SCHEDULE[i+36].PhysPort = xPhysPort[i];
     SCHED_RX_SCHEDULE[i+36].Quad = xQuad[i];
     SCHED_RX_SCHEDULE[i+36].Idle = 0;
     // Repeat for TX
     SCHED_TX_SCHEDULE[i+36].Port = xLogPort[i];
     SCHED_TX_SCHEDULE[i+36].PhysPort = xPhysPort[i];
     SCHED_TX_SCHEDULE[i+36].Quad = xQuad[i];
     SCHED_TX_SCHEDULE[i+36].Idle = 0;
}

 // Start Scheduler
 SCHED_SCHEDULE_CTRL.RxPage = 0;
 SCHED_SCHEDULE_CTRL.RxMaxIndex = 43;
 SCHED_SCHEDULE_CTRL.RxEnable = 1;
 SCHED_SCHEDULE_CTRL.TxPage = 0;
 SCHED_SCHEDULE_CTRL.TxMaxIndex = 43;
 SCHED_SCHEDULE_CTRL.TxEnable = 1;
```

5.5 Receive Queues and Transmit Queues

5.5.1 Queuing

The Frame Processing Pipeline forwards the destination decision to the RXQ_MCAST unit. This unit first enqueues the frame into a per-traffic-class receive queue (RXQ), and then uses the destination mask and mirroring commands to replicate the frame pointer to the different transmit queues. The frame gets dequeued from the received queues once it can be fully enqueued into transmit queues.

The destination mask is a 48-bit port-mask indicating which port is supposed to egress this frame. Also, a dual-mirror command is attached to the frame to indicate if one or two copies of the frame are scheduled for mirroring purpose.

The receive queues are fully provisioned and thus capable to keep track of up to 24,576 frames.

The TXQ unit manages the insertion and extraction into the transmit queues. Transmit queues are organized per port and per traffic class and could store up to 24,576 entries.

TXQ is fully provisioned as long as one-copy of the frame is requested (by example, normal, non-mirrored unicast frames). If some or many frames are multicasted, two events occur; (a) there a risk that the memory in TXQ becomes full before frame memory is fully exhausted causing the RXQ_MCAST unit to throttled back, (b) lower priority traffic with large replications delays could cause higher priority unicast traffic to be delayed. To reduce these side effect, the register MCAST_LIMITED_SKEW_MULTICAST defines how replication is done for each traffic class; either (a) replicate all pointers before processing another traffic class, (b) do a single replication before processing another traffic class. Case (b) is appropriate for lower priority classes.

In any case, if TXQ is running out of resources completely, RXQ_MCAST holds moving from receiver queues to transmit queues until resources are freed up in TXQ. The RXQ_MCAST continues to enqueue frames received from the frame processing pipeline into its receive queues while waiting for resources to free up in the TXQ. Therefore, frames are not lost during a temporary blockage of the TXQ.

Figure 5-2 illustrates the process.

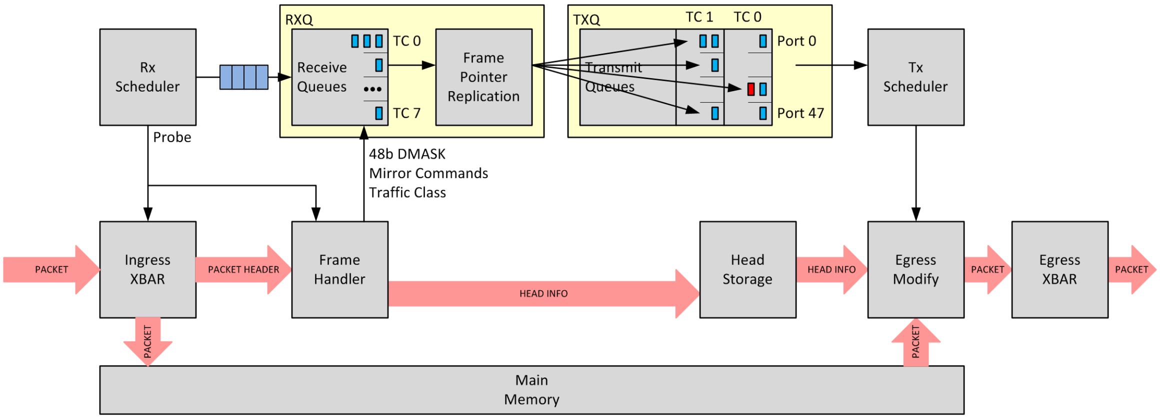


Figure 5-2 Frame Replication

The frame's payload is stored only once in the main memory, and only the pointer to the frame is actually replicated to the different queues.

5.5.2 Frame Replication on Same Port

At transmit time, frames are potentially replicated multiple times on a given port using different VLANs and/or DGLORTs for each copy. This is typically used for replication during IP multicast routing and/or replication across multiple PCIe VMs. Replicated frames are queued only once on a given port even if replication is required multiple times. From a scheduling perspective, each replicated frame is treated like an individual frame, i.e. shaping and prioritization applies to each frame that is replicated and the replicated frames might be interleaved with frames from other higher priority queues. However, for a given transmit class, all replication for a given frame has to be complete before processing the next frame for the same port and same traffic class.

Replication on the same port occurs when a non-zero IP multicast index (from GLORT_DEST_TABLE.*IP_MulticastIndex*) is selected by the Frame Processing Pipeline.

The tables involved in the multicast process are:

- SCHED_MCAST_DEST_TABLE
- SCHED_MCAST_LEN_TABLE
- MOD_MCAST_VLAN_TABLE

There are two masks involved in the process:

- DMASK — The actual destination port mask supplied by the Frame Processing Pipeline (not including logged or mirrored copies).
- MCAST_MASK — The port mask found in SCHED_MCAST_DEST_TABLE[IP_MulticastIndex].*PortMask*.

The DMASK is used to determine on which ports a frame is transmitted. The MCAST_MASK is used to determine which ports use VLAN lists. Specifically, MCAST_MASK determines the indexing of an intermediate table, SCHED_MCAST_LEN_TABLE, that defines the location and length of each VLAN list in the MOD_MCAST_VLAN_TABLE.

If the DMASK is 1 for a given port and MCAST_MASK is 1 for a given port, this frame is enqueued on this port and subject to replication. If the DMASK is 1 for a given port and the MCAST_MASK is 0 for a given port, it means that there is no VLAN replication required for this port and the native copy gets sent.

If the DMASK is 0 for a given port no frame is sent on that port regardless of the value of the MCAST_MASK. However, the number of VLAN lists is derived from the MCAST_MASK only, and independent of the actual DMASK. This allows the same list to be used regardless of whether the actual frame is forbidden to be transmitted on a given port, for whatever reason (by example, congestion or egress drop).

The entries in SCHED_MCAST_LEN_TABLE are packed according to the MCAST_MASK. Thus when DMASK is 1 and MCAST_MASK is 1 for a given port P, the index into SCHED_MCAST_LEN_TABLE is determined as follows:

> If there are N bits set in MCAST_MASK{P-1:0} (preceding bit P), the index used is SCHED_MCAST_DEST_TABLE.*LenTableIdx* + N (with addition modulo the table size).

The frame flow is shown in Figure 5-3.

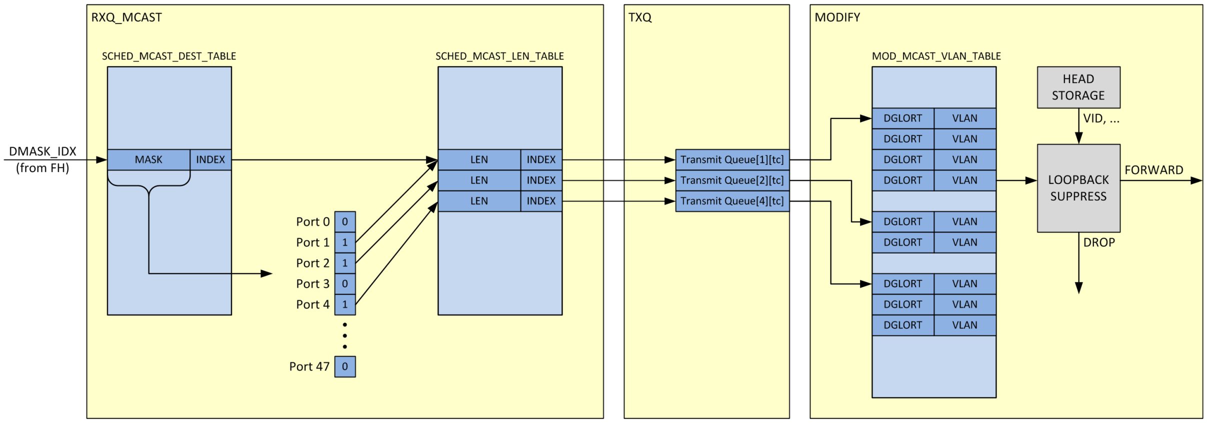


Figure 5-3 VLAN Frame Replication Indexing

The entries in the MOD_MCAST_VLAN_TABLE consist of:

- *VID* (12 bits) — New VLAN ID to attach to the frame.
- *DGLORT* (16 bits) — New DGLORT to attach to the frame.
- *ReplaceVID* (1 bit) — Indicates if VLAN ID updated.
- *ReplaceDGLORT* (1 bit) — Indicates if DGLORT is updated.

Mirror copies are sent in addition to the multicast copies described above. Mirror copies are not affected by SCHED_MCAST_DEST_TABLE, SCHED_MCAST_LEN_TABLE, or MOD_MCAST_DEST_TABLE configuration.

5.5.3 MODIFY Loopback Suppress

The MODIFY unit also includes a generic multicast loopback suppress. This avoids transmitting a frame back to the port it came from for the same VLAN. This function is described in Section 5.8.4.4.

5.5.4 Managing MCAST Tables

The SCHED_MCAST_LEN_TABLE is before scheduler while MOD_MCAST_VLAN_TABLE is after scheduler. Because frames could be held in the scheduler for a long time, it is not easy to determine when it is safe to modify MOD_MCAST_VLAN_TABLE, as frames in flight could still be relying on old entries.

To assist in management of this table, the hardware supports an “epoch” index (1 bit, set to 0b or 1b). The hardware tracks incoming and freed frames according to the epoch; MCAST_EPOCH_USAGE[0..1] provides a count of the number of frame memory segments in use for each epoch. For each incoming frame, the counter for the current epoch is incremented for each memory segment consumed by the frame, and the epoch is also attached to the frame. When the memory segments are freed (after all instances of the frame have been transmitted), the epoch's counter is decremented by a corresponding amount.

If the software needs to recover space for a new VLAN list, it removes the reference to the old VLAN list so that no new arriving frames will use it, then flips the epoch in the MCAST_EPOCH and monitor MCAST_EPOCH_USAGE[previousEpoch] until the counter reaches zero. At that point, it means there are no more frames using the old epoch, and the old VLAN list is no longer in use and can be safely recovered.

5.5.5 Multicasting Registers

Table 5-3 lists the full list of registers used during multicasting.

Table 5-3 Frame Replication Register Usage

Register	Field	Width (Bits)	Function
SCHED_MCAST_DEST_TABLE	*PortMask*	48	Defines if there is a VLAN replication list for each port. The index for this table is supplied as part of the forwarding information received from FPP.
	LenTableIdx	14	Defines the base index in the SCHED_MCAST_LEN_TABLE.
SCHED_MCAST_LEN_TABLE	*L3_McastIdx*	15	Pointer to the MOD_MCAST_VLAN_TABLE for the list of VLAN entries.
	L3_Repcnt	12	1 less than the number of entries in the VLAN list. Legal range: 0-4093
MOD_MCAST_VLAN_TABLE	*VID*	12	New VID to use if update requested.
	DGLORT	16	New DGLORT to use if update requested.
	ReplaceVID	1	Indicates if VID must be updated or not.
	ReplaceDGLORT	1	Indicates if DGLORT must be updated or not.
MCAST_EPOCH_USAGE[0..1]	Count	16	Frame memory segments in use per epoch.
MCAST_EPOCH	Current	1	Current epoch.

5.5.6 Replication to VFs

If a PEP has SR-IOV enabled, flood/multicast/broadcast frames need to be replicated in hardware to each virtual function (VF) on that PEP. In this case, the switch's broadcast as well as potentially the switch's flood GLORT needs to replicate a copy of the frame to each VF, leaving the VLAN the same but changing the DGLORT to point to the specific VF. The VF configures itself to subscribe/unsubscribe to floods and broadcasts. This must be reflected in this replication group.

To simplify the programming of these flood and broadcast replication groups, VLAN membership for these replicated copies is done in the PEP. Thus, if any of the VFs or the PF belong to a VLAN, the egress VLAN membership in the switch should include that PEP port. Each copy is compared against the per-VF VLAN membership table in the PEP. This causes extra copies to be made between the switch and PEP, but resolves the issue of having to replicate the flood and/or broadcast replication lists per VLAN.

In addition, the PEP checks that the destination VF's source GLORT does not equal the source GLORT found in the frame's FTAG (set when the frame was sent). This is described in the loopback suppression section of the PEP RX.

5.5.7 Loopback Suppression from PCIe Ports

A PCIe port may represent within it multiple logical ports (for example when SR-IOV is enabled, or when a VMDQ type logical port scheme is employed assigning ports to sets of queues). For this reason each queue in each PCIe port is given a source GLORT that is used for both loopback suppression as well as logical port identification in the frame processor.

The frame processor does not do loopback suppression across sub-ports. This is left to the PCIe port RX itself. When the ingress and egress VIDs are equal (denoting switching as opposed to routing) FTAG.USER[0] is set by the MCAST unit. This bit is used in the PCIe port RX to enable loopback suppression on a per sub-port basis.

5.5.8 Cut-through and Store-and-Forward Modes

The FM10000 supports cut-through and store-and-forward modes.

The switch processes frame headers and makes switching decisions as soon as the header is received. It does not wait for the end of the frame to be received. This allows the switch to start transmission of a given frame to an egress port while still receiving the frame on ingress. This mode is defined as cut-through.

Cut-through mode of operation presents the advantage of a lower latency through the switch and thus offers high-performance in latency-sensitive applications. However, for cut-through mode to work properly, the speed of the egress port must be lower or equal to the speed of ingress port. As an example, cut-through does not work properly if a frame is switched from a 10 GbE port to a 100 GbE port; the egress port under-runs.

Secondly, the cut-through mode of operation implies transmission of a frame is started before the end of frame status is received, and its end-of-frame status is known (good CRC or bad CRC). Therefore, if the frame egress has been started already by the switch and the frame has a bad CRC or a framing error, the switch terminates the frame with a bad CRC, thus carrying the error through. This behavior might not be desirable in some networks.

The FM10000 supports a mechanism to control when to apply cut-through. This is controlled via the SAF_MATRIX register, which defines an ingress/egress matrix of when to operate in cut-through or store-and-forward mode. This register also defines a per-ingress-port control of errored frame disposition and the cut-through latency.

For a given frame received, the SAF_MATRIX defines if cut-through is enabled for each egress ports. If, for example, a frame is received on port 1 (assuming at 10 GbE) and is multicasted to ports 2, 3 and 4 (speed 1 GbE, 10 GbE and 100 GbE, respectively), the SAF_MATRIX[ingressPort].*EnableSNF*[egressPort] defines if the frame is store-and-forward (1b) or cut-through (0b). In this example, if cut-through is desired, software sets the SAF_MATRIX[1].*EnableSNF*[2] to 0b (1 GbE slower than 10 GbE), SAF_MATRIX[1].*EnableSNF*[3] to 0b (10 GbE equal to 10 GbE) and SAF_MATRIX[1].*EnableSNF*[4] to 1b (100 GbE faster than 10 GbE, thus store-and-forward required). A given ingress frame is store-and-forward if EnableSNF is set for any of its egress ports.

The SAF_MATRIX contains the following fields:

- SAF_MATRIX[ingressPort].*EnableSNF* (48 bits) — Defines if a frame is store-and-forward (1b) or cut-through (0b) for each port.
- SAF_MATRIX[ingressPort].*CutThruMode* (2 bits) — Defines the cut-through behavior, choices are:

  00b = Full cut-through, forward right away.
  01b = Wait for one 192-byte segment or end of frame, then forward.
  10b = Wait for two 192-byte segment or end of frame, then forward.
  11b = Wait for end of frame, then forward.

- SAF_MATRIX[ingressPort].*IgnoreError* — Defines if bad frames are filtered or not in store-and-forward mode.

Note: "Forward" in this context means placing the frame in a transmit queue. The frame scheduler is the entity scheduling the frame transmission.

Table 5-4 defines the processing for each frame. A filtered frame is not forwarded (frame not queued), while a non-filtered frame is forwarded (frame queued).

Table 5-4 Store-and-Forward and Cut-Through Table Setting

EnableSNF	CutThruMode	IgnoreErr	Mode	Bad Frames
1	x	0	Wait full frame, then forward	Filtered
1	x	1	Wait full frame, then forward	Forwarded[1]
0	0	x	Forward immediately	Forwarded[1]
0	1	0	Wait 1 segment, then forward	Filtered if ≤ 192 bytes, forwarded[1] otherwise
0	1	1	Wait 1 segment, then forward	Forwarded[1]
0	2	0	Wait 2 segments, then forward	Filtered if ≤ 384 bytes, forwarded[1] otherwise
0	2	1	Wait 2 segments, then forward	Forwarded[1]
0	3	0	Wait full frame, then forward	Filtered
0	3	1	Wait full frame, then forward	Forwarded[1]

[1] If a frame is received with a framing error or a bad CRC and is forwarded, it always egresses the switch with a bad CRC, except if the frame is a mirrored truncated frame. Filtered frames are not mirrored.

5.6 Frame Scheduling

5.6.1 Overview

The FM10000 switch handles egress frame scheduling in a specific hardware subsystem called the ***scheduler***. The FM10000 scheduler provides the following frame-scheduling features:

- 8 queues per port for a total of 384 queues, one queue per traffic class per port.
- A traffic-shaping facility, which guarantees that specified subsets of queues are limited to a specified maximum bandwidth.
- Optional strict prioritization of traffic (high or low).
- Optional Deficit Round-Robin (DRR) scheduling for configurable, fair bandwidth allocation between groups of queues.
- Port- and class-based PAUSE.
- A timeout mechanism, which discards frames older than a specified age.

The scheduler also handles the scheduling of frame segments.

The external agent that configures the scheduler is referred to as the user. The user is normally a piece of system software.

The FM10000 scheduler performs scheduling in discrete scheduling rounds. It schedules up to one frame per scheduling round. On each scheduling round, the scheduler processes all queues associated with a single egress-port. Egress ports are processed in the sequence defined by the SCHED_CONFIG registers. The scheduler does not skip ports without waiting traffic.

5.6.2 Register Set

Maps:

- CM_APPLY_SWITCH_PRI_TO_TC — Refer to Section 11.19.2.1.
- CM_TC_PC_MAP[0..47] — Refer to Section 11.20.2.15.

Timeout configuration:

- FRAME_TIME_OUT — Refer to Section 11.14.2.2.

Traffic-shaping configuration:

- CM_BSG_MAP[0..47] — Refer to Section 11.20.2.21.
- TX_RATE_LIM_CFG[0..47][0..7] — Refer to Section 11.20.2.19.
- CM_GLOBAL_CFG.*ifgPenalty* — Refer to Section 11.20.2.14.

Priority and DRR scheduling configuration:

- SCHED_ESCHED_CFG_1[0..47] — Refer to Section 11.23.2.7.
- SCHED_ESCHED_CFG_2[0..47] — Refer to Section 11.23.2.8.
- SCHED_ESCHED_CFG_3[0..47] — Refer to Section 11.23.2.9.
- SCHED_MONITOR_DRR_CFG_PERPORT[0..47] — Refer to Section 11.23.2.15.

- SCHED_MONITOR_DRR_Q_PERQ[0..383] — Refer to Section 11.23.2.14.
    - — All XXX_PERQ registers are indexed [0..383] which is equal to PORT*8+TC.

5.6.3 Detailed Scheduling Algorithm

The FM10000 TX scheduler cyclically steps through the ports in the order given by the configuration of TX scheduler (refer to Section 5.4.1, “Scheduler Lists and Physical Port Mapping”). FM10000 traffic is made up of frames—each frame consists of a non-zero number of segments, from one segment up to a configurable number. The FM10000 scheduler is responsible for picking which frame and which segment to send next.

From the point of view of TX frame scheduling, a port that has started sending a frame is called *busy* until it is done with the frame, at which point it becomes *idle* and remains idle until another frame is scheduled for that port. Once the scheduler has chosen a frame, it sends the segments of that frame in order. Therefore, frame scheduling can only take place on idle cycles. The cycle when the final segment of a frame is scheduled is an idle cycle—the scheduler does not introduce gaps between frames as long as there is traffic that is allowed to egress.

When the scheduler reaches an idle port, it makes a frame-scheduling decision: either to schedule a frame and if so which frame, or not to schedule any frame if there is no traffic available that is allowed by the scheduler state to egress. This section describes the process by which the scheduler makes its frame-scheduling decision.

5.6.3.1 Timeout Mechanism

At the highest priority, the scheduler handles frame timeouts. Frames have timed-out if they have remained in the switch longer than the time configured using FRAME_TIME_OUT.*timeoutMult*. Such frames are immediately discarded, but the operation takes a scheduler cycle per segment; no frame can egress on a cycle when this happens.

The design does not guarantee that timeout bandwidth is fully provisioned. For more details, refer to Section 5.6.4.4, “Timeout Bandwidth Provisioning”.

The design does not guarantee a high degree of precision in the timeout period. The actual timeout period experienced by a frame could be +/- 50% from the configured value.

5.6.3.1.1 Register Field Cross Reference

- FRAME_TIME_OUT.*timeoutMult* — Refer to Section 11.14.2.2.

5.6.3.2 Eligible Traffic

Traffic that may egress is called *eligible* traffic.

Traffic that is present is eligible if it has not timed out, is in a queue that is enabled, not paused, and has not exceeded its maximum bandwidth allocation under traffic-shaping rules.

The user enables/disables queues by setting a control register bit (one for each queue) in SCHED_ESCHED_CFG_2.

The FM10000 monitors ports for pause frames from each port's link partner; it supports two kinds of pause frames: standard IEEE 802.3 (Annex 31B) pause and IEEE 802.1Q class-based pause frames.

The FM10000 supports the user's creating traffic-shaping groups with configurable bandwidth limits. Details of the traffic-shaping counters are in Section 5.6.4.2, “Traffic-Shaping Counter Calculations”.

5.6.3.2.1 Register Field Cross Reference

- CM_TC_PC_MAP.*PauseClass* — Refer to Section 11.20.2.15.
- CM_BSG_MAP.*BSG* — Refer to Section 11.20.2.21.
- TX_RATE_LIM_CFG.*Capacity* — Refer to Section 11.20.2.19.
- TX_RATE_LIM_CFG.*RateUnit* — Refer to Section 11.20.2.19.
- TX_RATE_LIM_CFG.*RateFrac* — Refer to Section 11.20.2.19.
- SCHED_ESCHED_CFG_2.*TcEnable* — Refer to Section 11.23.2.8.

5.6.3.3 Prioritization

5.6.3.3.1 Strict Priority Ranges

The FM10000 scheduler prioritizes eligible traffic by traffic class. In the simplest mode, the scheduler prefers traffic in queues with higher traffic-class indices over traffic in queues with lower traffic-class indices. This is called *strict priority*. Strict priority is exclusionary (i.e., no lower-indexed traffic egresses as long as higher-indexed traffic is available.

5.6.3.3.2 Deficit Round-Robin Range

The scheduler further allows the user's setting aside any contiguous range of traffic classes for Deficit Round-Robin (DRR) bandwidth sharing. Doing so overrides the strict index-based prioritization within that contiguous range. Traffic in higher-indexed classes than the DRR range are still strictly prioritized over anything in the DRR range, and anything in the DRR range is still strictly prioritized over lower-indexed classes. Within the DRR range, the DRR algorithm guarantees fair bandwidth sharing, as described below.

5.6.3.3.3 Scheduling Groups within DRR Range

The user is responsible for configuring the DRR range into non-overlapping scheduling groups consisting of any non-zero number of contiguous traffic classes. The user further configures the scheduling groups with specific (relative) bandwidth allocations, and the scheduler then ensures that the groups share the bandwidth fairly up to their allocations by delaying traffic that has exceeded its allotment.

The scheduling allocation is configured into the SCHED_MONITOR_DRR_Q_PERQ.*q* register corresponding to the highest numbered traffic class of each scheduling group. For each group, this highest indexed traffic class is referred to as the *group leader*. For further details of the meaning of the DRR quantum, refer to Section 5.6.4.3, “Notes on Deficit Round-Robin Scheduling”.

At times that multiple scheduling groups have DRR allocations available, the scheduler alternates between them in round-robin fashion.

5.6.3.3.4 Nested Strict Priority and InnerPriority within Scheduling Groups

Scheduling groups that contain more than one traffic class retain strict priority between internal traffic classes. Normally, higher-indexed traffic classes take priority over lower-indexed traffic classes. However, an extra user configuration bit, the inner priority, overrides the normal order and specifies that classes with inner priority is handled by the scheduler with precedence over all classes without inner priority (the sorting order is otherwise the same).

The user must set *TcInnerPriority* to zero for non-DRR (strict-high and strict-low) traffic classes. In any case, prioritization between traffic in different traffic classes within the same scheduling group is strict in the sense that the scheduler does not schedule any lower-priority traffic while there remains higher-priority traffic within the same scheduling group.

5.6.3.3.5 An InnerPriority Use Case

An application for inner priority is to enable the user to dynamically change traffic-class indices for a traffic flow without interfering with FIFO ordering of data frames through the switch.

For example, assume that the user has configured traffic classes 3 and 2 to belong to the same scheduling group and then decides to remap a traffic flow from class 2 to class 3. Before remapping the traffic, the user sets the *TcInnerPriority* bit for class 2. The user then changes the mapping so that the flow is directed to traffic class 3. Since traffic class 3 now has lower priority than traffic class 2, the scheduler allows any frames in class 2 to egress before allowing the frames that have been queued in class 3 to egress. Once the frames in class 2 have egressed, the user can unset *TcInnerPriority* for traffic class 2 and reassign that class for other purposes (e.g., shift it into another scheduling group or use it for strict-low priority traffic).

The overall effect is that the scheduler schedules the frames for egress in the same order that they were received even though the newer frames were sent through a traffic class with a higher index than the older frames.

5.6.3.3.6 Register Field Cross Reference

- SCHED_ESCHED_CFG_2.*StrictPriority* — Refer to Section 11.23.2.8.
- SCHED_MONITOR_DRR_CFG_PERPORT.*zeroLength* — Refer to Section 11.23.2.15.
- SCHED_ESCHED_CFG_1.*TcGroupBoundary* — Refer to Section 11.23.2.7.
- SCHED_MONITOR_DRR_Q_PERQ.*q* — Refer to Section 11.23.2.14.
- SCHED_ESCHED_CFG_3.*TcInnerPriority* — Refer to Section 11.23.2.9.

5.6.3.4 Example DRR Configuration

Described here is an example DRR configuration, assuming that the following configuration is desired for a given egress port of the switch. (The configuration is intended for illustrative purposes only and no suitability for any particular purpose is implied.)

- All eight traffic classes 7..0) are in use.
- Two strict-high priority traffic classes, one higher than the other, and both higher than DRR.
- One strict-low priority traffic class, with lower priority than DRR traffic.

- Three configured DRR groups, with the names alpha, beta, and gamma and with the given DRR quantum allocations Qalpha, Qbeta, and gamma.

  Note: The names used here for the groups are arbitrary. In the hardware configuration the groups are anonymous, and system programmers are free to invent their own names, should they need to name the groups within system software.

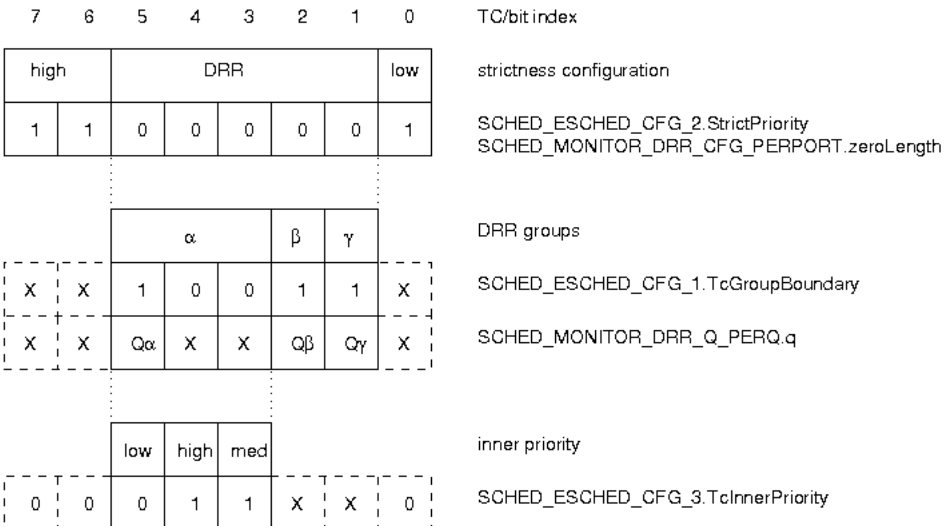
- The DRR group alpha contains three traffic classes with strict priority between them, and beta and gamma contain one traffic class each.
- It is desired that the normally highest-priority traffic class of the alpha DRR group have its priority temporarily lowered so that it is the lowest-priority traffic class within the alpha group.

To achieve the above desiderata in the FM10000, the user configures the scheduler as follows. Traffic classes 7 and 6 are marked as strict-high priority, which entails setting the *StrictPriority* and *zeroLength* bits for these classes. The user clears *StrictPriority* and *zeroLength* for the five DRR classes 5..1. The user is not free to choose any other classes given the desired number of strict-high and strict-low classes, since the range of DRR classes is required to be contiguous; we see that in consequence, the ranges of strict-high and strict-low classes also must be contiguous.

Within the DRR range 5..1 the user is free to choose any contiguous range of three for group alpha, and it is assumed that the user has chosen 5..3 for this group. it is further assumed that beta has been assigned traffic class 2, which leaves traffic class 1 for gamma. The user sets this configuration by setting *TcGroupBoundary* for the group leaders of the chosen groups, viz., for traffic classes 5, 2, and 1; the user also writes the Q values into the q registers of the same traffic classes. The user clears the remaining *TcGroupBoundary* bits, and is free to set the Q values of the corresponding traffic classes to any arbitrary value or not to set them at all, since they will not be used by the hardware in the chosen configuration. The configuration of *TcGroupBoundary* for strict-high or strict-low classes may also be left in any convenient state; these bits is also be used.

Within the group alpha, normally traffic class 5 would have the highest priority. The user's setting *TcInnerPriority* for traffic classes 3 and 4 elevates them over traffic class 5. By the Prioritization rules described in Section 5.6.3.3, the *TcInnerPriority* bits are cleared by the user for traffic classes 7, 6, and 0. *TcInnerPriority* for classes 2 and 1 may be left in any convenient state; these bits are not used by the hardware since the groups beta and gamma only contain a single traffic class each.

With the above assumptions, the configuration is as shown in Figure 5-4. Configuration values marked X are don't-cares.



7	6	5	4	3	2	1	0	TC/bit index
high		DRR					low	strictness configuration
1	1	0	0	0	0	0	1	SCHED_ESCHED_CFG_2.StrictPriority SCHED_MONITOR_DRR_CFG_PERPORT.zeroLength
		α			β	γ		DRR groups
X	X	1	0	0	1	1	X	SCHED_ESCHED_CFG_1.TcGroupBoundary
X	X	Qα	X	X	Qβ	Qγ	X	SCHED_MONITOR_DRR_Q_PERQ.q
		low	high	med				inner priority
0	0	0	1	1	X	X	0	SCHED_ESCHED_CFG_3.TcInnerPriority

Figure 5-4 Example Configuration of DRR Mechanism

5.6.4 Algorithm Notes

5.6.4.1 Effective Frame Length

The *effective frame length* of a frame is equal to the frame's length in bytes as it is transmitted (thus after egress modification) plus an interframe gap plus a preamble byte count penalty (together the total penalty). The effective frame length models the higher bandwidth cost of scheduling small frames versus scheduling large frames. The total penalty is user configurable as follows.

- For the purpose of Egress Rate Limiting, the effective length of a packet is corrected using CM_GLOBAL.*ifgPenalty* and not programmable per port.
- For the purpose of Deficit Round Robin schedule, the effective length of a packet is corrected using SCHED_MONITOR_DRR_CFG_PERPORT[port].*ifgPenalty* and is programmable per port.

The per-port or global values could be programmed differently if desired. The default values for the penalty are 0, indicating to not take IFG and preamble into account as part of the computation.

5.6.4.2 Traffic-Shaping Counter Calculations

The FM10000 performs traffic shaping on traffic-shaping groups. The hardware performs the following operations:

- Every unit of time, a user-configurable 1/UB g bytes of credit (tokens) are added to a register specific to the shaping group, called the token bucket.
- Credit is subtracted from the bucket's count when traffic in the shaping group egresses the switch. The amount subtracted is equal to the effective frame length (refer to Section 5.6.4.1).
- When the token count goes to zero or less, all associated traffic classes become shaping limited and therefore ineligible.

- The capacity of the bucket determines the maximum amount of burstiness that is permitted by the scheduler in the shaping group's egress bandwidth profile. Credit stops accumulating in the token bucket once this capacity is reached.

In the FM10000, the token-bucket parameters have the following definitions:

Table 5-5 Scheduler Token Bucket Parameters

Symbol	Name	Scope	Units	Description
T	Time unit	Global	Seconds	The period of the bandwidth shaping sweeper process, equivalent to CM_GLOBAL_CFG.*NumSweeperPorts* x fabric clock period.
C_g	Capacity	Group	Bytes	Token bucket capacity, in units of 64 bytes.
r_g	Rate	Group	Bytes	Number of bytes credited every sweeper period.
R_p	Bandwidth	Port	B/s	Total egress bandwidth of port.
EMTU	Effective MTU	Traffic	Bytes	MTU plus interframe gap and preamble.

With these definitions, the total shaping bandwidth $UB_g = r_g/T$. Furthermore, the maximum burst (MB) the scheduler allows to be transmitted within a shaping group, starting from an idle, fully credited state is as follows:

$$\mathbf{MB = C + EMTU + R*T + r}$$

The time for the switch to transmit this burst is MB/R. The additional EMTU term in the formula is seen in the worst case because the scheduler makes scheduling decisions without regard for the lengths of frames. Therefore, an MTU-long frame could be scheduled with the token count at its minimum positive value (1), causing the count to go negative to -(EMTU-1). This non-ideal condition occurs only on short timescales. The scheduling algorithm guarantees that the UB_g bound is respected on timescales longer than 2*MB/R.

For a fabric frequency of 700 MHz, the FM10000's bandwidth-shaping mechanism supports egress rate limits over a range from 58 KB/s to 29 GB/s with capacity values from 1 KB to 8 MB.

5.6.4.3 Notes on Deficit Round-Robin Scheduling

The original specification of the DRR algorithm (Shreedhar and Varghese, 1995) assumed store-and-forward switching: the length of the current frame is taken into account in the determination whether it can be transmitted or not. However, in the FM10000 as in any other cut-through switch, the next scheduling decision must be made before the length of the frame is known. The DRR implementation therefore must delay its decrement of the scheduled queue's deficit count relative to the standard algorithm. This variant of the DRR algorithm as is referred to as “Delayed Deficit Round-Robin” (DDRR).

Given a set of per-queue DRR quanta Q i (all greater than or equal to 2*MTU), the guaranteed minimum bandwidth of a particular group, as a proportion of the port's total egress bandwidth, can be expressed as follows:

$$\mathbf{LB_i = Q_i \ /\backslash sum\ Q_j}$$

where j runs over the group leaders of that port.

If any Q_i is less than 2*MTU, the minimum bandwidth allotted to each group becomes dependent on the actual frames queued. All Q_i must be set to at least the 192-byte segment size.

5.6.4.4 Timeout Bandwidth Provisioning

Timeout bandwidth is not guaranteed to be fully provisioned in the scheduler; the minimum guaranteed performance is as follows. Frame segments are discarded at maximum segment rate for that port unless that rate is faster than one segment in approximately 15 ns, in which case only one segment per approximately 15 ns is guaranteed.

For example, for a 10 GbE port, the segment polling rate is 67 ns and the time to delete 2000 frame segments would be 134 microseconds (=2000*67 ns). Similarly, for a 100 GbE port, the segment polling rate is 6.7 ns, but the time to delete 2000 frame segments cannot be guaranteed to be less than 30 microseconds (=2000*\max(15 ns,6.7 ns)).

If a single frame is to be replicated multiple times on a given port (e.g., for L3 multicast), the scheduler recalculated the timeout value at each replication, and if the frame thus becomes subject to timeout, it is deleted immediately and further replication is canceled. In other words, replication on a port does not affect timeout bandwidth.

5.6.4.4.1 Discussion of Minimum and Maximum Timeout Period

In the extreme case that the entire frame memory is filled to its maximum capacity with frames enqueued to a single port, the scheduler requires 1.65 ms to drain the memory if the target port is a 10 GbE port. (This time is calculated by multiplying the segment rate mentioned above by the maximum number of segments that can be held in the frame memory, namely, 24 K.) To guarantee reliable timeout, the global time counter must be configured to increment less frequently than this.

At the other extreme, the timeout period is limited only by the bit width of the FRAME_TIME_OUT register (refer to Section 11.14.2.2).

5.7 Frame Processing Pipeline

5.7.1 Overview

The Ingress Crossbar forwards a 128-byte frame header to the Frame Processing Pipeline for forwarding decision. The ingress crossbar sends a frame header to Frame Processing Pipeline whenever there is a probe from the scheduler. This header is empty if there is not a frame header for a particular port. The scheduler also sends a copy of the probe to the Frame Processing Pipeline but using the logical port rather than the physical port.

The Frame Processing Pipeline analyzes the frame header and forwards decisions to downstream units:

- Destinations, multicast pointer, priority and traffic class are forwarded to the RXQ_MCAST.
  - A null destination indicates the frame is dropped.
- Extra new frame information such as new DMAC or new VLAN are forwarded to MODIFY, this new frame information stored away in a Head Storage memory until the actual frame is transmitted for real.

The Scheduler also accumulates the length of segments and forwards a frame trailer information to the Frame Processing Pipeline at the end of each frame. The frame trailer includes the logical port, the frame size and end-of-frame status. The Frame Processing Pipeline use this information for policers, stats and anything depending on frame size and end-of-frame status.

The Frame Processing Pipeline also tracks memory utilization and frees memory when frame transmission is completed, either a single time or multiple times. In doing so, the Frame Processing Pipeline informs the EGRESS/MONITOR unit when PAUSE is received or when shaping is required. The EGRESS/MONITOR units also receive information when frames are transmitted to update its shapers accordingly.

5.7.1.1 Frame Processing Pipeline Components

The frame processing pipeline structure and elements are shown in Figure 5-5.

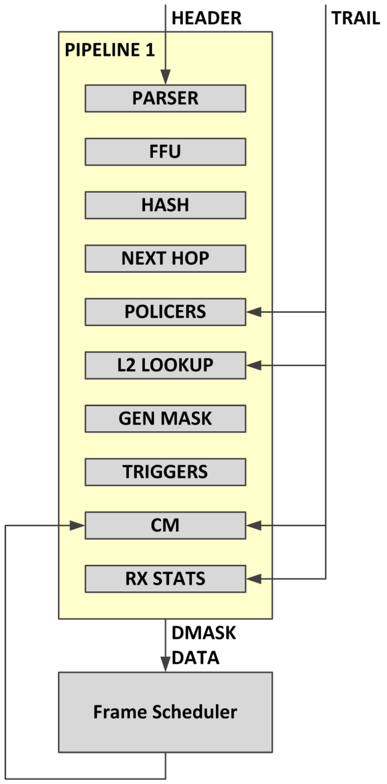


Figure 5-5 Frame Pipeline Overview Block Diagram

The frame header (first 128 bytes) goes through all the stages of the pipeline, the frame trailer (indicating frame length and status) goes through only some stages. The scheduler also returns an indication when the frames are transmitted or timed out or deleted.

The elements of the pipeline are briefly described here:

- **Parser** — Parses the frames. Supports Ethernet, IPv4, IPv6, TCP, UDP and extended deep inspection and assign default values for undefined field.
- **FFU** — Performs frame lookup and associates actions to the frame. This unit includes two sub-units: Mapper and FFU core. The Mapper produces new small keys from some of the extracted fields from the frames, and most of the extracted fields become keys as well, which are all presented to the FFU core for search. The FFU core is a flexible unit organized around a set of TCAM/SRAM slices which, contain rules against which the frames keys are compared. Matching rules are prioritized and actions are assigned to the frames.
- **Hashing Unit** — Performs hash computation for load balancing, link aggregation, and ECMP.
- **Next Hop Lookup** — Defines the new DMAC/VLAN lookup or multicast group ID for IP routing.
- **Policers** — Measures flow rate and associates an action if committed rate or excess rates are exceeded. Policer banks can be re-purposed as counters if desired.
- **L2 Lookup** — Performs DMAC/SMAC lookup and associates a logical destination ID (Destination GLORT) with each entry.

- **Destination Mask Generation** — Generates a destination mask based on destination GLORT, ingress/egress VLAN membership, ingress/egress spanning tree states, egress ACLs, policer results, and various loopback conditions.
- **Triggers** — Applies a set of configurable rules to modify destination mask based on various information collected during earlier stages of the pipeline.
- **Congestion Management** — Tracks shared memory usage and applies drop masks depending on usage.
- **Statistics** — Collects statistics on frames

5.7.1.2 VLAN/GLORT Processing

Figure 5-6 shows the VLAN and GLORT processing through the Frame Processing Pipeline.

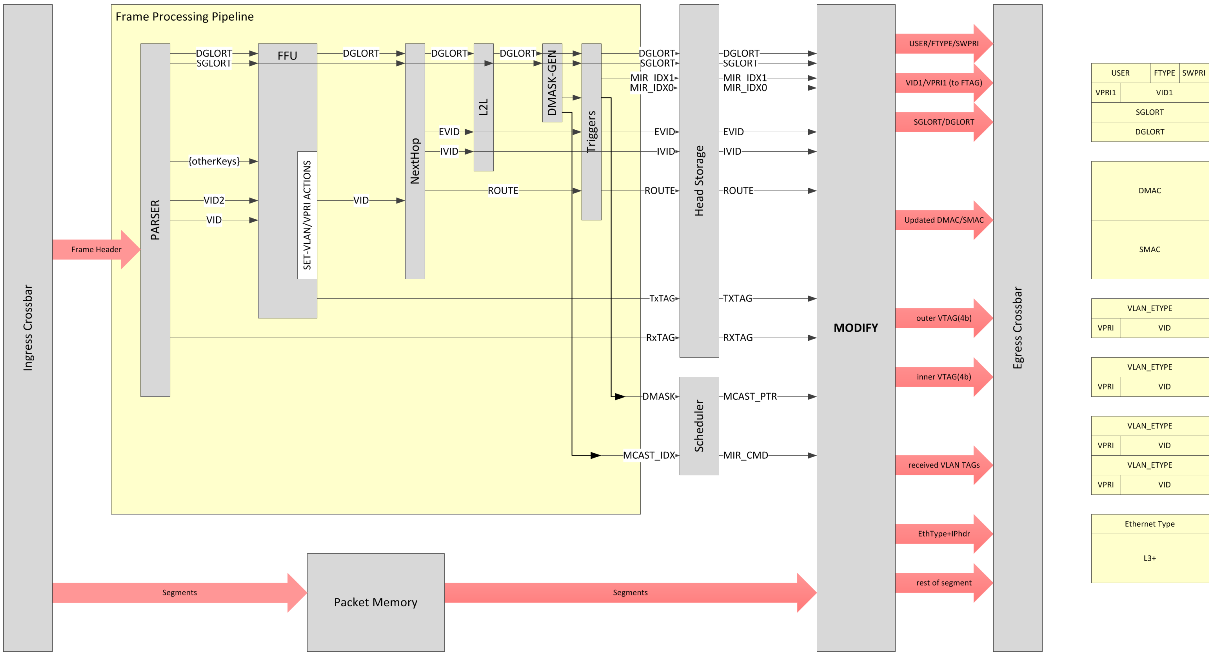


Figure 5-6 VLAN/GLORT Processing

For VID/VPRI processing, the FPP proceeds in the following steps:

Step 1 - PARSER:

The Parser unit extracts VID and VID2 from the incoming packet and forwards these fields to the FFU. The Parser also produces an RxTag sent directly to Head Storage, which provides frame parser information to MODIFY.

Step 2 - FFU:

The FFU unit updates VID based on SET-VLAN actions. The rules leading to the SET-VLAN actions might include any set of keys (for details, refer to Section 5.7.3, “Filtering and Forwarding Unit”). In absence of any rules, {VPRI,VID} remain unchanged. The FFU might generate a TxTag command (0 by default) to instruct the MODIFY unit to override default VLAN tagging instructions.

Step 3 - NEXTHOP:

The FFU might generate a route command to Next Hop. The Next Hop unit preserves VPRI from FFU, copies VID from FFU into an ingress VID (IVID) and, if requested by FFU, performs a lookup into the ARP table which might lead to a route action; route=1, new DMAC/DGLORT or new VID retrieved. The new VID retrieved becomes the egress VID (EVID). If there is no route command executed, the egress VID is equal to the ingress VID.

Step 4 - L2LOOKUP:

The L2L uses the DMAC,EVID and SMAC,IVID to perform a MAC lookup (EVID and IVID are translated to EFID and IFID respectively before the lookup — for details, refer to Section 5.7.7, “Layer 2 Lookup”). The DMAC,EVID is used to determine the destination GLORT (where to go) while the SMAC,IVID is used to detect new or moved MAC addresses.

Step 5 - DMASK-GEN:

The DMASK Generation does not change IVID, EVID or VPRI.

Step 6 - TRIGGERS:

The Triggers have the option to update the EVID on special programmed rules. In absence of rules, the EVID remains unchanged. The IVID is not changed.

Step 7 - HEAD STORAGE/SCHEDULE:

The Head Storage unit stores data generated by the Frame Processing Pipeline until the scheduler schedules the frame for transmission.The EVID and IVID are not changed.

Step 8 - MODIFY:

The MODIFY unit uses the data from Head Storage, indications from Scheduler (multicast pointer, mirror command) and the original segment data to determine the new VLAN egress tags. It typically reverts the egress VID (EVID) into a {VID1,VID2} set and follows rules defined in the MODIFY unit to determine which VLAN tags are present. As a reminder, the EVID is equal to IVID for non-routed frames (unless the Triggers unit has changed the EVID).

The DGLORT/SGLORT processing follows the following steps:

SGLORT:

Set by the Parser (from incoming FTAG or a default value assigned to the port in absence of FTAGs). It is not changed in the frame processing pipeline.

DGLORT - STEP 1 - PARSER:

Set by parser, from incoming FTAG or set to 0b in absence of FTAG.

DGLORT - STEP 2 - FFU:

The FFU might assign a SET-GLORT or SET-ROUTE action.

DGLORT - STEP 3 - NEXTHOP:

The NextHop uses the command from FFU to do a lookup and update the DGLORT if the command implies a DGLORT setting. The DGLORT remains unchanged if there are no command to change it.

DGLORT - STEP 4 - L2LOOKUP:

The L2LOOKUP does a DMAC,EVID lookup. If there is a hit, the entry provides a new DGLORT. If it is a miss, the unit assigns a default DGLORT (Flood DGLORT) if the DGLORT is 0. Refer to Section 5.7.7.6 on DGLORT setting.

DGLORT - STEP 5 - POLICERS:

The Policer unit has no effect on DGLORT.

DGLORT - STEP 6 - DMASK-GEN:

The DMASK Generation does not change the DGLORT but uses it to create DMASK and the IP multicast index.

DGLORT - STEP 7 - TRIGGERS:

The Triggers have the option to update the DGLORT and DMASK.

DGLORT - STEP 8 - HEAD STORAGE/SCHEDULER:

Stores the SGLORT/DGLORT until the frame is actually scheduled for transmission. Actual data is not changed.

DGLORT - STEP 9 - MODIFY:

The MODIFY unit uses the SGLORT/DGLORT from Head Storage and indications from Scheduler (multicast pointer, mirror command) to update the DGLORT. The DGLORT is not updated unless it is a mirror command.

5.7.1.3 Frame Priority Processing

The switch supports four priority fields:

- VPRI — VLAN priority from a first VLAN tag.
- VPRI2 — VLAN priority from a second VLAN tag.
- SWPRI — Switch priority, used by congestion management.
- DSCP — IP differentiated services code point.

Figure 5-7 shows the priority processing through the pipeline.

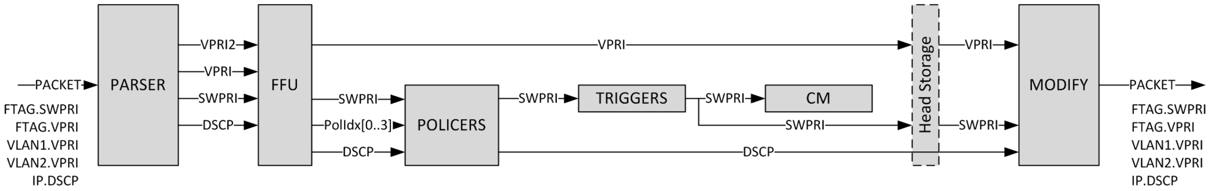


Figure 5-7 Frame Priority Processing

For VPRI processing, the FPP proceeds in the following steps:

Step 1 - PARSER:

The Parser unit extracts VPRI, VPRI2, SWPRI and DSCP from the incoming frame and forwards these fields to the FFU. The parser assigns default values if the frames do not contain those fields or overrides them if desired. For details on priority field handling, refer to Section 5.7.2, "Parser".

Step 2 - FFU:

The FFU can be used to update VPRI, SWPRI or DSCP by associating SET-VLAN/SET-VPRI/SET-DSCP actions to user programmed frame rules. The FFU can also be used to associate policers to frames rules, the policers indexes are sent to the POLICERS unit for the winning rules.

The FFU can use VPRI and VPRI2 as frame keys but only forward VPRI (updated or not), the field VPRI2 is discarded.

Step 3 - POLICERS:

The policers use the indexes supplied by FFU to monitor incoming bandwidth and apply a down map to downgrade the SWPRI or DSCP field or both if the committed or excess rate are exceeded. The policers cannot be used to change the VPRI.

Step 5 - TRIGGERS:

The Triggers have the option to update the SWPRI on special programmed rules. In absence of rules, the SWPRI remains unchanged. The triggers cannot be used to change VPRI or DSCP.

Step 6 - HEAD STORAGE/SCHEDULE:

The Head Storage unit stores data generated by the Frame Processing Pipeline until the scheduler schedules the frame for transmission. The SWPRI is not changed.

Step 7 - MODIFY:

The MODIFY unit uses the data from Head Storage, indications from Scheduler (multicast pointer, mirror command) and the original segment data to determine the new VLAN egress tags. It typically reverts the egress VPRI into a {VPRI1,VPRI2} set, set the SWPRI into FTAG if this tag is present, and, finally, update DSCP if updating this field is allowed and the frame is IP.

5.7.1.4 Action Codes

The Frame Processing Pipeline maintains a list of action codes as it processes each frame. At the end of the pipeline (prior to trigger, LAG filtering rules and congestion management), the switch applies the highest precedence action. For a list of action codes, refer to the TRIG_APPLY *AMASK* enumerated data type in Section 11.17.2.

For each action code, an unknown layer 2 source MAC address may or may not be reported in the TCN FIFO for generating a LEARN event (the triggers have the capability to override the default learning behavior). Lower action code values have higher precedence than higher values.

Note that after this step, the frame then goes through the following final 3 steps (in order):

1. Triggers which may override any prior decision.
2. LAG pruning and filtering.
3. Congestion management.

5.7.1.5 Trap Codes

For trapped frames, one of the codes in Table 5-6 is used as the lower 8 bits of the DGLORT, with the upper 8 bits of the DGLORT coming from the TRAP_GLORT register.

```
DGLORT = (TRAP_GLORT & 0xFF00) | trap_code
```

The DGLORT can be used by the Switch Manager to identify the reason the frame was trapped.

Table 5-6 Trap Codes

Value	Usage
0x00..0x3F	Frame trapped due to trigger action. The value is the trigger number that caused the trap.
0x80	Frame trapped by an FFU rule.
0x83	IEEE Reserved MAC address frame trapped.
0x86	IGMP frame trapped.
0x90	ICMP frame with TTL <= 1
0x91	Frame with IP option(s).
0x92	Frame matches CPU MAC address.
0x94	Frame with an MTU too big for egress VLAN.
0x96	Any frame with TTL <= 1

5.7.1.6 Log Codes

For logged frames, one of the following log codes is used as a mirror 0 index profile. This index is used in MODIFY (MOD_MIRROR_PROFILE_TABLE[0][code]) to recover the egress DLORT or VID to use when the frame egresses the switch on the CPU port. Refer to Section 5.8 for mirror indexing.

The log codes are listed in Table 5-7.

Table 5-7 Log Codes

Value	Usage
0x00..0x3F	Frame logged due to trigger action. The value is the trigger number that caused the log.
LOG_MIRROR_PROFILE.*FFU*	Frame logged to the CPU as a result of FFU action.
LOG_MIRROR_PROFILE.*ResrvedMAC*	IEEE Reserved MAC address frame logged.
LOG_MIRROR_PROFILE.*ARPRedirect*	Logged ARP redirect (unicast frame routed back to ingress VLAN.
LOG_MIRROR_PROFILE.*ICMP*	Multicast ICMP frame was logged to the CPU because its TTL was <= 1.
LOG_MIRROR_PROFILE.*TTL*	Any multicast frame logged to the CPU because its TTL was <= 1.

5.7.2 Parser

5.7.2.1 Overview

The parser is the first stage of the frame processing pipeline. The overall context for this unit is shown in Figure 5-8.

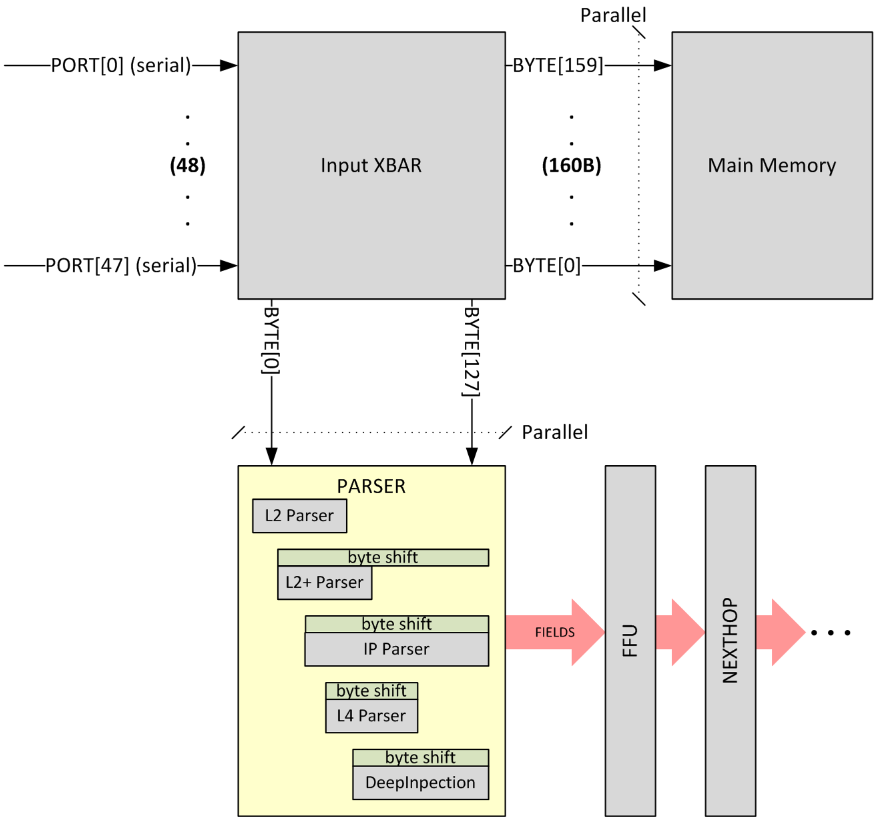


Figure 5-8 Parser Overview

The input crossbar receives data serially from any of the 48 ports and assembles them into 192-byte segments which are written in parallel to the main memory. The first 128 bytes of the first segment are also sent to the parser which parses the frames and extracts the fields needed for routing and switching decisions.

The frame formats supported are shown in Figure 5-9.

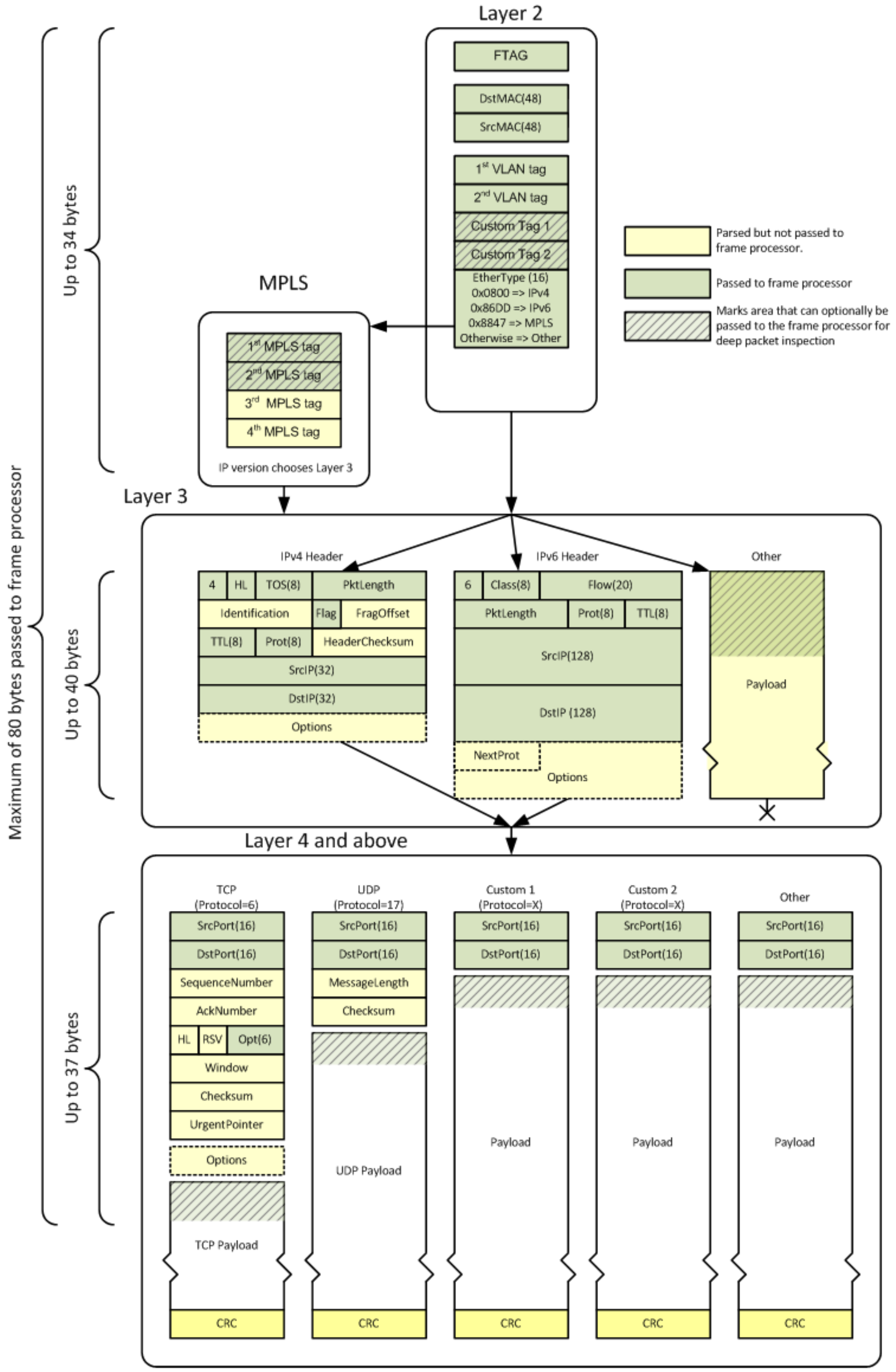


Figure 5-9 Frame Parsing

5.7.2.2 FTAG

The FTAG is an optional field at the head of the Ethernet frame; presence is configurable per port. If enabled, all Ethernet frames coming or going to that port are expected to have the FTAG present.

The FTAG is used when multiple switches are aggregated together using Ethernet ports to create a larger logical switch. The FTAG carries extra information about the packet. The FTAG is also used in providing extra information to the PCIe interfaces and, ultimately, to the hosts attached to those interfaces.

The FTAG is 8 bytes long; it is designed to replace the preamble and SFD of an Ethernet frame. The first byte is not available on Ethernet ports; it is overwritten by the Start control character on transmission and is replaced by a port-configurable constant on reception. The remaining 7 bytes are available. The first byte is available on PCIe interfaces to allow the hosts to receive or transmit an extra 8 bits of information. This extra 8 bits are presented as a key to the FFU (host to fabric direction), and the FFU can change/add a new 8-bit user field (fabric to host direction).

The location and format of the FTAG is shown in Figure 5-10.

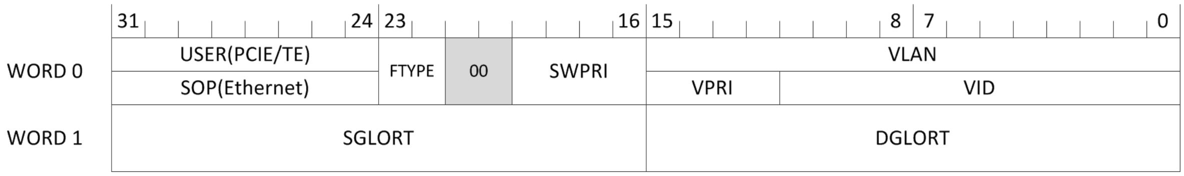


Figure 5-10 FTAG Location and Format

The FTYPE is:

00b = Normal frame
10b = Special frame

FTYPE=11b (reserved) is not present (or received with VID1=0) x1: VID present (VID1 != 0) 0x: VID2 allowed to be used, the parser marks the frame for dropping.

The FTYPE=01b is reserved for future use and presented to FFU, the frame is not present (or received with VID2=0) 1x: VID2 present (VID2 != 0) marked for dropping.

The VPRI/VID is the mandatory VPRI and VID attached to the frame from a multi-chip purpose.

The DGLORT and SGLORT are destination and source global resource tags.

5.7.2.3 Layer 2 Processing

The layer 2 processing is as follows, in order:

- **Destination MAC address** — Always forwarded to the frame handler.
- **Source MAC address** — Always forwarded to the frame handler.
- **VLAN tag processing** — Up to two VLANs supported. Only two supported for routing.
- **Custom Ethertype Tags** — Up to 2 custom Ethertypes.
- **MPLS tag processing**.

5.7.2.3.1 VLAN Processing

A VLAN tag consist of the following fields, as shown in Figure 5-11.

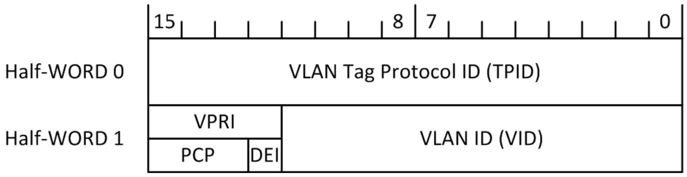


Figure 5-11 VLAN Tag Format Definition

The VPRI field encompasses the PCP (3 bits, Priority Code Point) and the DEI (1 bit, Discard Eligible Indicator). The FM10000 manages both the PCP and DEI as a single 4-bit entity (VPRI), with the option, on egress, to preserve the DEI bit received.

Note: The bit 12 of the second word is the Discard Eligible Indicator (DEI) bit and is defined in 802.1Q-2011.This same bit was previously known as a Canonical Format Indicator (CFI) bit in 802.1Q-2008 and previous versions. The FM10000 follows 802.1Q-2011 interpretation (DEI). It can provide transport of CFI but does not support addresses in canonical format.

The VLAN stacking options are shown in Figure 5-12.

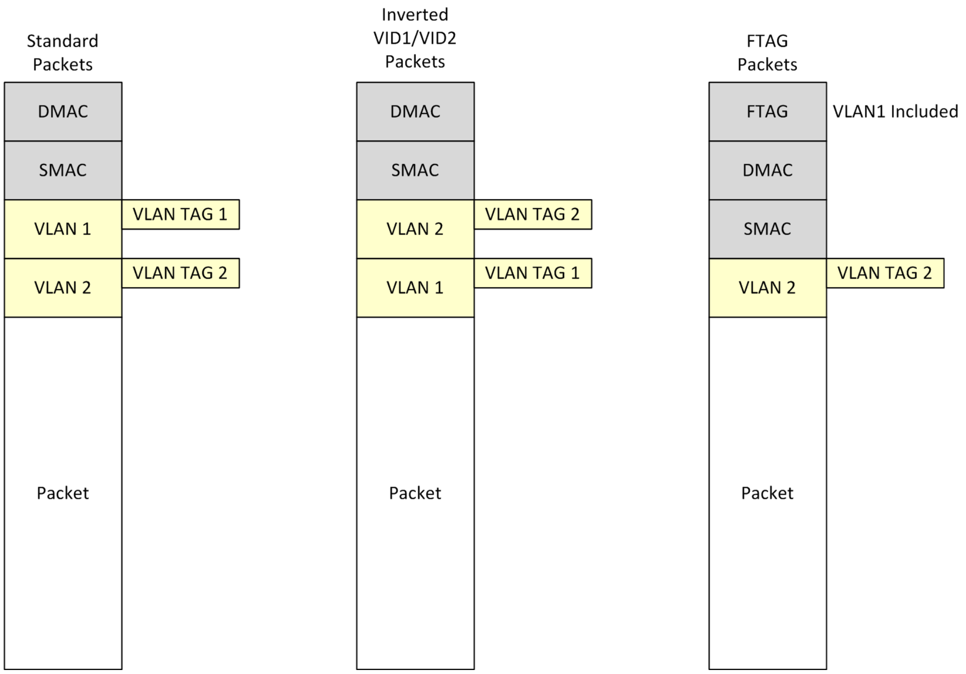


Figure 5-12 Parser VLAN Tagging

The parser supports four configurable global VLAN Tag Protocol ID values, which are defined in PARSER_VLAN_TAG register. The PARSER_PORT_CFG_1 register then defines which tag values are used to recognize VLAN1 and VLAN2 in a receive frame for each port. This allows different ports to use different tags. If the tag values appearing in each tag position in a frame are valid for both VLAN1 and VLAN2, then PARSER_PORT_CFG_1 defines which tag is considered VLAN1 and which VLAN2.

If VLAN1 or VLAN2 tags are absent, default VPRI/VID and VPRI2/VID2 are assigned. If the VLAN1 or VLAN2 is present but the VID or VID2 supplied is zero, a default VID or VID2 is assigned respectively. Default VIDs and VPRIs are configurable per port. An option allows the default VPRI/VID to be imposed regardless of what the frame includes. However, the VPRI2/VID2 default values cannot be imposed.

If more than two VLAN tags are present in the frame and valid, the parser only processes the first two VLAN tags, and then proceeds in comparing the following tag against the custom tags or MPLS tags.

The switch has a feature to drop frames with VLAN tags (PARSER_PORT_CFG_2[port].*dropTagged*) or drop frames without VLAN tags (PARSER_PORT_CFG_2[port].*dropUntagged*). Setting both to 0b means no frames are dropped due to the presence or absence of VLAN tags, while setting both to 1b means all frames are dropped. A frame is considered VLAN tagged if and only if the frame was received with at least one VLAN tag present with a VID different than 0b. Otherwise the frame is considered VLAN untagged.

Note: The parser outputs VPRI/VID and VPRI2/VID2. The VPRI/VID is associated with VLAN1 while VPRI2/VID2 is associated with VLAN2.

The parser supports also custom tagging and MPLS tagging. This is shown in Figure 5-13.

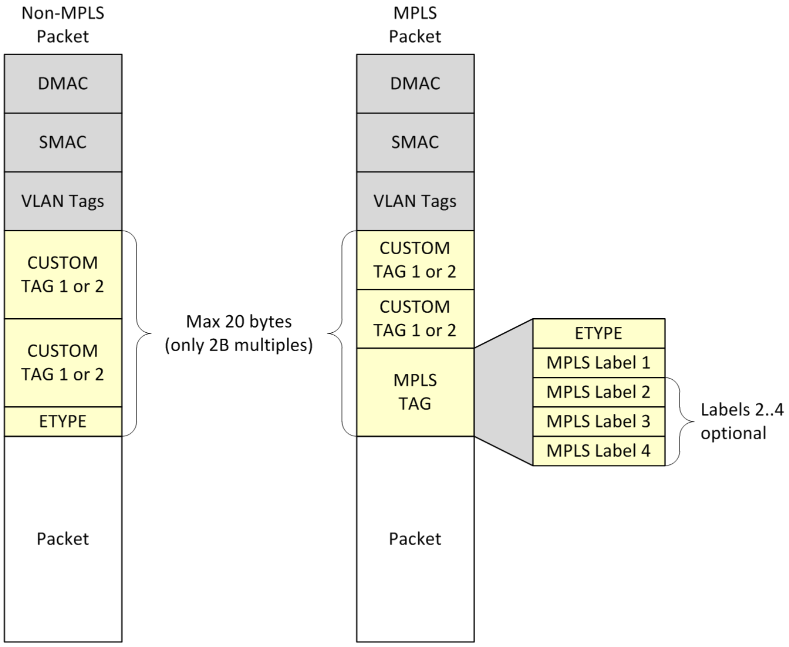


Figure 5-13 Parser CUSTOM and MPLS Tagging

Two custom tags are supported after VLAN1 and VLAN2 tags. The parser has four configurable global custom tags, those are defined in PARSER_CUSTOM_TAG register. The PARSER_PORT_CFG_2 register (fields *CustomTag1* and *CustomTag2*) then defines which tag values are active per port. It is assumed that the custom tags are between the VLAN tags and the MPLS tags. Each custom tag can be present or absent; the parser recognizes any ordering if both present.

The *CustomTag1* and *CustomTag2* must be configured to avoid ambiguity. Essentially, the *CustomTag1* and *CustomTag2* must be non-overlapping, all PARSER_CUSTOM_TAG[0..3].*Tag* must be different from each other, and all must be different from the standard Ethertypes 0x0800 (IPv4), 0x86dd (IPv6), 0x8808 (MAC Control), or the MPLS Ethertypes configured in PARSER_MPLS_TAG (the hardware behavior is undefined if these rules are not followed by software). The content of each custom tag is optionally captured in deep inspection fields L4A..L4D. Which 16-bit word is captured is configurable per custom tag.

If there are more than 2 custom tags present, only the first two are compared against the custom tags, and the following one is compared against MPLS tag processing. The custom tags must fit within 18 bytes. If the first two custom tags total more than 18 bytes, the packet is flagged with a parsing error.

The MPLS tag processing follows the custom tag processing. The MPLS tag is recognized by matching either of the EtherTypes in PARSER_MPLS_TAG. The first two label stack entries of the MPLS tag can optionally be saved in L4A..L4D, if they fall entirely within the 20 bytes past the SMAC or VLAN tags. Label stack entries that fall partly or entirely outside 20 bytes are treated as 0x0. The MPLS tag parsing stops when the bottom-of-stack bit is set (BS bit below).

0 19	20 22	23	24 31
LABEL	QOS	BS	TTL

=> Network bit ordering

If an MPLS tag is present, it is assumed that an IP header follows without any Ethernet type present (ETYPE is then set to the MPLS tag type). The IP version of the IP header (the first 4 bits following the MPLS tag) is used to determine the IP type (IPv4 or IPv6). If this number is not 4 or 6, the frame is treated as a non-IP frame.

If the combined length of custom tags + MPLS tag is 20 bytes or less, parsing continues to Layer 3 processing. If not (the MPLS tag extends more than 20 bytes past the SMAC or VLAN tags), no further parsing is done, and the frame is treated as an L2 frame.

5.7.2.3.2 Layer 2 Tagging Information

The presence of the different tags in an Ethernet frame is passed to the FFU as a KEY, and to MODIFY via an 8-bit RXTAG field. The application software might use this information to condition rules on presence or absence of a given tag. The MODIFY unit uses this information during Ethernet fame modifications (add/remove/change the FTAG, add/remove/change VLAN tags and apply routing modifications).

The RXTAG encoding is as follows:

Bit	Description
0	FTAG present or not. 0b = FTAG not present. 1b = FTAG present.
1	VLAN1 tag present or not. 0b = VLAN1 tag not present. 1b = VLAN1 tag present. This bit is mutually exclusive to FTAG presence bit, and is bit is set to 0b if FTAG is present.
2	VLAN2 tag present or not. 0b = VLAN2 tag not present. 1b = VLAN2 tag present.
3	VLAN2 first. Set only if both VLAN1 and VLAN2 are present and VLAN2 was detected to be before VLAN1.

Bit	Description
4	IPV4 header detected. This bit is not set if IPv4 present but custom tags + EtherType or MPLS tag totals more than 20 bytes.
5	IPV6 header detected. This bit is not set if IPv6 present but custom tags + EtherType or MPLS tag totals more than 20 bytes.
6	MPLS tag present.
7	At least one custom tag present.

5.7.2.4 Pause Frame

A frame is treated as a valid pause frame if all of the following conditions are met:

- The frame's DMAC must be either the MAC_CTRL DMAC or must match PORT_CFG_3.*DMAC*.
- The MAC_CTRL Ethertype must be present (0x8808).
- The MAC_CTRL opcode must be either normal pause or class-based pause.
- The frame has no VLAN tags, MPLS tags, or custom tags.
- The frame was received without tail errors.
- The frame length is at least 64 bytes, and at most 192 bytes.
  - The minimum frame length is checked by the EPL, which marks any frame smaller than minimum frame size (default 64 bytes) with a tail error.

A PAUSE frame received on an FTAG-ed port is expected to have an FTAG of all zeros. No validation of this is done in the parser.

5.7.2.5 Layer 3 Parsing

Layer 3 parsing is enabled only if PARSER_PORT_CFG_2[port].*ParseL3* is enabled. If layer 3 parsing is disabled, the parser treats any IP frame as an L2 frame.

The following applies only if layer 3 is enabled.

If the frame is not a pause frame, the FM10000 detects the presence of an IPv4 or IPv6 header by comparing the Ethernet type to the known values for IPv4 and IPv6 Ethernet types and by checking the version field (first 4 bits of the layer 3 header). If the EtherType is 0x0800 (and the IP version is 4), the frame is parsed as IPv4 frame. If the EtherType is 0x86dd (and the IP version is 6), the frame is parsed as IPv6. If the EtherType is either 0x0800 or 0x86dd, but the IP version does not match, the packet is flagged with a parsing error.

If the Ethernet frame is MPLS tagged, the frame is assumed to contain an IP packet, and the first nibble is inspected to identify the version (IPv4 or IPv6). If the version is neither 4 or 6 (or if there is no payload following the MPLS label stack), the frame is treated as an L2 frame, and layer 3 is not parsed.

Note: The FM10000 does not support LLC or SNAP encoded Ethernet frames.

The port may be configured to always stop parsing after layer 2, but when layer 3 parsing is required, it proceeds as follows:

IPv4:

- The parser always forwards the version, header length, TOS/DS field, packet length, TTL, layer 4 protocol, destination IP and source IP addresses to the frame processor.
- The parser also checks the presence of header options by checking the Internet Header Length (IHL) field, stored in bits 4-7 of the IPv4 header. IHL is the length of the IPv4 header in 32-bit words, including options. The minimum value of IHL is 5, indicating no options; if IHL > 5, then options are present. It is not necessary to parse the contents of the IPv4 options. The parser skips over all options.
- The parser also checks if the packet is a fragment, and if it is the first fragment. If it is the first fragment, the layer 4 is processed. Otherwise, the parsing stops. The parser detects if it is the first fragment by checking that IPv4 fragment offset in bits 51-63 of the IPv4 header is 0b.
- If there is an error during decoding, the packet is flagged as having a parse error, and the frame is discarded and counted. Note that a trigger action could override the default disposition and possibly trap the packet to the local processor for further processing. Packets that have a parse error are never routed.

IPv6:

- The parser always forwards the class, flow, payload length, Next Header (protocol), Hop Limit (TTL), destination and source IP addresses to the frame processor.
- The parser is also capable of detecting the presence of options in the packet by comparing the protocol field to the known options and skipping over the options if they are present. The parser stops after the current header if the protocol in the next header does not match one of the following:

  0x00 = IPv6 Hop-by-Hop Option (length 8N+8 bytes).
  0x2b = Routing Header for IPv6 (length 8N+8 bytes).
  0x2c = Fragment Header for IPv6 (length 8 bytes; N is always 0).
  0x3c = Destination Options for IPv6 (length 8N+8 bytes).
  0x33 = Authentication Header (length 4N+8 bytes -- not the same as the other options).

- The parser stops after 128 bytes and sets the protocol to 0xFF if the protocol is not found by the time it reaches this limit.
- The parser also checks if the packet is a fragment, and if it is the first fragment. If it is the first fragment, the layer 4 is processed. Otherwise, the parsing stops. If an IPv6 packet is fragmented, it carries a Fragment Header option. If it has a Fragment Header, and the fragment offset (bits 16-28 of the Fragment Header option) is non-zero, this is not the first fragment.
- If there is an error during decoding, the packet is flagged as having a parse error, and the frame is discarded and counted. Note that a trigger action could override the default disposition and possibly trap the packet to the local processor for further processing. Packets that have a parse error are never routed.

If the IP packet is fragmented, and this is not the first fragment, parsing terminates at this point because the layer 4 header is not available. The parser verifies the IPv4 fragment offset is in bits 51-63 of the IPv4 header; if the fragment offset is non-zero, this is not the first fragment.

If an IPv6 packet is fragmented, it carries a Fragment Header option. If it has a Fragment Header, and the fragment offset (bits 16-28 of the Fragment Header option) is non-zero, this is not the first fragment.

The Parser reports the fragmentation information to the FFU as well as presence of options through an IPMISC field:

- Bit 0: Presence of option or not.
- Bit 1: If fragmentation allowed.
- Bit 2: Head fragment (fragment offset == 0).
- Bit 3: More fragment (MF bit).

The port may also be configured to always stop parsing after layer 3, regardless of fragmentation. Any layer 4 fields that are either not parsed or not present due to fragmentation are treated as '0' in the frame processing pipeline.

5.7.2.6 Layer 4 Parsing

Layer 4 parsing is enabled only if PARSER_PORT_CFG_2[port].*ParseL4* is enabled. If layer 4 parsing is disabled, the parser treats the packet as an L3 packet only.

If layer 4 processing is enabled, the switch checks for the presence of a layer 4+ payload if and only if the packet payload length is non-zero. The protocols recognized by the switch are TCP (protocol = 0x06) and UDP (protocol = 0x11). Any other protocol is assumed to have L4SRC/L4DST as the first four bytes and checked to have a length of at least 4 bytes.

Note: The parser parsing depth is limited to 128. If either the L4SRC and L4DST have not been fully received by the 128th byte, they are both set to 0. If PARSER_DI_CFG.*CaptureTCPFlags* is set, and the TCP flags have not been fully received by the 128th byte, L4A is set to 0xFFFF.

5.7.2.7 Layer 2/3/4 Parsing Error

A parsing error indicates that a packet is malformed. The parser declares a parsing error if any one of the following conditions is met:

L2 parsing errors:

- The L2 header is found incomplete because the packet is too small.
- The custom tags total more than 18 bytes.

L3 parsing errors, if L3 parsing is enabled and no parsing error at L2:

- The IP header is found incomplete because the packet is too small.
- The Ethertype is IP, but IP version does not match Ethernet type.
- The IPv4 checksum is incorrect.
- The IPv4 header length (IHL) is less than 5.
- The packet is TCP, and the fragment offset is 1.
- The length of the IPV6 authentication header is not a multiple of 8 bytes.

L4 parsing errors, if L4 parsing is enabled and no parsing errors at L2 or L3:

- The TCP or UDP header is found incomplete because the packet is too small.
- The TCP data offset is less than 5.

PAUSE parsing error:

- The frame meets all the requirements for a valid pause frame (refer to Section 5.7.2.4) except for length and possibly tail error, but the parsed frame is not long enough to include the entire pause payload (4 octets after EtherType for normal pause, 20 octets after EtherType for class-based-pause).

The parser stops at the stage the error is encountered and any field expected to be extracted at that stage or follow-on stages are undefined. For example, if parsing fails at layer 3, then the layer 3 & 4 are undefined.

Normal frames (FTYPE=NORMAL) with parsing errors are dropped and counted as ParserErrorDrops (refer to Section 5.7.11, "Statistics and Monitoring") unless the trigger unit reverses the disposition. Frames with FTYPE=SPECIAL are not dropped even if they have parsing error. For priority order, refer to Section 5.7.1.4, "Action Codes".

5.7.2.8 Deep Inspection

5.7.2.8.1 For IP Frames

The switch assumes the presence of a layer 4+ payload if the packet length is non-zero. The protocols recognized by the switch are TCP (protocol = 0x06) and UDP (protocol = 0x11). Any other protocol is assumed to have L4SRC/L4DST as the first four bytes and assumed to have a length of 4 bytes. Note that if the frame is IPv6, the deep inspection does not overwrite the upper bits of the SIP and DIP, the parser just stops after L4D.

The PARSER_DI_CFG defines six configurable filters to define how to capture data from certain packets. The filters indicates protocol and word offsets to capture for each protocol in each DI fields. The protocol is defined as PROT field with optional match on L4SRC or L4DST. The filter 0 is selected if all other five filters are no match. If more than one filter match, the highest number takes precedence.

The pseudo code for search is as follows:

```
m = 0
foreach i (1..5) {
    if ( PARSER_DI_CFG[i].Enabled == 1 &&
         PARSER_DI_CFG[i].Protocol == PROT &&
         (PARSER_DI_CFG[i].L4Port == L4_SRC ||
          PARSER_DI_CFG[i].L4Port == L4_DST ||
          PARSER_DI_CFG[i].L4Compare == 0) )
        m = i
}

if ( PROT == UDP )
    l4HeaderLength = 2
else if ( PROT == TCP )
    l4HeaderLength = (TCP header HL field or 5 if HL is not
                      visible within the 128B parsing depth)
else
    l4HeaderLength = 1

mux[0..7] = 0x1F;   // Means disable capture
captureTCPFlags = PARSER_DI_CFG[m].CaptureTCPFlags
foreach i (0..7) {
    if ( PARSER_DI_CFG[m].WordOffset[i] != 15 )
        mux[i] = l4HeaderLength + PARSER_DI_CFG[m].WordOffset[i];
}

if ( PARSER_DI_CFG[m].CaptureTCPFlags ) {
    L4A = TCP Flags;
}
```

5.7.2.8.2 For Non-IP Frames

If layer 3 parsing is not performed, the switch captures the first 32 bytes of the frame payload after the Ethernet type or MPLS tag into the SIP/DIP fields. This capture is not performed when the custom tags and MPLS tag total more than 20 bytes; in that case SIP/DIP are undefined.

Note: If the packet is MPLS tagged, the MPLS label stack is included in the captured data.

Half-word	Mapped to FFU	Half-word	Mapped to FFU
0	SIP[127:112]	8	DIP[127:112]
1	SIP[111:96]	9	DIP[111:96]
2	SIP[95:80]	10	DIP[95:80]
3	SIP[79:64]	11	DIP[79:64]
4	SIP[63:48]	12	DIP[63:48]
5	SIP[47:32]	13	DIP[47:32]
6	SIP[31:16]	14	DIP[31:16]
7	SIP[15:0]	15	DIP[15:0]

5.7.2.9 Architecture

The architecture of the block is shown in Figure 5-14.

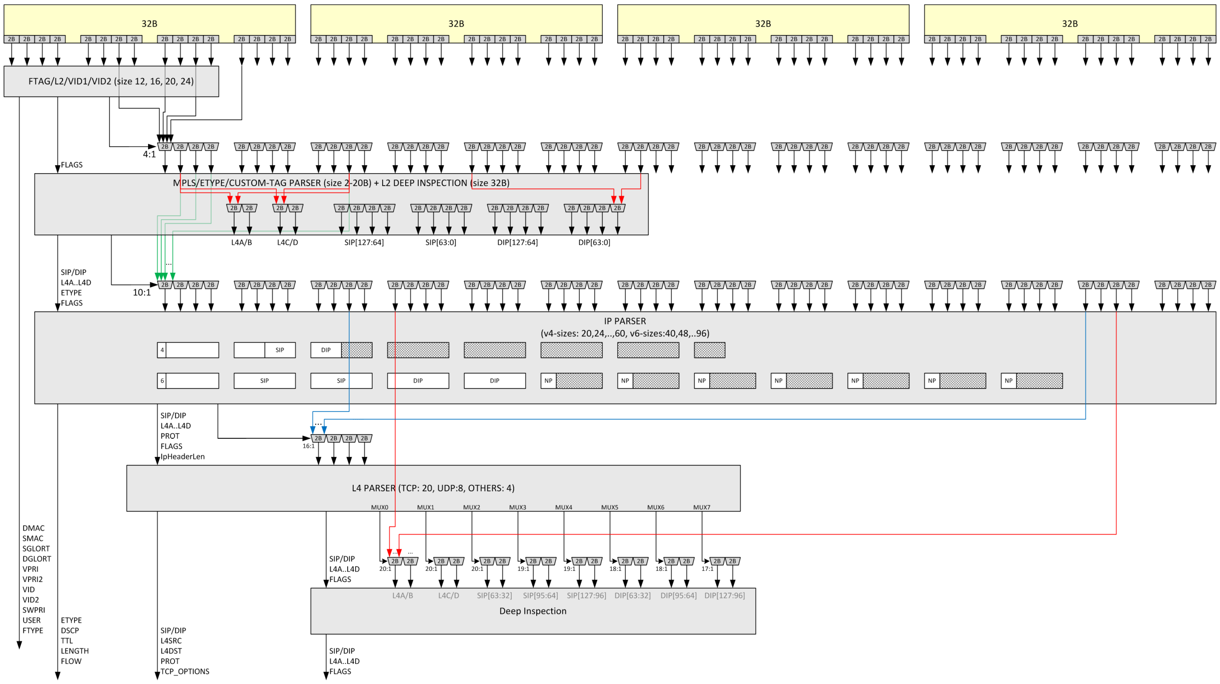


Figure 5-14 Parser Architecture

The parser is parallel and divided into 5 stages:

- Stage 1: Basic L2 parsing
- Stage 2: Extra L2 parsing
- Stage 3: IP parsing
- Stage 4: L4 parsing
- Stage 5: Deep inspection for IP

The first stage receives all the 128 data bytes directly and extracts FTAG, DMAC, SMAC, VID1, VID2. There are four different possible frame sizes:

- DMAC + SMAC => 12-bytes
- DMAC + SMAC + 1 x VLAN => 16-bytes
- FTAG + DMAC + SMAC => 20-bytes
- DMAC + SMAC + 2 x VLAN => 20-bytes
- FTAG + DMAC + SMAC + VID2 => 24-bytes

The first stage extracts the FTAG, DMAC, SMAC, VID1, VID2 and a shift to indicate how to shift data to align the next header.

The second stage includes a 4-way shifter and walks through two custom tags, MPLS tag and ETYPE. The content of the tags is potentially passed to deep inspection fields L4A...L4D. The size of each custom tags is configurable, the size of the MPLS tags is variable, MPLS tag + up to 4 labels. The total length of these tags does exceed 20 bytes, and length varies between 2 bytes and 20 bytes in 2 bytes increment. Then the next 32 bytes following the tags is stored in SIP/DIP for deep inspection. This stage extracts L4A..L4D and SIP/DIP. This parser stage does not specify a priority order for L4A..L4D capture if a packet has both custom tags and MPLS tags and software has enabled L4A..L4D field capture for both type of tags.

The third stage is IP parsing. It starts with a 10-way shifter and walks through the IP header extracting SIP, DIP, PROT, TTL, LENGTH, TOS and miscellaneous fields. The size of this header is up to the maximum for IPv4 header (15 words) and up to 96 bytes for IPv6 header (IPv6 header size is unbounded per spec but limited to 96 bytes in the FM10000). The IP parser produces a 4-bit shift for next stage which represent the size of the header possibles.

The fourth stage is L4 parsing. It starts with a 16-way shifter to align the data to L4 header, and extracts source and destination port and, for TCP, extracts the length and TCP options. The TCP header is between 5 words and 15 words long. Also, the L4 stages products a mux output for each word for deep inspection, last stage of the parser. The value of the mux output depends on the configuration of the deep inspection and consist of an adder for L4 header size and 32-bit word offset desired for the protocol detected. The pattern recognized is defined in the PARSER_DI_CFG registers.

The last stage is the deep inspection. It has 8 x 32-bit shifters. The size of each is variable due to the size limit of the header. The deep inspection does not capture a field if its corresponding mux value is higher than the maximum possible, it does not change the SIP[127:32] and DIP[127:32] if the packet is IPv6 and does not change any of the deep inspection field previously captured at second stage if the frame is not IP. If any deep inspection field is enabled and capture at this stage, it replaces the one captured previously at the second stage.

5.7.2.10 Default Values and Priority Mapping

The parser assigns default values as follows:

- A default VID1 is assigned if the Ethernet frame has no VLAN1 tag, or the VID it contains is 0, or if the port is configured to ignore the VID received (*useDefaultVlan*=1b).
- A default VID2 is assigned if the Ethernet frame has no VLAN2 tag, or the VID2 it contains is 0.
- A default VPRI1 is assigned if the Ethernet frame has no VLAN1 tag, or if the port is configured to ignore the VLAN tag received (*useDefaultVLAN*=1b).
- A default VPRI2 is assigned if the Ethernet frame has no VLAN2 tag.
- A default SGLORT is assigned if the port is not Fabric tagged.
- A default DSCP is assigned if the port is configured to ignore the DSCP received (*useDefaultDSCP*=1b) or frame is not IP.

The default values are applied after the proper ordering for VLAN tagging is applied. For example, if the VID2 is the outer tag and VID1 is the inner tag, the default VID2 is applied to the outer tag and the default VID is applied to the inner tag.

The VLAN, DSCP and switch priority are derived as follows:

VPRI

The RX VLAN user priority (including the DEI bit) is mapped into a 4-bit internal VPRI if present, using the RX_VPRI_MAP register. A per-port configurable VLAN default priority field (PARSER_PORT_CFG_1.*defaultVPRI*) is used to assign a VPRI if the frame does include a VLAN priority tag. This default priority is also used if the PARSER_PORT_CFG_1.*useDefaultVLAN* is set.

DSCP

The DSCP is extracted from the IP header if PARSER_PORT_CFG_2[p].*useDefaultDSCP* is set to 0b, the frame is IP, and IP decoding is enabled. The DSCP is set to PARSER_PORT_CFG_2[p].*defaultDSCP* if PARSER_PORT_CFG_2[p].*useDefaultDSCP* is set to 1b, regardless of the type of frame received. The DSCP is set to 0 if PARSER_PORT_CFG_2[p].*useDefaultDSCP* is set to 0b, and the frame is not IP or IP decoding is not enabled.

Derive Switch Priority

The internal switch priority can be assigned from 4 sources, controlled by PARSER_PORT_CFG_2:

- If the frame is Fabric tagged, and the *SwitchPriorityFromISL* bit is true, the switch priority is set to the ISL Priority field.
- Otherwise, if the frame is IP, and the *SwitchPriorityFromDSCP* bit is true, the switch priority is retrieved from the global DSCP_PRI_MAP register, which maps the 64 possible DSCP codes received into a 4-bit switch priority code.
- Otherwise, if the frame has a VLAN 1 Tag, and the *SwitchPriorityFromVLAN* bit is true, the switch priority order: is retrieved from the global VPRI_PRI_MAP table, which maps the 16 possible VPRI code received into a 4-bit switch priority level.
- Otherwise, the per-port PORT_CFG_ISL.*defaultPriority* is used.

If both the DSCP and VLAN priorities are enabled and present, there is a *SwitchPriorityPrefersDSCP* to break the tie. If 1b, the DSCP priority is preferred over VPRI.

Figure 5-15 shows the mapping.

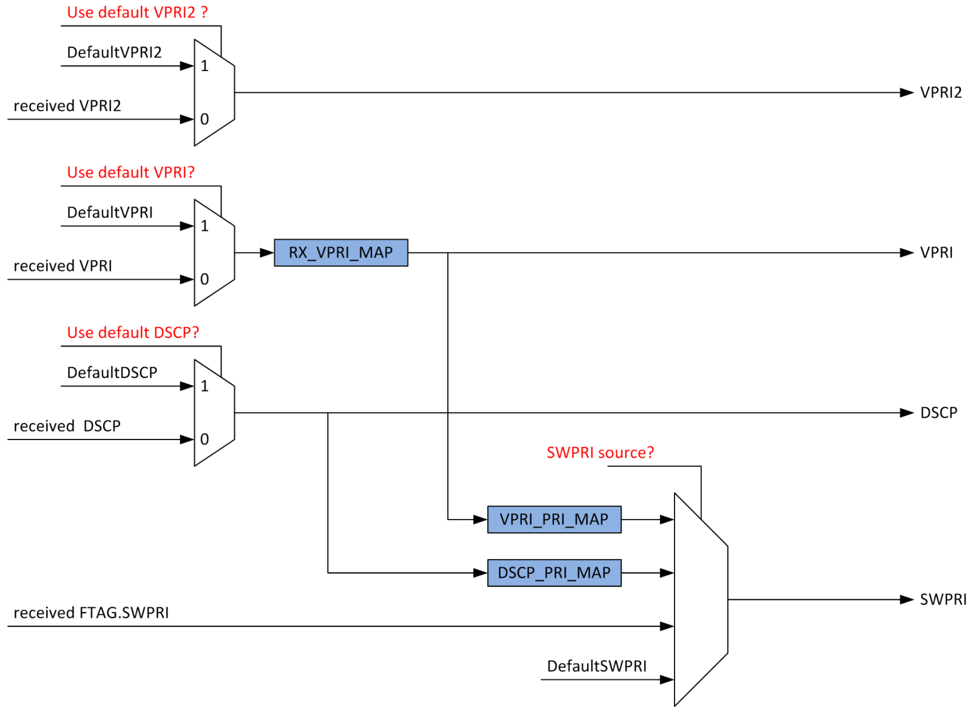


Figure 5-15 Parser Priority Mapping

The expressions are:

- Use Default VPRI2
  — Selected if the VLAN TAG 2 is not present.
- Use Default VPRI1
  — Selected if the VLAN TAG 1 is not present, or PARSER_PORT_CFG_1[p].*useDefaultVLAN* is set to 1b.
- Use Default DSCP
  — Selected if PARSER_PORT_CFG_2[p].*useDefaultDSCP* is set to 1b.

5.7.3 Filtering and Forwarding Unit

5.7.3.1 Overview

The Frame Filtering and Forwarding Unit (FFU) is a generic resource for frame matching and action setting. It contains 2 basic structures; a mapper which derives small keys from larger keys, and 32 CAM/SRAM slices for pattern lookup and action setting. The last slice(s) can optionally be used for egress ACLs that apply in parallel to multiple egress ports.

Each slice has the following elements:

- Configuration to select which frame header fields are selected for the TCAM comparison.
- 1024-entry x 40-bit TCAM block used for key comparison.
- Hit detection circuitry which can cascade across consecutive slices to create keys larger than 40 bits.
- Priority hit detection to determine the highest index hit in each slice.
- 1024-entry x 48-bit SRAM block to store an ingress action(s) associated with the highest priority hit.
- The last slice can optionally feed its final 1024 hits into the egress ACL unit.
- Configuration registers to control the slice behavior for various frame scenarios and key configurations.

The overall unit is shown in Figure 5-16.

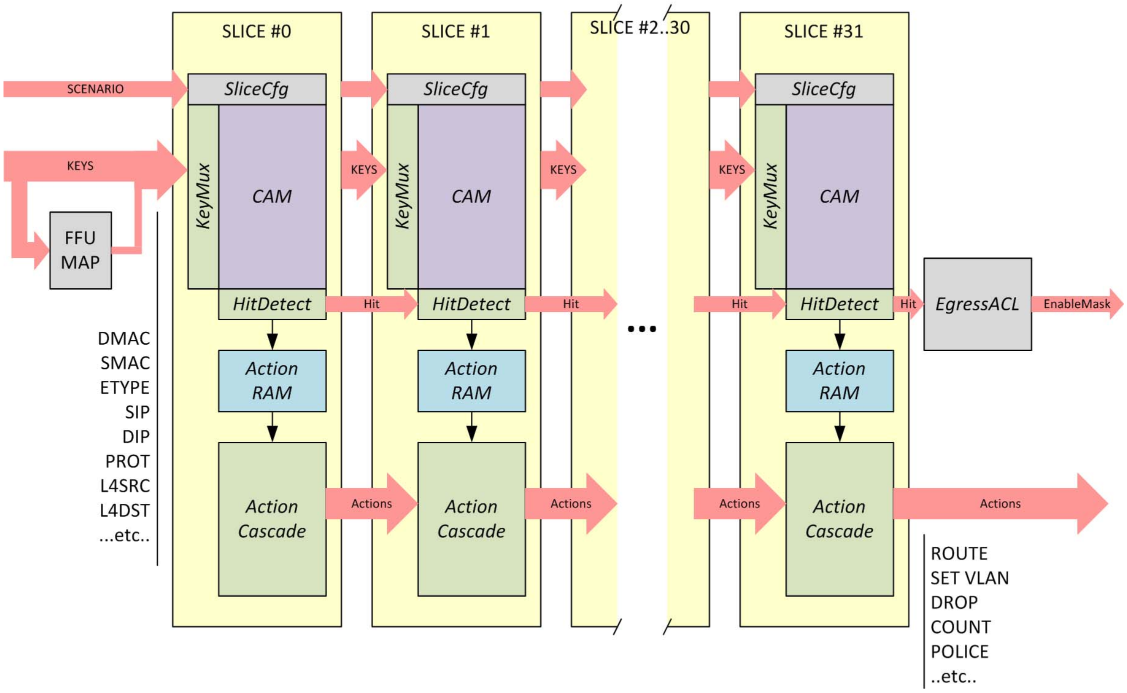


Figure 5-16 Filtering and Forwarding Unit

Each TCAM entry has 80 bits of configuration:

- 40-bit key
- 40-bit key invert

For exact bit positions, see the FFU_SLICE_TCAM table (Section 11.5.4.1).

5.7.3.2 Frame Header Mapper

The Frame Header Mapper is used to create smaller mapped keys from some frame header fields to better utilize the TCAM resources. These mapped keys are available in the FFU slices, along with all the original frame fields. Most of these mapped keys are 4-bits and may be used as bits 32..35 or bits 36..39 of the larger key, leaving 32 bits for another key field such as DIP. The mapper also detects when frames should be routed, based on per-source-port, per-VLAN, and per-DMAC settings. The mapper supports the following functions:

- Maps 8-bit L4 protocol to 4-bit *MAP_PROT* field (see FFU_MAP_PROT[0..7] registers).
  - The mapper recognizes up to 8 L4 protocols. If the protocol received matches one of these protocols, the 3-bit *MAP_PROT* field associated is retrieved, or 0 if the protocol does not match. The 4th bit of *MAP_PROT* is 1b if the frame is a non-head IP fragment for IPv4 or if the IPv6 options were too long (which means that the L4 header information is not valid, since a layer 4 header could not be found).
- Maps 16-bit Ethernet type field into 4-bit *MAP_TYPE* field (see FFU_MAP_TYPE[0..15] registers).
  - The mapper recognizes up to 16 Ethernet types. If the Ethernet type received matches one of the known ones, a 4-bit field associated with the protocol is retrieved. If there is no match, this 4-bit field is set to 0b. Useful for detecting non-IP Ethernet frames, such as ARP. The scenario distinguishes IPv4 and IPv6 from other L3 protocols. Therefore, *MAP_TYPE* is not needed for that purpose.
- Maps 48-bit DMAC and SMAC addresses into a 4-bit *MAP_DMAC* and *MAP_SMAC* fields (see FFU_MAP_MAC[0..31] registers).
  - The mapper recognizes up to 16 Ethernet MAC addresses. If the frame's DMAC address matches one of the known ones (and its *validDMAC* bit is true), a 4-bit MAP_DMAC field associated with the MAC address is retrieved and also sets a router bit indicating if this DMAC address is an address of one of the virtual routers. If there is no match, this 4-bit field is set to 0b. Likewise, the source MAC address is mapped to *MAP_SMAC*, but without the router bit. Even if you do not intend to use the *MAP_DMAC* or *MAP_SMAC* as TCAM keys, at least one MAC entry must be specified with its "router" bit true to do IP routing. The entry can be configured to ignore a specified number of LSB bits when comparing MAC addresses, which is used for VRRP and can also be used to detect the reserved IEEE multicast ranges.
- Maps physical source port into a 4-bit *MAP_SRC* field (see FFU_MAP_SRC[0..47] registers).
  - The mapper retrieves a 4-bit field associated with each source port, along with a routable bit indicating if this port is routable. This mapping is very useful for applying ACL's to a set of equivalent ports without duplicating the TCAM entries.
- Maps 16-bit L4 destination and source ports into a 16-bit *MAP_L4DST* and *MAP_L4SRC* fields (see FFU_MAP_L4_DST[0..63] and FFU_MAP_L4_SRC[0..63] registers).
  - The mapper distinguishes up to 63 contiguous ranges in the 16-bit port address space. It compares the L4 source port to all port boundaries stored in the FFU_MAP_L4_DST registers and retrieves a new 16-bit *MAP_L4DST* from the register if the frame's port is greater or equal to the port stored in the current register, and less than the port stored in the next register (or end of list for this protocol or end of table) if and only if the protocol matches the 3-bit protocol

retrieved from the FFU_MAP_PROT for the current entry, and if the frame is a head fragment, and the entry is marked valid (next entry validity not checked). If there is more than one match, the last match is used. If there is no match, the *MAP_L4DST* is set to the original L4_DST (not 0). This usually enables the TCAM to only look at *MAP_L4DST* and never need the raw L4_DST, thus making exact-match port comparisons consume no resources. The L4 source port is mapped in a similar way except using independent configuration registers. Pseudo code is:

```
// For L4DST (L4SRC similar using L4_DST registers)
 MAP_L4_DST = L4_DST;
 for (i=0;i<64;i++)
     if ( L4_DST >= MAP_L4_DST[i].L4DST &&
          MAP_PROT == MAP_L4_DST[i].MAP_PROT &&
          ( i == 63 ||
            L4_DST < MAP_L4_DST[i+1].L4DST ||
            MAP_PROT != MAP_L4_DST[i+1].MAP_PROT )
          MAP_L4_DST[i].VALID && // MAP_L4_DST[i+1].VALID is not checked
          head-fragment )
     {
          MAP_L4_DST = MAP_L4_DST [i].MAP_L4DST
     }
```

- Maps SIP and DIP addresses into a 4-bit *MAP_SIP* and *MAP_DIP* fields (see FFU_MAP_IP_LO[0..15] and FFU_MAP_IP_HI[0..15] registers).
  — The IP address prefix is compared with 16 known IP address prefixes. If the frame's IP address prefix matches one of the known ones (and the appropriate *validSIP* or *validDIP* bits is true), a 4-bit field associated with this IP address prefix is retrieved. If there is no match, this 4-bit field is set to 0b. Each register contains a value and the number of least significant bits to ignore. This feature is especially useful when most IPv6 routing or ACL rules use only a few common prefixes. A 32-bit IPv4 address is padded with 0's before being compared to the 128-bit value in the registers.
- Maps VLANs into a 12-bit *MAP_VLAN* key (see FFU_MAP_VLAN[0..4095] table).
  — Maps all 4096 VLANs into an arbitrary set of VLAN groups and indicates if each VLAN is routable or not. Rules which treat several VLANs the same can compress those VLANs into the same VLAN group, thus avoiding TCAM expansion.
- Maps packet length into a 4-bit *MAP_LENGTH* field (see FFU_MAP_LENGTH[0..15] registers).
  — The mapper recognizes up to 16 packet length ranges (from IP packet length field, not the Ethernet frame length). It compares the current IP packet length received to all packet lengths stored in the FFU_MAP_LENGTH registers and retrieves a 4-bit key stored in each register if the length received is greater or equal to the length stored in the current register, and for all but the last register, less than the length stored in the next register. Can be used to keep statistics on user-defined length bins, or to drop long packets based on complicated rules, or even to route differently based on length. Typically, entries should be sorted in increasing length order.

In all cases, if there is more than one match within one MAP, the highest entry wins.

5.7.3.3 Slice Activation and Case Configuration

A single slice can be used to hold different type of keys depending of the type of packet received, the parser and mapper construct a 5-bit scenario field:

Table 5-8 Slice

Bit(s)	Description	
1:0	Frame format: 00b = Not an IP frame 01b = IPv4 10b = IPv6 11b = IPv6 with an IPv4-in-IPv6 DIP. Can optionally use IPv4 routing rules for this frame.	
4:2	Frame handling:	
	000b = Switched	This frame should not be routed, and has a destination GLORT of 0 (i.e. no ISL tag). Ingress ACL's may be applied.
	001b = Switched-GLORT	This frame should not be routed, and destination GLORT has been set != 0 in the ISL tag by an ingress chip in a multi-chip fabric. Ingress ACL's may be skipped.
	010b = Reserved.	
	011b = Special	FTYPE is “special”. This usually means mirror traffic or CPU directed control frames, but ACL's may be used.
	100b = Routable	This frame is a unicast IP frame. The ingress port and ingress VLAN are marked as routable, the DMAC is routable and the dglort is zero. This usually means ingress ACL's and routing rules should be applied.
	101b = Routed-GLORT	This frame is a unicast IP frame. The ingress port and ingress VLAN are marked as routable, the DMAC is routable and the DGLORT is non-zero. Generally, ingress ACL's would no longer be applied. However, unicast routing rules must still be applied so that the egress port can make routing modifications to the VLAN/DMAC/TTL of the frame.
	110b = Routable-multicast	This frame is IP multicast, the ingress port and ingress VLAN are marked as routable, the DMAC is either routable or a multicast or broadcast DMAC, and the destination GLORT is zero. Ingress ACL's and multicast IP routing rules are applied.
	111b = Routed-multicast-GLORT	This frame is IP multicast. The ingress port and ingress VLAN are marked as routable, the DMAC is either routable or a multicast or broadcast DMAC, and the destination GLORT is non-zero. Ingress ACL's are generally not applied. However, multicast routing rules must still be applied so that the egress port can make routing modifications the VLAN/DMAC/TTL of the frame.

This 5-bit scenario is then used to configure behavior of each slice through the FFU_SLICE_CFG register, which is indexed by the slice number and scenario. The register set defines the key multiplexing for each slice, as well as specifying whether this slice starts a hit and/or action cascade.

The key muxing is shown in Figure 5-17.

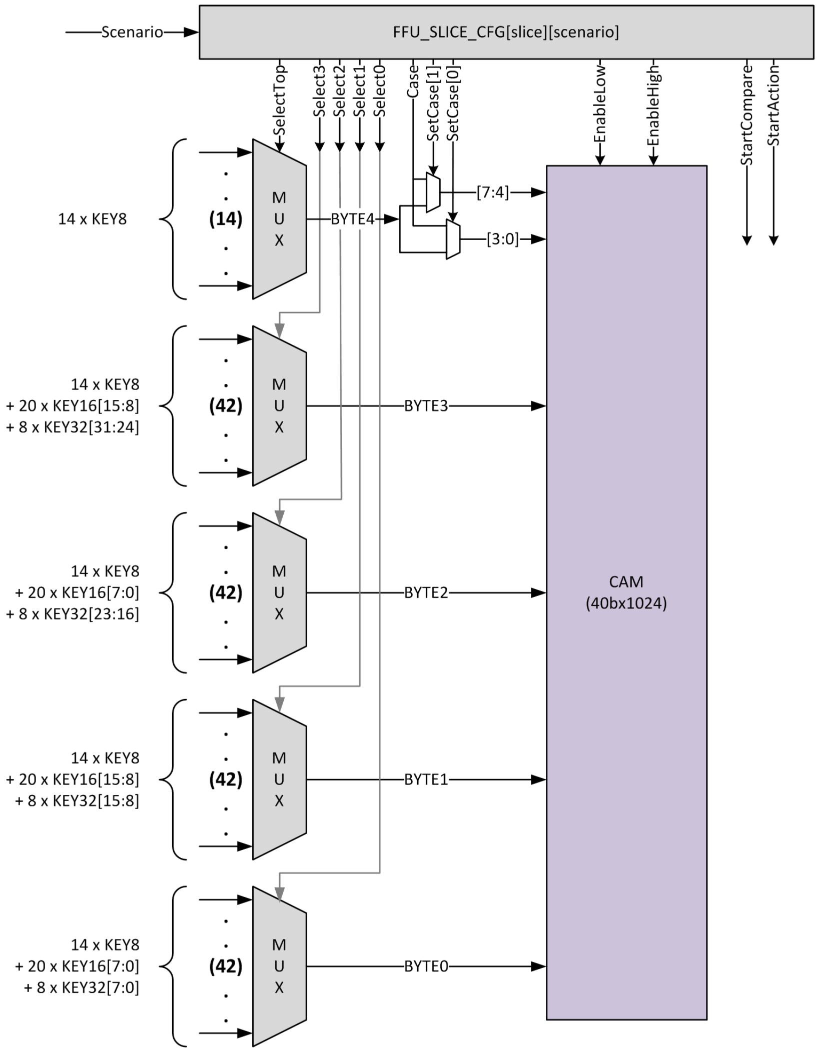


Figure 5-17 FFU Key Muxing

If a slice is used to store more than one type of key depending of type of packet received (by example IPv4 unicast or IPv4 multicast), the “case” is used to qualify each rule. The “case” is a 4-bit value assigned by software per scenario and slice which could be used as a key on either the 4 top bits or next 4 bits of the FFU key search.

For power saving purpose, the slices are divided into two sets of 512 rules. Each set can be individually enabled per slice and per scenario. If a set is not enabled, all the rules loaded in that slice are ignored in the slice and considered non-match.

5.7.3.4 Search Key Selection

The FFU supports the following keys:

- 8 x 32-bit keys (KEY32) — Aligned to the lower 32-bit of the TCAM entry.
- 20 x 16-bit keys (KEY16) — Duplicated on lower 16 bits of the TCAM entry (bits 15:0) and the next 16 bits of the TCAM entry (bits 31:16).
- 14 x 8-bit keys (KEY8) — Replicated on all bytes of the TCAM entry. The top 8 bits can only received the 8-bit and have an extra pair of 4-bit multiplexers to insert the frame case.

The list of keys are defined in the F*FU_MuxSelect* enumeration (Section 11.5.2) and listed in Table 5-9.

Table 5-9 FFU Keys

Key	Content
KEY32 (32-bit keys)	Select_L3_SIP (128-bits split into four 32-bit words) Select_L3_DIP (128-bits split into four 32-bit words)
KEY16 (16-bit keys)	Select_L2_DMAC (48-bits split into three 16-bit halfwords) Select_L2_SMAC (48-bits split into three 16-bit halfwords) Select_DGLORT Select_SGLORT Select_L2_TYPE Select_VPRI_VID set Bits[11:0] = VID Bits[15:12] = VPRI Select_VPRI2_VID2 set Bits[11:0] = VID2 Bits[15:12] = VPRI2 Select_MAP_VPRI1_VID1 set Bits[11:0] = Mapped VLAN (from FFU_MAP_VLAN[vid] table) Bits[15:12] = VPRI (a.k.a. VPRI1) Select_L4_SRC (TCP/UDP ports) Select_L4_DST (TCP/UDP ports) Select_MAP_L4_SRC Select_MAP_L4_DST Select_L4{A..D} (Deep inspection field, 64-bits split into four 16-bit halfwords)

Table 5-9 FFU Keys [Continued]

Key	Content
KEY8 (8-bit keys)	Select_MAP_DIP_MAP_SIP Bits[3:0] = MAP_SIP Bits[7:4] = MAP_DIP Select_MAP_DMAC_MAP_SMAC Bits[3:0] = MAP_SMAC Bits[7:4] = MAP_DMAC Select_MAP_PROT_MAP_LENGTH Bits[3:0] = MAP_LENGTH Bits[7:4] = MAP_PROT Select_MAP_SRC_MAP_TYPE Bits[3:0] = MAP_TYPE Bits[7:4] = MAP_SRC Select_USER Select_FTYPE_SWPRI Bits[3:0] = SWPRI Bits[7:4] = FTYPE Select_TOS Select_PROT Select_TTL Select_SRC_PORT Select_VID_7_0 Select_VPRI_VID_11_8 Bits[3:0] = VID[11:8] Bits[7:4] = VPRI Select_IPMISC (Miscellaneous flags describing IP packets) Bit[0] = TrapIPoptions Set if certain types of IP options are seen, as determined by the PARSER_PORT_CFG_2[port].FlagIP* configuration. Bit[1] = DoNotFrag Set for IPv4 when the DF (do not fragment) flag is set in the IP header. Set for IPv6 unconditionally. Bit[2] = HeadFrag Set on the first or only fragment of an IP packet. (i.e., when the packet is unfragmented or the fragment offset is zero). Bit[3] = MoreFrag Set on an IP packet fragment that is not the last fragment of the packet. (i.e., when the MF flag (in the IPv4 header) or the M flag (in the IPv6 fragment header) are set). Bit[4] = Routable Bit[5] = 0 Bit[6] = 0 Bit[7] = 0 Select_RXTAG (Flags indicating the format of the RX packet) Bit[0] = FTAG present Bit[1] = VLAN1 tag present Bit[2] = VLAN2 tag present Bit[3] = VLAN2 tag before VLAN1 tag Bit[4] = IPV4 header present Bit[5] = IPV6 header present Bit[6] = MPLS present Bit[7] = Custom tag present

5.7.3.5 Slice Cascading

Slices can be cascaded together to produce larger search line and/or larger set of actions. An example of cascading is shown in Figure 5-18.

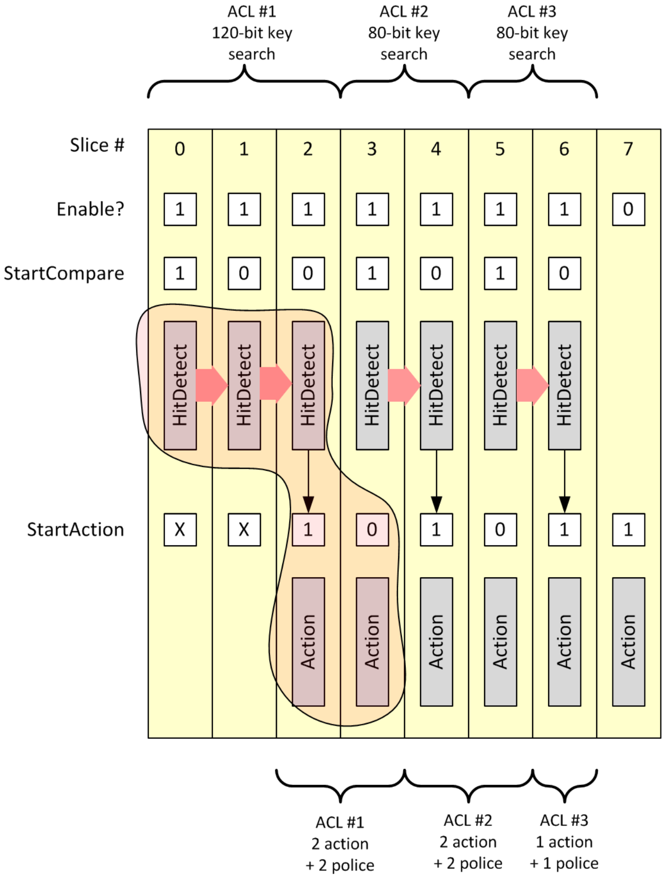


Figure 5-18 FFU Slice Cascading

This example shows three CAM cascades and two ACTION cascade. The three CAM cascades are using slices 0-1-2, slices 3-4 and slices 5-6 respectively. The ACTION cascade are using slices 2-3 and slices 4-5 respectively.

When two or more CAMs are cascaded together, the 'hit' line is the highest line that produces a match across all CAM entries together.

The cascading of CAMs and ACTIONS tables are configurable per slice and per scenario in the FFU_SLICE_CFG register. The FFU_CASCADE_ACTION registers control the cascading option for each slice. Therefore, it is possible to have different cascade arrangements for each scenario.

If there is more than one match, the highest entry wins.

5.7.3.6 Ingress Action

The Filtering and Forwarding Unit produces several different “action fields” which are subsequently used to modify the frame and/or determine its destination. When a TCAM entry “hits” in the TCAM, its associated action entry, which is stored in the FFU_SLICE_SRAM register, is evaluated to determine how the “action fields” should be modified.

Since multiple TCAM entries may “hit” (up to one per slice), there may be multiple action entries which attempt to modify the “action fields” for a given frame. As long as each action entry attempts to modify different action fields, there is no conflict. However, if one action field is modified by more than one action entry that has hit, the conflict must be resolved.

Each action entry contains a 3-bit *Precedence* field which is used to resolve such conflicts. If more than one action entry attempts to modify the same action field, the action field is assigned the value from the action entry that has the highest precedence. If more than one action entry for a given action field have equally high precedence, the tie is resolved in favor of the action entry which resides in the higher-numbered slice.

The list of orthogonal action fields are as follows:

- ROUTE (One action only, 2 types/encodings):
  - ROUTE_ARP (arp_index: 16, arp_count: 4)
  - ROUTE_GLORT (DGLORT: 16, flood: 1)
- SET_FLAGS (mask: 8, value: 8)
- SET_VLAN:12 (also set TXTAG)
- SET_VPRI: 4
- SET_PRI: 4
- SET_DSCP: 6
- SET_USER (mask: 8, value: 8)
- SET_TRIG (mask: 8, value:8)
- COUNT_0: 12
- COUNT_1: 12
- COUNT_2: 12
- COUNT_3: 12

The SET_FLAGS/SET_USER/SET_TRIG provide mask/value pairs. Each bit of these actions has its own 3-bit precedence, and each bit makes its own decision about when to replace the bit with a new value. That allows multiple slices to set different bits in the USER/TRIG/FLAGS fields, or to pass some bits through unmodified.

FLAGS includes six single bit actions which can be set or cleared independently.

Table 5-10 Ingress Action Flags

Flag	Action	Default	Description
FLAG[0]	DROP	0	If bit is 1b after cascade actions, the frame is dropped unless the trigger stage reverts the action.
FLAG[1]	TRAP	0	If bit is 1b after cascade actions, the frame is trapped unless the trigger stage reverts the action.
FLAG[2]	LOG	0	If bit is 1b after cascade actions, the frame is logged unless the trigger stage reverts the action.
FLAG[3]	NOROUTE	0	If bit is 1b after cascade actions, L3 routing action is canceled. If 0b, route action gets applied normally.
FLAG[4]	RX_MIRROR	0	If bit is 1b after cascade actions, the frame is mirrored unless the trigger stage reverts the action.
FLAG[5]	CAPTURE_TIME	USER[1]	If bit is 1b after cascade actions, the time is captured as the frame is transmitted on an Ethernet port. ***Note:*** This bit is set by default to USER[1], which reflects the request for capturing time from a host attached to PCI-Express interface. All USER bits are 0b for frames received from Ethernet ports.

The ROUTE action is used to define the next hop. There are two types of ROUTE action; ROUTE-GLORT and ROUTE-ARP. The ROUTE-ARP is used for ARP lookup, while the ROUTE-GLORT is used for special layer 2 switching when the destination cannot be determined using a normal layer 2 lookup. The ROUTE-ARP data includes the arp_index which defines the base ARP entry (only lower 14 bits used), and the arp_count which specify the number of ECMP arp entries to hash over. The ROUTE-GLORT data includes the new destination GLORT and a flood bit. The FFU produces only one ROUTE action. The FM10000 does not support the ROUTE-ARP action on frames that include MPLS or CUSTOM tags; the presence of these tags may be checked for in the RXTAG key if necessary

The SET_TRIG sets a tag that is passed to the Trigger unit.

The SET_VLAN replaces the ingress VLAN and set the TXTAG option. The TXTAG option is designed to override the normal port tagging option (for details, refer to Section 5.8, “Egress Modifications”) The FM10000 does not support the ROUTE-ARP action when TXTAG is nonzero.

The COUNT's represent indices into 4 parallel banks of policers/counters. A counter index of 0 is interpreted as “do not count”. Refer to Section 5.7.6 for details on counters usage.

The actions are encoded in a 40-bit SRAM as shown in Figure 5-19.

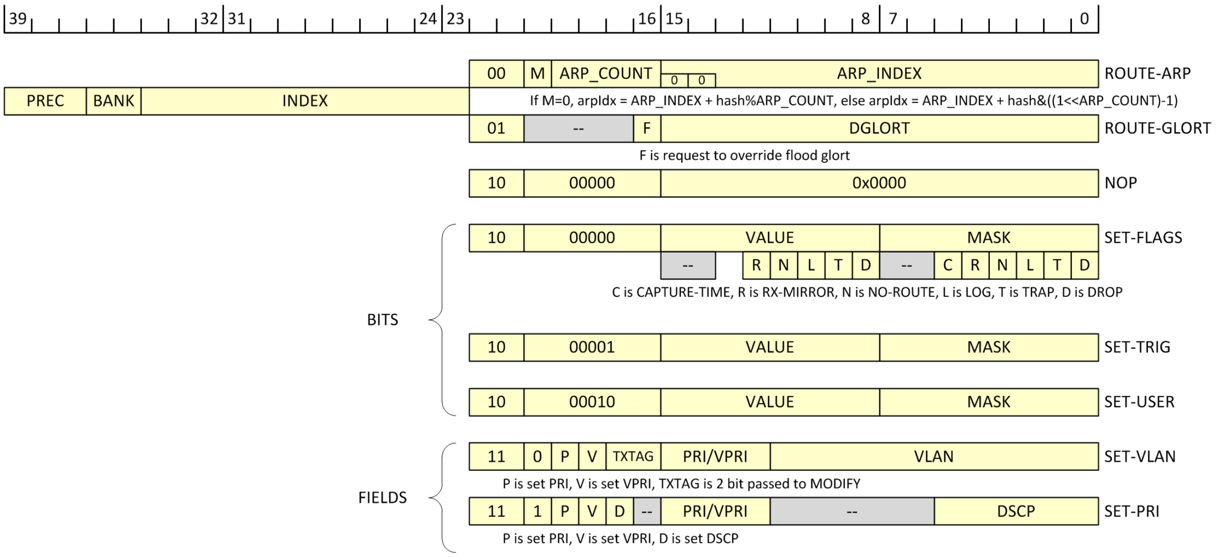

39
32
31
24
23
16
15
8
7
0
00
M
ARP_COUNT
0
0
ARP_INDEX
ROUTE-ARP
PREC
BANK
INDEX
If M=0, arpIdx = ARP_INDEX + hash%ARP_COUNT, else arpIdx = ARP_INDEX + hash&((1<<ARP_COUNT)-1)
01
--
F
DGLORT
ROUTE-GLORT
F is request to override flood glort
10
00000
0x0000
NOP
10
00000
VALUE
MASK
SET-FLAGS
--
R
N
L
T
D
--
C
R
N
L
T
D
C is CAPTURE-TIME, R is RX-MIRROR, N is NO-ROUTE, L is LOG, T is TRAP, D is DROP
BITS
10
00001
VALUE
MASK
SET-TRIG
10
00010
VALUE
MASK
SET-USER
11
0
P
V
TXTAG
PRI/VPRI
VLAN
SET-VLAN
P is set PRI, V is set VPRI, TXTAG is 2 bit passed to MODIFY
FIELDS
11
1
P
V
D
--
PRI/VPRI
--
DSCP
SET-PRI
P is set PRI, V is set VPRI, D is set DSCP

Figure 5-19 Action Encoding

Note: The SET-VLAN command includes provision to also allow setting the VPRI or PRI or both at the same time. Similarly, the SET-PRI command allows to set any or all of the priority fields (DSCP, PRI, VPRI).

5.7.3.7 Egress ACLs

The egress ACL unit follows the last slice. It takes the cascade hit from the last slice, and splits the 1024 hits into 32 groups ("chunks") of 32 entries each (FFU Rules 0..31 refer to Chunk 0 up to FFU Rules 992..1023 refer to Chunk 31). Each group can produce a single precedence hit (which is the index with the highest numbered hit within the group). The egress ACL unit can thus produce 32 actions simultaneously, as opposed to the slices which can produce only one ingress action per slice. Each entry of each group has a single drop action associated with it:

0b = Permit
1b = Drop

This action is defined in FFU_EGRESS_CHUNK_ACTIONS.

Chunks can be cascaded together to form larger control lists. The boundaries of each list are defined per chunk using the FFU_EGRESS_CHUNK_CFG[chunk].*StartCascade* fields. Setting *StartCascade* to 1b for all chunks produces 32 independent lists of 32 entries each, while setting *StartCascade* to 0b on all chunks produces a single 1024-entry list. The highest indexed rule of any given list wins. The pseudo code below shows the order in which the rules are processed.

```
#### In FFU
dropChunk[0..31] = 0;
c=-1;
foreach chunk (31..0)
    if ( FFU_EGRESS_CHUNK_CFG[chunk].StartCascade == 1 ) c=-1;
    foreach rule (31..0)
        if ( hit[chunk*32+rule] == 1 && c < 0 )
            c = chunk;
            dropChunk[c] = FFU_EGRESS_CHUNK_ACTIONS[c*32+rule].Drop;
            break;
```

The example below shows 5 lists of various sizes.

```
Chunk StartCascade
  31        x     \   (highest precedence chunk of list #1)
  30        0      |
  29        0      |
  28        0       > List #1 : 288 entries (9x32)
  27        0      |
  26        0      |
  25        0      |
  24        0      |
  23        0     /   (lowest precedence chunk of list #1)

  22        1     \   (highest precedence chunk of list #2)
  21        0      |
  20        0      |
  19        0      |
  18        0      |
  17        0       > List #2 : 352 entries (11x32)
  16        0      |
  15        0      |
  14        0      |
  13        0      |
  12        0     /   (lowest precedence chunk of list #2)

  11        1     \   (highest precedence chunk of list #3)
  10        0      |
   9        0       > List #3: 192 entries (6x32)
   8        0      |
   7        0      |
   6        0     /   (lowest precedence chunk of list #3)

   5        1     \   (highest precedence chunk of list #4)
   4        0      |
   3        0       > List #4: 160 entries (5x32)
   2        0      |
   1        0      /

   0        1   --> List #5: 32 entries
```

The FFU_EGRESS_PORT_CFG[port] defines for each port which egress chunk drop action is applicable to that port. If a port has no egress chunk drop enabled, then a packet going to that port is never dropped due to an egress ACL rules.

```
#### In DMASK generation
# Compute Egress ACL DMASK
eaclDmask[0..47] = 1
foreach port (0..47)
    if ( (dropChunk & FFU_EGRESS_PORT_CFG[port].DropApply) != 0 )
        FFU_EGRESS_DROP_COUNT[port]++
        eaclDmask[port] = 0;

### Apply Egress ACL DMASK
dmask &= eaclDmask
```

5.7.3.8 Router Source MAC

When routing, a 4-bit *RouterId* is used by MODIFY to look up which 48-bit router MAC is used for the routing SMAC update operation. At the end of the Frame Lookup stage, the *RouterId* is defined as:

```
if (routable && unicast)
  then RouterId = MAP_DMAC // The DMAC used to identify the frame as routable
else RouterId = 0
```

This supports the mode where a unicast route can use the MAC used to identify this router as a destination as a source on transmit. If the frame is multicast, the default router MAC is used. This *RouterId* may later be overridden by specifying a *RouterId* in a NextHop route, to have a per-route router SMAC.

5.7.4 Frame Hashing

5.7.4.1 Overview

For purposes of ECMP and Link Aggregation, two hash values are calculated from the header fields of each frame:

- **Layer 3/4 Hash** (36 bits) — For ECMP.
- **Layer 2/3/4 Hash** (48 bits) — For local and distributed link aggregation (filtering and pruning, respectively).

The keys to these hash functions are constructed in a configurable manner to provide the following features:

- Symmetry — Hash value remains the same when source and destination fields are swapped.
- Static field dependence — Support for including a specific set of header fields in the hash function.
- Dynamic field dependence, based on frame type — Certain fields can be omitted or included when a frame is IPv4/IPv6.

In a multi-chip switch, it may be desirable to load balance frames over different ECMP sets or LAGs in a statistically independent fashion. For example, the same hash function should not be used to distribute traffic over the second-layer links of a 3-tier fat tree as the first-layer links. Thus multiple independent frame hashes are calculated for a given frame, with configuration settings determining which of these hashes applies to a given frame and device. Any given device can apply a maximum of three independent hash values to a given frame: one of three independent choices for ECMP, and two of four independent choices for link aggregation.

Binning of the selected hash value is performed using division for ECMP up to 16 ways, and modulo for link aggregation and higher-radix ECMP. Division (also known as Hash Threshold) has the advantage of providing better stability of the bin mappings when the number of bins is changed (cf RFC 2992). This property is only important for ECMP. Therefore, in the interest of maintaining backwards compatibility, link aggregation continues to employ modulo binning. Both functions provide equally balanced hash binning.

In a multi-chip switch, it may be desirable to load balance frames over different LAGs in a statistically independent fashion. For example, the same hash function should not be used to distribute traffic over the second-layer links of a 3-tier fat tree as the first-layer links. Thus multiple independent frame hashes are calculated for a given frame, with configuration settings determining which of these hashes applies to a given frame and device. Any given device can apply two of four independent choices for link aggregation and binning of the selected hash value is performed using modulo.

Table 5-11 Binning Functions

Type	Equation
Division binning (ECMP, bin_count=1..16)	index = base + (hash * bin_count) / 4096
Modulo binning (ECMP, bin_count=2^n up to 4096)	index = base + hash % bin_count
Modulo binning (Link Aggregation)	index = base + hash % bin_count

Figure 5-20 illustrates the frame hashing data flow in the Frame Processing Pipeline.

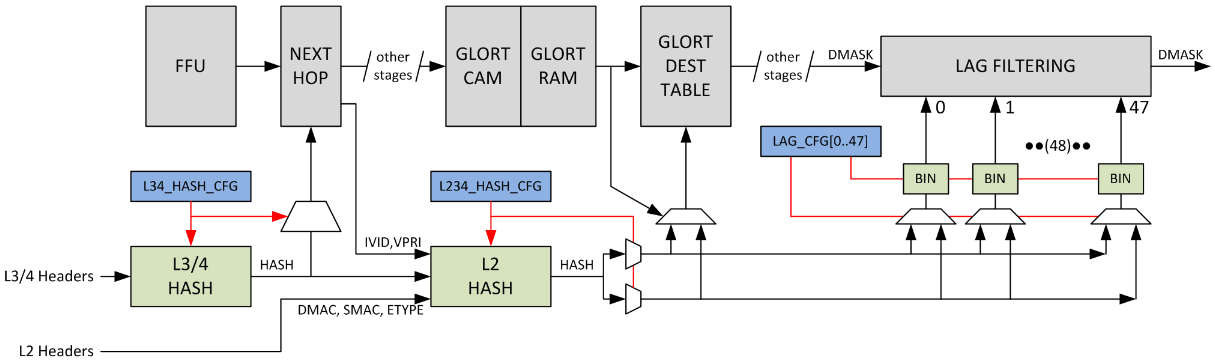


Figure 5-20 Frame Hashing

Note: Although the L2/3/4 hash produces four 12-bit rotations, only two are available for use on a given chip. This restriction limits the amount of parallel binning circuitry in the implementation. In the register definitions, these two globally-selected rotations are referred to as Rotations A and B.

5.7.4.2 Layer 3/4 Hash

The 36-bit layer 3/4 hash value is calculated using two 32-bit CRC functions and the going through a 12-bit permutation table:

```
// CRC of 42 bytes if IP frame.
H34 = 0;
if (frameIsIP)
   {
      H34[31:0] = CRC32(0xEDB88320, Bytes[0..41]);
      H34[35:32] = CRC32(0x82F63B78, Bytes[0..41])[3:0];
   }

// permutation table
H34 = H34 >> (L34_HASH_CFG.ecmp_rotation*12);
H34 &= 0x0fff; // 12-bit ptable index
P34 = ptable[H34];

// binning function
if (arp_entry.Arp_Type)
   {
      // bin_count = 1..4096
      bin_count = 1 << arp_entry.Arp_Count;
      index = base + {H34[11:4], P34[3:0]} % bin_count;
   }
else
   {
      // bin_count = 1..16
      bin_count = arp_entry.Arp_Count;
      if (bin_count == 0)
         bin_count = 16;
      index = base + ({P34[3:0], H34[7:0]} * bin_count) / 4096;
   }
```

In this notation, the first parameter of the CRC32 function specifies the CRC polynomial; the second parameter specifies the byte sequence input key to the CRC. The 0xEDB88320 polynomial corresponds to the standard 802.3 Ethernet CRC32.

The input byte sequence is assigned from up to 42 bytes of the frame header:

Bytes 15:0	Bytes 31:16	Bytes 36:32	Byte 37	Bytes 39:38	Bytes 41:40
SIP	DIP	L3FLOW	L4PROT	L4SRC	L4DST

The particular values used in the SIP, DIP, L3FLOW, L4PROT, L4SRC, and L4DST fields are determined based on the L34_HASH_CFG configuration and the contents of the packet header, as illustrated in the following tables.

Table 5-12 Layer 3/4 Hashing SIP Field

	UseSIP	Symmetric	Bytes 11:0	Bytes 15:12
IPv4	1	0	0	SIP[31:0]
	1	1	0	SymA(SIP[31:0], DIP[31:0]*)
IPv6	1	0	SIP[127:32]	SIP[31:0]
	1	1	SymA(SIP[127:32], DIP[127:32]*)	SymA(SIP[31:0], DIP[31:0]*)
IPV4/IPv6	0	x	0	

* DIP field is zero if *UseDIP* == 0.

Note:

Table 5-13 Layer 3/4 Hashing DIP Field

	UseDIP	Symmetric	Bytes 27:16	Bytes 31:28
IPv4	1	0	0	DIP[31:0]
	1	1	0	SymB(SIP[31:0]*, DIP[31:0])
IPv6	1	0	DIP[127:32]	DIP[31:0]
	1	1	SymB(SIP[127:32]*, DIP[127:32])	SymB(SIP[31:0]*, DIP[31:0])
IPV4/IPv6	0	x	0	

* SIP field is zero if *UseSIP* == 0.

The function Sym(x,y) is ordering individual bytes of different fields so that Sym(x,y) and Sym(y,x) are resulting to same result:

```
SimA(SIP,DIP):
     byte0 = MAX(DIP[7:0],SIP[7:0]);
     byte1 = MAX(DIP[15:8],SIP[15:8]);
     ...
     byte15 = MAX(DIP[127:120],SIP[127:120]);

SimB(SIP,DIP):
     byte16 = MIN(DIP[7:0],SIP[7:0]);
     byte17 = MIN(DIP[15:8],SIP[15:8]);
     ...
     byte31 = MIN(DIP[127:120],SIP[127:120]);
```

Table 5-14 Layer 3/4 Hashing Flow Field

	Byte 32	Bytes 35:33[19:0]	Byte 35[7:4]	Byte 36
IPv4	DiffServ[5:0] & DiffServMask	0	0	ISL_USER & UserMask
IPv6	TrafficClass[5:0] & DiffServMask	FlowLabel & FlowLabelMask	0	ISL_USER & UserMask

The flow identification (Flow Label) fields are each individually bit-masked to provide fine control over which bits are included in the hash value. The masks are configured in L34_FLOW_HASH_CFG. The ISL_USER field comes from the ISL tag, or from the default specified in PORT_CFG_ISL if the frame is not F64-tagged.

Table 5-15 Layer 3/4 Hashing Protocol Field

	UsePROT	Symmetric	Byte 37
IPv4	1	x	Protocol
IPv6	1	x	NextHeader
IPv4/IPv6	0	x	0

Table 5-16 Layer 3/4 Hashing Layer 4 Fields

	UseL4{SRC,DST}	Symmetric	Byes {39:38, 41:40}
UseTCP==1 and Protocol is TCP UseUDP==1 and Protocol is UDP UsePROT1==1 and Protocol is PROT1 UsePROT2==1 and Protocol is PROT2	1	0	{SRC,DST}_PORT
	1	1	{SymA(SRC_PORT, DST_PORT), SymB(SRC_PORT, DST_PORT)}
	0	x	0
Otherwise	x	x	0

5.7.4.3 Layer 2/3/4 Hash

The 48-bit L2/3/4 hash value used for link aggregation pruning and filtering is calculated in a similar manner to the L3/4 hash, using an additional sixteen layer 2 bytes from the frame header:

```
// Include hash from L2 if needed
H234 = 0;
if (L234_HASH_CFG.UseL2ifIP || !frameIsIP)
   {
      H234[31:0] = CRC32(0xEDB88320, Bytes[0..15]);
      H234[48:32] = CRC32(0x82F63B78, Bytes[0..15])[15:0];
   }

// Include hash from L34 if requested
if (L234_HASH_CFG.UseL34 && frameIsIP)
   {
      H234[35: 0] ^= H34[35:0];
      H234[47:36] ^= H34[11:0];
   }

// retrieve L234 permutation selector
if (glort_ram_entry.hash_rotation)
   rot = L234_HASH_CFG.rotationB * 12;
else
   rot = L234_HASH_CFG.rotationA * 12;

// select permutation
H234  = H234 << rot;
H234 &= 0x0fff; // 12-bit ptable index
P234  = ptable[H234];
H234  = {H234[11:4], P234[3:0]}; // insert 4 bits from ptable
```

Four different "rotations" of the 48-bit value are used to derive four distinct hash values from which may be chosen Rotation A and Rotation B (see L234_HASH_CFG register):

- Rotation 0: H234[11:0]
- Rotation 1: H234[23:12]
- Rotation 2: H234[35:24]
- Rotation 3: H234[47:36]

The 48-bit H234 hash value provides four independent hash functions that optionally include L2 or L3/4 header fields, depending on configuration settings and the type of frame.

The *UseL2ifIP* and *UseL34* configuration bits in L234_HASH_CFG provide the following options for dynamic header field dependence:

UseL2ifIP	UseL34	Semantics
1	1	H234 always includes the maximum amount of header information.
1	0	H234 only includes layer 2 header fields.
0	1	H234 only includes layer 3/4 header fields when the frame is IP. If the frame is non-IP, the layer 2 header is used for hashing.
0	0	H234 is always zero for IP frames. Otherwise it uses the layer 2 header. Invalid configuration.

The sixteen layer 2 input bytes to the CRC are defined in the following tables:

Table 5-17 Layer 2 Hashing DMAC/SMAC Fields

Bytes 0-12	Symmetric = 0	Symmetric = 1
0	DMAC[7:0]	MAX(DMAC[7:0],SMAC[7:0])
1	DMAC[15:8]	MAX(DMAC[15:8],SMAC[15:8])
2	DMAC[23:16]	MAX(DMAC[23:16],SMAC[23:16])
3	DMAC[31:24]	MAX(DMAC[31:24],SMAC[31:24])
4	DMAC[39:32]	MAX(DMAC[39:32],SMAC[39:32])
5	DMAC[47:40]	MAX(DMAC[47:40],SMAC[47:40])
6	SMAC[7:0]	MIN(DMAC[7:0],SMAC[7:0])
7	SMAC[15:8]	MIN(DMAC[15:8],SMAC[15:8])
8	SMAC[23:16]	MIN(DMAC[23:16],SMAC[23:16])
9	SMAC[31:24]	MIN(DMAC[31:24],SMAC[31:24])
10	SMAC[39:32]	MIN(DMAC[39:32],SMAC[39:32])
11	SMAC[47:40]	MIN(DMAC[47:40],SMAC[47:40])

Notes: The DMAC field is zero if *UseDMAC* == 0.
The SMAC is be zero if *UseSMAC* == 0.

Table 5-18 Layer 2 Hashing Type Field

	UseTYPE	Bytes 13:12
EtherType < 0x600	1	0
EtherType ≥ 0x600	1	EtherType

The EtherType included in the hash is the layer 2 header's first non-VLAN, non-CUSTOM tag EtherType as output by Parser stage.

Table 5-19 Layer 2 Hashing VID/VPRI Fields

UseVPRI	UseVID	Byte 14[7:4]	Bytes 15:14[11:0]
0	1	0	VLAN ID
1	0	VPRI	0
1	1	VPRI	VLAN ID
0	0	0	0

The VLAN ID used in the hash function is the ingress VID (IVID) as defined in Section 5.7.5. Similarly, the VPRI included in the hash is the final associated value at the output of the FFU.

5.7.4.4 ptable

The ptable returns 4 bits to be inserted into a 12-bit hash key to improve hashing distribution. “p” stands for permutation. {ptable_index[11:4], ptable_out[3:0]} is a permutation of {ptable_index[11:0]}.

The ptable is following fixed function:

```
uint32 ptable(uint32 index)
{
   uint32 block;
   block = raw_table[index / 8];
   block = block >> ((index % 8) * 4));
   return block & 0x0f;
}

// raw table is 4096x4b (packed as 128x32b)
unsigned int raw_table[128] = {
   0x53A0647B, 0x89CF21ED, 0x0A2B5763, 0x89DE41FC,
   0x1D7FB80C, 0x3A265E49, 0xEB7DF938, 0x41A5062C,
   0x60A97D15, 0xEFC2B483, 0x3BD59FE0, 0xA8C71264,
   0xB4731A08, 0x29DFE6C5, 0x1F8C7526, 0x3409DABE,
   0x9DBE134F, 0x278650AC, 0x4517D2FB, 0xAE3C0689,
   0x9538E4D1, 0xC26FA7B0, 0x78D9B23E, 0xFA64C510,
   0x2D8B9E74, 0x35601AFC, 0x20CF7D19, 0x86A34BE5,
   0xD37ECA65, 0x02F9B148, 0x96BF7C58, 0xE1A03D42,
   0xF50862E7, 0x41BC9A3D, 0x0C189DF6, 0xE547A2B3,
   0x09678245, 0xF1BDAE3C, 0x9E105FB3, 0x647CAD82,
   0x8742BD69, 0xC0E3FA15, 0xA7CE9542, 0x0FD1386B,
   0x1CDFA637, 0x092E8B45, 0x36E94F17, 0x5B028DCA,
   0xA4E901F5, 0x3D7B286C, 0xA718B056, 0xEF9CD234,
   0x1785392F, 0xADB0E6C4, 0xE4516F2B, 0x703DC98A,
   0xADE9C526, 0x834FB107, 0xBA729150, 0xE63F84CD,
   0x602A93F5, 0xC47D18BE, 0x582DE9A1, 0x03B6C47F,
   0x3F92ED76, 0x84B5C01A, 0x581CB40F, 0x92A63D7E,
   0xFB719D5E, 0x264A8C03, 0x4526BF1D, 0x8EAC0937,
   0xC6B108FA, 0x9D3247E5, 0xDE032A8B, 0x9C645F17,
   0x53C1407A, 0x8B629DEF, 0xBD87165C, 0xE409A2F3,
   0x39E426D1, 0xA78FB5C0, 0xC83B0F16, 0x2A79E45D,
   0xAF79608E, 0xB3C1452D, 0x364EDB80, 0xF71C25A9,
   0xF3EC8BA0, 0x49251D76, 0x2A45FDC0, 0x37EB6981,
   0x846BCD5F, 0x39A712E0, 0x65A9C412, 0xED708F3B,
   0x6341B52A, 0x98D70EFC, 0x6EB0C5D9, 0x24F3A187,
   0x1C49078E, 0xF5A62B3D, 0x031657AD, 0xFC4B29E8,
   0x19D3AB45, 0x08CE672F, 0xDC0A14E7, 0xBF395826,
   0x8F4A0E62, 0xDB53C179, 0xF0AD27C1, 0xEB895364,
   0x0EB96DCA, 0x3582F417, 0x4A9D52C0, 0x3E1F768B,
   0x3907AFE1, 0x82D6C45B, 0x6B5A8D27, 0x39F0CE41,
   0x4F2509C6, 0x7EBD1A83, 0xB67EF39D, 0x5C41A208,
   0xCA098B52, 0xDE64371F, 0xF3B7D91E, 0x8A6240C5,
   0x7694A531, 0xBE0C2F8D, 0x57183294, 0xCF0BA6DE,
   0xC16BA3E5, 0x4F0D2879, 0xD65C89EB, 0x4320A17F,
   0xDB7F106C, 0xA485329E, 0x732BADC4, 0x85061E9F,
   0x89534D6F, 0x1AB270EC, 0xE63D52F4, 0x0C87AB19,
   0x97D41850, 0xEF326CAB, 0x3127FA64, 0xEDBC5980,
   0xED610ACB, 0x925F8473, 0x06A923D4, 0xBC8F15E7,
   0x571390DE, 0xFC4BA826, 0x5F2B0438, 0x1AD7C6E9,
   0xA4F52716, 0x98D3BCE0, 0x0B5DA61E, 0x7923C48F,
   0x53FBE1DA, 0x6290C487, 0x9C13D867, 0xFBAE4520,
   0x97052641, 0xEDFC8AB3, 0x51DC3849, 0xE6FBA072,
   0xA0E63492, 0xD1B7CF85, 0x93A8BF56, 0xD74E20C1,
   0xA13EC492, 0xD857F0B6, 0xEA7013B9, 0x65F482DC,
   0x2AE6179C, 0xF35804DB, 0xC1A94527, 0xD360FE8B,
   0xA473FE9C, 0x8B20D615, 0x3CE7B26F, 0x40A1D598,
   0xDF9B807A, 0x24361C5E, 0xDA9B3E70, 0x1C456F28,
   0xD37BF289, 0x6EC1A045, 0x149780D5, 0xABF2EC36,
   0x498C6531, 0x7FBDE0A2, 0xEBC38F50, 0x16AD9247,
   0x912F0AB6, 0x5E38DC47, 0xA6F41C5B, 0x08927D3E,
   0x5C37A12B, 0xF96ED804, 0x6A0CFB19, 0xE57D8234,
   0x78C302AB, 0x5FED1694, 0xE762DA4C, 0x95801F3B,
   0x93F165D0, 0x78AB2E4C, 0x3AC9561D, 0x2E8047FB,
   0xF4DE93CB, 0x0652A178, 0xB61EC4F9, 0x083DA257,
   0x0C2516DA, 0x37FEB849, 0x4B8617EC, 0xF023A5D9,
   0x285D06BE, 0x1A34FC97, 0xD52136E4, 0x8C70FB9A,
   0xA982F073, 0xCD5416EB, 0xD1528B04, 0x6CF9A37E,
   0x50B21EC4, 0x63D9F87A, 0x3E8A5492, 0xFB0C61D7,
   0x7E496A2F, 0xC318B0D5, 0x6F2DB43A, 0x915CE807,
   0x76CBDF49, 0x83A0251E, 0x698D12FC, 0x7BA0E453,
   0x5B780396, 0x14CEAFD2, 0x2078B3AD, 0x1C596EF4,
   0x356D14BF, 0xEA980C27, 0x3BEA9D84, 0x5C6F0217,
   0xD8FA4501, 0xBE27639C, 0x9BFE6105, 0x274D8CA3,
   0x31ECD28A, 0x6B7540F9, 0x5E7C1843, 0x69F0DA2B,
   0x920E735A, 0xD1C68B4F, 0x792E8FD1, 0xCB403A65,
   0xC159B30F, 0x4EA786D2, 0x10DBF47E, 0x2896CA35,
   0x7649D102, 0xC5E38BAF, 0x32F74E89, 0xCD61AB50,
   0xA63725E9, 0x0BC4F8D1, 0xED40298B, 0x13ACF765,
   0x2F4D1CA3, 0x956E70B8, 0xC2A91D0B, 0x54783E6F,
   0x1B32FE7C, 0x69D85A40, 0xC406D5A9, 0xB2E3F871,
   0x6D482095, 0xF3EC1B7A, 0x6E3A2F70, 0xD45891BC,
   0x728109D5, 0xC43F6AEB, 0x2EB935A6, 0xC74D0F81,
   0xFB3C6A42, 0x51807DE9, 0x8EDA5943, 0x710C6FB2,
   0x8407ACD6, 0xBF5E1923, 0x0B2831D5, 0xA497C6FE,
   0x8DA9F015, 0x34BC2E76, 0xA386E90C, 0x754D1B2F,
   0x6F5DAC3E, 0x782B9014, 0x0A497B8C, 0xD31F6E52,
   0x10C32AE8, 0xBF65497D, 0xBA5EF20D, 0x73486C91,
   0x932A1B47, 0xC80D65EF, 0xC7BAD304, 0x5821FE96,
   0x4C93F5D7, 0x01BA862E, 0x3CFB0E75, 0x98412AD6,
   0x927C01E3, 0x5A84D6BF, 0x56CEF843, 0xB90D712A,
   0x13475E62, 0x0C8FAD9B, 0xEC90B347, 0xD51FA862,
   0x5F416B70, 0x89C2AED3, 0x9D5416FB, 0x28A370EC,
   0x0B4E18FA, 0x9726D3C5, 0xE9302D7F, 0x5A164BC8,
   0x9BE81D64, 0x57A3CF20, 0x1FE50ACB, 0x6832D794,
   0x9C286F34, 0xD17A0BE5, 0x93EC508A, 0xF1672D4B,
   0xE34B8C9A, 0x7DF56102, 0x5308A761, 0xC9E42FDB,
   0xC9851F73, 0xA602E4BD, 0xC60DEF57, 0xBA492813,
   0x74901EFD, 0xA3852CB6, 0xD60F197C, 0x8E3AB245,
   0xBE7A1902, 0x835CDF64, 0xEA7F21B4, 0x09638C5D,
   0x24B57D6C, 0xEA81930F, 0x4C5D6E19, 0x03B8F2A7,
   0xBD51AF37, 0x826EC094, 0xD290B67E, 0xC8314A5F,
   0x4837EDF9, 0xC2056BA1, 0x56FD4ECB, 0x97A13280,
   0xFA053D1E, 0x8B74962C, 0xF982730D, 0xB5641EAC,
   0xBA637F54, 0x80CD1E92, 0x2EA7D536, 0xBF981C40,
   0x182BADCF, 0x956307E4, 0x130C4B9A, 0xF25D78E6,
   0xF4B69835, 0xA0C12E7D, 0xA403FC95, 0xB8D7261E,
   0x39CE4BF8, 0x6DA71052, 0xCF2D6E03, 0x18A54B79,
   0x7D01CFA6, 0x3BE95842, 0x47BD526F, 0xACE31980,
   0xBD479613, 0xFA0C25E8, 0x7385BA94, 0x1CE260FD,
   0x6EA1C950, 0x42B73FD8, 0x3210B4F5, 0xC796DEA8,
   0x1C98B630, 0xA7F5E2D4, 0xBD4E7A29, 0x3815CF60,
   0xE340A9C8, 0x6D7F521B, 0xF280B491, 0x5DC37AE6,
   0x91BE3705, 0x8AF4CD62, 0xEB298047, 0x1365FACD,
   0xF8DB4E63, 0x1A5C2790, 0xC9E47BA8, 0xF621D305,
   0xA9280F53, 0x7ECDB461, 0xA4E8761F, 0x25B30D9C,
   0x90ADF45C, 0x7163B8E2, 0x38D21E54, 0xF706A9BC,
   0x596D3ABE, 0x1824C70F, 0xA6739FEB, 0x10D8C245,
   0x79CE6A13, 0x2D4B58F0, 0x16D934FA, 0xE570CB82,
   0x4EB26798, 0xFA3150DC, 0xDCF9E861, 0xA307B452,
   0x04BC65A8, 0x2E391F7D, 0x2BA69E15, 0x438C0DF7,
   0xF8A35C7E, 0x09D614B2, 0x5B84290C, 0xDF6317AE,
   0x04D6528A, 0x1B9F37CE, 0x352B17E4, 0x89C6DF0A,
   0x01D48B7C, 0x9E253AF6, 0x8AE3DB49, 0xF201675C,
   0x60859CAF, 0xD3B471E2, 0xF49857ED, 0x321AB06C,
   0xCB154D32, 0x0E78AF69, 0xF2DA638E, 0x15BC9740,
   0xD7301245, 0xC9F6AB8E, 0x86CD317E, 0xFA25B490,
   0x27D05F1A, 0xEB34C896, 0xC3AD2F47, 0x61B580E9,
   0x2E046B7F, 0xD1C3598A, 0xA91B7F20, 0xD438C6E5,
   0xB49CD175, 0x23E08AF6, 0x54A83E16, 0xD027BFC9,
   0x37A14695, 0x82BEDF0C, 0x5D2A897E, 0x34B06CF1,
   0xE1F3B80A, 0x5DC47926, 0x8E62F179, 0xADB4C035,
   0x4C7F3682, 0x09DE15AB, 0xE2BD61A8, 0x4C053F79,
   0x2D180B56, 0x9E3CAF74, 0x438201BC, 0xDA9E5F76,
   0xCBD854F1, 0xEA279306, 0xDF0B695A, 0x481E27C3,
   0xB835E72A, 0x4FC9160D, 0x7BE9586D, 0x204A1F3C,
   0x7D2AF196, 0xE358C0B4, 0x78AB2CDF, 0x69E14530,
   0xB201D493, 0xEC6F578A, 0xEB4326A9, 0xC5D710F8,
   0xF6A97834, 0xB2E51CD0, 0x5DF41E89, 0xBCA03276 };
```

5.7.5 Next Hop

5.7.5.1 ARP Unit

The packet is routed if and only if the FFU returns a route-arp action. The inputs to the ARP units are:

- A source MAC and VLAN ({VID,VPRI}).
  - These are not changed by the ARP unit.
- The route action from the FFU including the following fields:
  - ARP index (16 bits).
  - Path count (4 bits).
  - Path count type (1 bit).
- The route action is executed only if:
  - FFU has issued a route-arp action.
  - FFU has issued an ARP-index different than 0.
  - FFU has not issued a NO_ROUTE action.

The outputs of this unit are:

- Updated DMAC.
- Updated DGLORT.
- Ingress VID (IVID), always set to received VID from FFU.
- Egress VID (EVID).
- VPRI, always set to received VPRI from FFU.
- ROUTE flag.
- Various trapping flags for special frames.

The actions of this unit are:

- No action
  - In this case no routing was triggered in the FFU, meaning that the frame is either forwarded by its original destination MAC or the FFU specifies a destination GLORT to use. The egress VID is set to ingress VID.
- New egress VLAN, destination MAC and RouterId (*TYPE*=MAC, *RouterId* non-zero)
  - If the FFU specified a route action (and the NOROUTE flag was not set). the ARP is looked up according to the ARP index and path count. If the entry's *TYPE*=MAC, it produces a new destination MAC and new egress VID for use in the L2 lookup stage.
  - A *RouterId* of 15 is used to specify that the default Router ID derived from FFU_MAP_MAC should be retained. If *RouterId* is between 1 and 14, this new *RouterID* replaces the one derived from FFU_MAP_MAC.
- New destination GLORT, egress VLAN and MTU-index (*TYPE*=GLORT, *RouterId* equal zero, *markRouted* equal one, *IPv6Entry* equal zero)

- — An updated DGLORT is produced, along with a 3-bit MTU index (for multicast MTU checking). This action is used for IP multicast. The egress VID is updated from the entry, but in most cases is set to zero, Egress VID and STP state for VLAN 0 is set to always permit forwarding. For IP multicast, these checks are encoded in the VLAN replication list.
- — By setting *markRouted* to one, this frame is treated as being routed from the perspective of downstream processing. As with the MAC entry type, setting *RouterIdGlort* to 15 means *RouterId* from FFU_MAP_MAC remains unchanged. Otherwise the SMAC is updated according to the RouterId specified. The MTU and TTL is checked, and the VLAN is changed.

- New egress VID, derived MAC from IPv6 and *RouterId* (TYPE=MAC, RouterId equal zero, *markRouted* equal don't care, *IPv6Entry* equal one)
  - — This acts just like *TYPE*=MAC, except that the new destination MAC is derived from the destination IPv6 address, supporting stateless auto-configured MAC addresses. As for entry MAC, setting *RouterIdGlort* to 15 means *RouterId* from FFU_MAP_MAC remains unchanged, otherwise the SMAC is updated according to the *RouterId* specified.
- New DGLORT (*TYPE*=GLORT, *RouterId* equal zero, *markRouted* equal zero, *IPv6Entry* equal zero)
  - — This does not mark the frame for routing, instead the only field that is updated is the destination GLORT. The MTU index and VID are not used. The egress VID is set to ingress VID. The *RouterIdGlort* should be set to 15.

The unit computes a hashing function using L3/L4 fields before indexing the table. The output of this hash function is used to compute the final index and is typically used for implementation of ECMP or load balancing (note that since the hashing function is on L3/L4 fields, load balancing cannot be implemented in the ARP unit for non-IP frames). The details on the hashing function can be found in the hashing chapter.

Note: ARP table entry 0 is reserved and cannot be used for routing.

The 16-bit ARP index received as part of the route-arp command points to the first entry in the ARP table. The 4-bit ARP count defines the number of entries, including the first one, that could be used as alternative paths and uses the result of the hash function to compute an ARP index depending on the path count type:

```
If path count type = 0, then
    If ARP count == 0
       then PathCount == 16
       else ARP count = PathCount
    ARP_TABLE Index = ARP Index + (hash[11:0] * PathCount) / 4096
If path count type = 1, then
    ARP count = PathCount, ARP values of 13..15 are reserved and not used
    ARP_TABLE Index = ARP Index + hash[11:0] &  ((1<<PathCount)-1)
```

This supports ECMP modulo of 1..16, 32, 64, 128, 256, 512, 1024, 2048 and 4096. The ARP table is 16 K x 64 and contains three fields:

- Type of entry (2 bits)
  - — Indicates the type of entry (MAC, MACfromIPv6, GLORT) and how the rest of the table is interpreted.
- If MAC address:
  - — Next-hop MAC address (48 bits).
  - — Egress VLAN (12 bits).

- If MAC address extracted from IPv6:
  - Next-hop MAC address (48 bits).
  - Egress VLAN (12 bits).
- If GLORT:
  - Destination GLORT. Always replace the received destination GLORT.
  - 3-bit MTU size index.
  - Egress VLAN. Should be set to 0b for routing IP multicast frames.
- Parity (1 bit)

Note: If a destination GLORT is specified, a MAC Lookup is still done but only for the source address lookup for TCN updates. If a MAC address is specified, the MAC lookup is done normally (both destination and source).

5.7.5.2 ARP_USED Table

The ARP_USED table is a 16 K x 1 table (implemented as a 512 x 32) to indicate if the corresponding entry in the ARP_TABLE has been used recently or not. The CPU may clear the bit by writing a 1b to it, while writing a 0b has no effect. The hardware automatically sets the bit whenever an entry is used. Software may poll this table periodically to detect which entries where used recently, and delete unused entries from the table if needed by writing back the read value.

The ARP_USED table is updated at the time the header is processed and is updated regardless of whether the packet has a framing or data errors.

5.7.5.3 IP Traps & Logs

The switch is capable of capturing the following IP frames:

- Trap routed IP unicast frames with TTL = 1 or TTL = 0.
- Log routed IP multicast frames with TTL = 1 or TTL = 0.
- Trap IGMP Frames.
- Trap IP frames with certain options.
- Trap MTU size violations.
- Log routed IP unicast to same LAN for potential redirect.
- Log routed IP unicast received as MAC multicast.

5.7.5.3.1 Trapping of IP Frames with Options

The switch has the capability of trapping IP frames with options to the CPU. The trapping is globally enabled by turning on the SYS_CFG_ROUTER.*trapIPOptions*, and also requires configuring the PARSE_CFG[i] register in the parser to enable the options of interest for each port.

In the FM10000 the *trapIPOptions* configuration is expanded to either not trap, trap all routed frames, or trap all frames.

5.7.5.3.2 Trapping and Logging of Frames with TTL=1 or TTL=0

The switch has the capability of trapping (unicast) or logging (multicast) IP routed frames with a TTL of 1 or 0. This feature is useful for tracing routes in a network by doing a hop-by-hop route discovery. There are 3 options as configured in SYS_CFG_ROUTER.*trapTTL*:

For routed unicast frames:

1. Discard frames with TTL 1 or TTL 0.
2. Trap ICMP frames with TTL 1 or TTL 0 and discard all others.
3. Trap all frames with TTL 1 or TTL 0.

For routed multicast frames:

1. Do not route frames with TTL 1 or TTL 0.
2. Log ICMP frames with TTL 1 or TTL 0 and do not route all others.
3. Log all frames with TTL 1 or TTL 0.

Note: This field only applies to frames that are routed. If a unicast or multicast frame is not routed, this frame is switched normally regardless of its TTL value. For multicast frames that are both switched and routed, the frame is still switched within the VLAN it was received from regardless of its TTL value. The routing part is conditional on the TTL value received and the configuration of this field.

5.7.5.3.3 Trapping of Frames That Exceed MTU Size

The MTU size is checked for any IP routed frame. The switch supports up to 8 different MTU sizes which are stored in the MTU_TABLE. A 3-bit index is defined in the ARP_TABLE (only for arp-GLORT entries) and in the EGRESS_VID_TABLE. The MTU size defined in arp-GLORT has precedence over the one defined in EGRESS_VID_TABLE and is normally used for multicast while the EGRESS_VID_TABLE is normally used for unicast. If the actual packet length (as contained in the IP header) is greater than this MTU size, the packet is declared oversized for this VLAN and the parameter SYS_CFG_1.*trapMTUViolations* defines the disposition of the frame - either trap to CPU or silently discard.

Every egress VLAN has a default MTU size, which can all be set to 16 K if needed. Any routed frame then overwrites the egress VLAN MTU with whatever is in the ARP Table. This means that all IP frames have their MTU enforced. It also requires that one of the MTU sizes needs to be used for the default value.

In the case of multicast, the arp-GLORT entry is loaded with an MTU index pointing to the smallest MTU possible on any VLAN on the distribution list for this multicast group. There is no per-VLAN control of MTU checking for multicast groups, as the MTU is checked before the replication across VLAN is started.

In the FM10000 the *trapMTUViolations* configuration is expanded to either not trap, trap all routed frames, or trap all frames.

5.7.5.3.4 Logging Redirectable IP Unicast Frames

The switch can detect if a unicast routed packet is headed back to the same VLAN it came from. Such frames are logged to the CPU so that software can generate an ICMP REDIRECT message. A trigger may optionally match on these frames (using the LOG_ARP_REDIRECT action code) to rate-limit or suppress logging of these frames or to take any other action.

5.7.6 Policers

5.7.6.1 Overview

The Frame Processing Pipeline includes a policing unit that monitors the incoming traffic rate and can change the priority or drop packet if needed. The policing unit includes 4 banks: two 4 K-entry and two 512-entry banks. Other than their sizes, the four banks have the same feature set; each can be used to police or count, each bank is configured independently to partition their space into a police partition and a counter partition.

The registers defined for this unit are:

- POLICER_APPLY_CFG_4K[0..1,0..4095] — Defines apply behavior (4 K banks).
- POLICER_APPLY_CFG_512[0..1,0..511] — Defines apply behavior (512-entry bank).
- POLICER_CFG_4K[0..1,0..4095] — Per-entry configuration (4 K banks).
- POLICER_CFG_512[0..1,0..511] — Per entry configuration (4 K banks).
- POLICER_STATE_4K[0..1,0..4095] — Counters/token-buckets (4 K banks).
- POLICER_STATE_512[0..1,0..511] — Counters/token-buckets (4 K banks).
- POLICER_CFG[0..3] — Bank global configuration.
- POLICER_SWEEPER_PERIOD_CFG — Refresh rate.
- POLICER_DSCP_DOWN_MAP[0..63] — Priority remapping.
- POLICER_SWPRI_DOWN_MAP[0..15] — Priority remapping.

The FFU produces four indexes:

- COUNT_0 — Index to 4 K bank 0.
- COUNT_1 — Index to 4 K bank 1.
- COUNT_2 — Index to 512 bank 0.
- COUNT_3 — Index to 512 bank 1.

A index of 0b can be used to disable the corresponding bank.

The register POLICER_CFG[bank].*IndexLastPolicer* defines the index of the last policer entry for that bank. Any index above that limit indicates that those entries are used for counting. Setting this limit to 0b implies that the entire bank is used for counting.

The POLICER_SWEEPER_PERIOD_CFG defines the sweeper period and if the sweeper is enabled. The sweeper is responsible for replenishing the token buckets periodically for policing, and must be enabled for policing to function properly. The sweeper is not required if all the banks are used for counting.

The sweeper scans through the four banks in parallel, one entry per clock cycle, index 0..4096. For each index, the sweeper goes through the policers active at that index: 1 to *IndexLastPolicer* or policer bank size. The index 0 is thus always a no-op.

Once the sweep has terminated, the sweeper is dormant until the next sweeper period.

The sweeper is a lower priority task than packet processing or any other fabric register access. If the pipeline is empty, the entire sweep terminates in ~6 ns (4096/700 MHz). In a heavy traffic scenario, the fabric scheduler is always setup to reserve some time for register (for example, 20 MHz). Therefore, if there is no management access but full line rate of traffic, the time to perform as sweep is ~200 ns (4096/20 MHz). However, if there is lots of registers access, then the sweep might take much longer.

If a sweep takes more time than a sweeper period, that sweeper period is skipped, creating an imprecision in sweep period and thus policer rates. It is unlikely that register accesses and constant minimum size packet rate keeps the pipeline busy so much that the sweep is delayed forever, but the sweeper at least covers the case for worst packet rate.

The recommendation is to set the period to twice the time required to sweep 4096, assuming highest packet rate, so 279,762 (400 ns @ 699.404 MHz), which is roughly 2 x 4096/20 MHz based on the assumption that 20 MHz or more was reserved for management access by the fabric scheduler. The default value is 73,685 (~105 ns).

5.7.6.2 Policing

Policing entries are dual token buckets named “committed” and “excess”. Tokens are used from the “committed” bucket first. When the “committed” TB is depleted, packets conceptually are marked as yellow, and use the “excess” TB's tokens.

Counting entries are implementing a packet counter and a byte counter.

The policer entries are looked up on the head of the frame by the COUNT_0...COUNT_3 variables set by the FFU using following algorithm:

```
# Configuration must be set to POLICER_CFG[0..3].IngressColorSource=2

# Current priorities
#    swpri = frame's SWPRI after parsing and FFU actions
#    dscp = frame's DSCP after parsing and FFU actions

# Green=2, Yellow=1, Red=0
dscpColor = Green
swPriColor = Green

# Process each bank
foreach bank (0..3) {

    index = COUNT_<bank>

    # Recover state and config
    overCommitted = POLICER_APPLY_CACHE[bank][index].OverCommitted
    overExcess = POLICER_APPLY_CACHE[bank][index].OverExcess
    committedAction = POLICER_APPLY_CFG[bank][index].CommittedAction
    excessAction = POLICER_APPLY_CFG[bank][index].ExcessAction
    markDSCP = POLICER_CFG[bank].MarkDSCP;
    markSwitchPri = POLICER_CFG[bank].MarkSwitchPri;

    # Check over committed
    if (overCommitted && overExcess) bankColor[bank] = Red
    elseif ( overCommitted ) bankColor[bank] = Yellow
    else bankColor[bank] = Green

    # Drop if requested
    if (bankColor[bank] = Red && excessAction == 1) then drop=1
    if (bankColor[bank] = Yellow && excessAction == 1) then drop=1

    # Downgrade if requested
    if (markDSCP && bankColor[bank] < dscpColor) dscpColor = bankColor[bank];
    if (markSWPRI && bankColor[bank] < swpriColor) swpriColor = bankColor[bank];
}

# Apply downgrade if required
if ( dscpColor == Red )
    # Down twice
    dscp = POLICER_DSCP_DOWN_MAP[POLICER_DSCP_DOWN_MAP[dscp]]
if ( dscpColor == Yellow )
    # Down once
    dscp = POLICER_DSCP_DOWN_MAP[dscp]
if ( swpriColor == Red )
    # Down twice
    swpri = POLICER_SWPRI_DOWN_MAP[POLICER_SWPRI_DOWN_MAP[swpri]]
if ( dscpColor == Yellow )
    # Down once
    swpri = POLICER_SWPRI_DOWN_MAP[swpri]
```

The color for each bank (bankColor[bank]) and respective indexes are preserved for this frame until the end of frame is received, and then used for policer token bucket updates.

If the resulting color for DSCP is yellow, it is down-marked once. If the resulting color for DSCP is red, it is down-marked twice. Similarly for SWPRI, if the resulting color for SWPRI is yellow, it is down-marked once, and if it is red, it is down-marked twice.

As an example, a frame comes in and hits two banks. The first bank sets its color to red but only MarkDSCP is set for that bank, while the second bank sets its color to yellow and has MarkDSCP and MarkSWPRI both set. In that case the frame exits with an updated DSCP to red (twice down-marked) and an updated switch priority to yellow (once down-marked).

The token buckets of each bank are potentially decremented on the tail of the frame depending on the bankColor[bank] captured with that frame and the disposition of the frame.

For each bank:

- Neither token buckets are decremented if the frame is not forwarded (no copies sent to any port, including mirror ports), or if bankColor[bank]= red.
- The excess token bucket is decremented if bankColor[bank] = yellow.
- The committed token bucket is decremented if bankColor[bank] = green.

If token buckets are decremented, they are decremented by the size of the frame as received, ignoring the tail error of that frame (good frame, bad FCS or framing error). If any token bucket becomes empty after being decremented (from greater than 0 to equal to 0 or negative), the corresponding POLICER_APPLY_CACHE[bank][index] states are updated to reflect the new state.There is a delay between when the token buckets are decremented, and new states are reflected back the head. Therefore, a particular token bucket might go negative by at least one frame, regardless of its size, and up to several frames if the frames are small.

Finally, the policer sweeper replenishes all token buckets periodically. Each token bucket is programmed using the following fields:

- *RateMantissa* (4 bits)
- *RateExponent* (5 bits)
- *CapacityMantissa* (4 bits)
- *CapacityExponent* (5 bits)

For example, if a 100 GbE policer is to be monitored, and the sweeper period is 400 ns, the rate setting is:

```
rate = 100G/8 * 400us                     => 5,000,000

RateExponent = floor(log2(rate)-3)        => 19
RateMantissa = ceil(rate/2**RateExponent) => 10

actualRate = 10 * 2**19 * 8 / 400us       => 104G
```

The minimum rate is with *RateExponent*=0 and *RateMantissa*=1 and result of a rate of 20 Kb/s. The capacity is normally set to a value higher than twice the rate.

If a token bucket becomes positive after replenishing (from 0 or negative to greater than 0), the corresponding POLICER_APPLY_CACHE[bank][index] state is updated to reflect the new state.

5.7.6.3 Handling Dropped Frames

Frames that match on a policer bank but are completely dropped by the switch (the destination mask of the frame sent to the scheduler is zero) are not debited against matching policer token buckets. This includes frames that are policer-dropped due to a token bucket being over rate on the head of the frame.

Note: The only exclusions to this rule are frames dropped due to an FCS error on a non-head segment, frames dropped in the modify, or frames dropped in the PCIe block for loopback suppression.

5.7.6.4 Differentiated Services (DiffServ)

DiffServ uses an 8-bit field for the IP header to carry priority information about the frame. This field structure is shown here:

0	1	2	3	4	5	6	7
DSCP						ECN	

The Differentiated Services Code Point (DSCP) defines the class of service while the Explicit Congestion Notification (ECN) reports congestion in the forwarding direction. The switch forwarding supports traffic conditioning as required by DiffServ which includes the following: meter, marker, shaper and dropper.

- The 'meter' and 'marker' are detailed together in the policing section.
- The 'shaper' is detailed in the Scheduler chapter.
- The 'dropper' is detailed in the Policing and Congestion Management of this chapter.

5.7.7 Layer 2 Lookup

5.7.7.1 Overview

The layer 2 lookup unit uses the following fields from the frame header to alter the course of the packet.

- Ingress VLAN association / Source MAC Address / Source GLORT
  - Used for learning, security checking and ingress VLAN/Spanning Tree filtering.
- Egress VLAN association / Destination MAC Address / Destination GLORT
  - Used for forwarding and egress VLAN/Spanning Tree filtering.

The layer 2 lookup handling involves the following table structures:

- **MAC Address Table** (MA_TABLE) — Maintains MAC address port (GLORT) associations, either by VLAN or by classes of VLANs. Table entries are written to the table by software, either as dynamic (can age) or static (does not age).
- **Ingress VLAN Table** (INGRESS_VID_TABLE) — Contains security and configuration information associated with the VLAN on which a frame arrives.
- **Egress VLAN Table** (EGRESS_VID_TABLE) — Contains security and configuration information associated with the VLAN on which the frame is sent.
- **Ingress Spanning Tree Table** (INGRESS_MST_TABLE) — Maintains the state of the spanning tree instance to which the frame belongs on ingress.
- **Egress Spanning Tree Table** (EGRESS_MST_TABLE) — Maintains the state of the spanning tree instance to which the frame belongs on egress.
- **Tagging Table** (MOD_VLAN_TAG_VID1_MAP, MOD_PER_PORT_CFG_4) — Determines whether frames egressing on a particular (Port, VLAN) should be 802.1Q-tagged.
- **MA Table Change Notification FIFO** (MA_TCN_FIFO) — Notifies software of SMAC lookup misses and port moves so that software can learn new MA_TABLE entries and station moves, and process security violations.
- **MA Table Used Table** (MA_USED_TABLE) — Provides a bit per MA_TABLE entry, set when a lookup hit occurs, cleared by software.

The following features are supported by the layer 2 unit.

- Flooding packets when DMAC is not known.
- Detection of SMAC lookup misses, and reporting this to software through the MA_TCN_FIFO.
- Detection of table entry usage (lookup hits) so that software can track which entries have become inactive.
- Detection of MAC movement from one port to another, and reporting this to software. If an SMAC is secured, the MAC movement can be silently dropped or reported to the trigger unit for possible action, including reporting to software.
- Support for up to 256 spanning tree instances with per-port states of DISABLED, LISTENING, LEARNING and FORWARDING.
- Up to 4095 VLANs and up to 4096 Forwarding domains (FIDs).
- Arbitrary mapping of sets of VLANs into forwarding domains and spanning tree instances.

- Output tagging control (enabled/disabled).
  - This table is actually located in the EPL.
- Optional trapping of known addresses (BPDU, LACP, 802.1X).
- Optional trapping of IGMP frames.
- Configurable per-VLAN, per-port membership.
- Configurable per-VLAN, per-port tagging.

5.7.7.2 VLANs

The switch supports 802.1Q Virtual LANs (VLAN) with the following VLAN association and security capabilities:

- Each port has a default VLAN ID and default priority.
- Per port VLAN association, ingress rule is one of the following:
  - Untagged packets received on one port are associated with the default VLAN ID and have the default priority configured for this port.
  - For tagged packets, each port can be configured to either (1) leave the VLAN ID and VLAN priority as is, or (2) overwrite the VLAN ID and VLAN priority fields with the port's default values.
  - If there is more than one VLAN tag in the frame:
    - The parser identifies one as VLAN1 and one as VLAN2 according to TPID and/or position.
    - Only VLAN1 is mapped to a FID for processing by L2 Lookup.
    - VLAN2 may be processed by the FFU.
  - The FFU can be programmed to change the VLAN association for any type of packet.
- Per port VLAN “security settings”, which can be set to any of the following:
  - Discard all untagged packets on ingress, including frames with a VLAN tag with VID=0 (PARSER_PORT_CFG_2).
  - Discard all tagged packets on ingress (PARSER_PORT_CFG_2). This feature is useful but not required in IEEE 802.1Q.
  - Discard packets if the ingress port is not a member of the VLAN per the INGRESS_VID_TABLE and VLAN ingress boundary violation checking is enabled (PORT_CFG_3).
  - The SMAC in a packet should not be learned if the ingress port is not a member of the VLAN in which the packet arrived.
  - Discard packets if the egress port is not a member of the VLAN per the EGRESS_VID_TABLE (VLAN egress boundary violation). The SMAC is be learned unless the VLAN has no member ports.
  - Further discard rules can be applied using the FFU.

The disposition of packets subject to VLAN security rules is summarized in Table 5-20.

Table 5-20 VLAN Security Packet Disposition Summary

IBV Enabled	Ingress Port in Ingress VLAN	Egress Port in Egress VLAN	TCN Event?	Software Learn?	Drop?	Action Code/ Stat?
X	Yes	Yes	Yes	Yes	No	FIDForwarded
X	Yes	No	Yes	Yes	Yes	VlanEgressDrops
No	No	Yes	Yes[1]	No	No	FIDForwarded
No	No	No	Yes[1]	No	Yes	VlanEgressDrops
Yes	No	X	No	(No)	Yes	VanIngressDrops

[1] An egress boundary violation always generates a TCN FIFO event. Therefore, in the case where the ingress port is not a member of the ingress VLAN, software must drop the TCN FIFO event so the SMAC is not learned.

The VLAN information is divided into different tables, listed here:

- INGRESS_VID_TABLE (4096 entries, indexed by VLAN ID)
  - *membership* (48 bits) — Port membership for this VLAN, 1 bit per port.
  - *FID* (12 bits) — Forwarding domain: learning group for this VLAN.
  - *MST_Index* (8 bits) — Multiple spanning tree instance index for this VLAN on ingress.
  - *CounterIndex* (6 bits) — Group 11 counter index to count frames ingressing on this VLAN.
  - *reflect* (1 bit) — Whether to permit L2 switched frames to egress on the ingress port (refer to Section 5.7.8).
  - *TrapIGMP* (1 bit) — Whether to trap or switch IGMP frames on this VLAN.
- EGRESS_VID_TABLE (4096 entries, indexed by VLAN ID)
  - *membership* (48 bits) — Port membership for this VLAN, 1 bit per port.
  - *FID* (12 bits) — Forwarding domain: learning group for this VLAN.
  - *MST_Index* (8 bits) — Multiple spanning tree instance index for this VLAN on egress.
  - *MTU_Index* (3 bits) — Index into MTU_TABLE for routed frames.
  - *TrigID* (6 bits) — Egress VLAN trigger identifier.

5.7.7.3 Spanning Trees

The switch maintains state for up to 256 spanning tree instances. Each VLAN is mapped to a spanning tree instance in the INGRESS_VID_TABLE and EGRESS_VID_TABLE registers. The following table indicates each of the spanning tree states provided and the effect of each state on:

- **RX BPDU** — Reception of Spanning Tree Protocol BPDUs.
- **TX BPDU** — Transmission of Spanning Tree Protocol BPDUs.
- **RX Traffic** — Forwarding of received non-network management frames.
- **TX Traffic** — Transmission of non-network management frames.
- **Learning** — Whether unknown or moved SMACs on frames cause a new TCN FIFO entry.

Note that IEEE network management frames besides STP BPDUs, such as LACP and 802.1x, are always forwarded and transmitted, regardless of spanning tree state.

State	RX BPDU	TX BPDU	RX Traffic	TX Traffic	Learning
DISABLED	No	No	No	No	No
LISTENING	Yes	Yes	No	No	No
LEARNING	Yes	Yes	No	No	Yes
FORWARDING	Yes	Yes	Yes	Yes	Yes

The spanning tree information is divided into two tables, listed here:

- INGRESS_MST_TABLE (256 entries for ports 0-23 and 256 entries for ports 24-47)
  - *STPState* (48 bits) — 2-bit spanning tree state for each port in this spanning tree instance.
- EGRESS_MST_TABLE (256 entries for all ports)
  - *Forwarding* (48 bits) — Per-port forwarding enable, 1 bit per port.

5.7.7.4 MAC Address Table

The MAC Address Table (abbreviated MA Table or MA_TABLE) associates layer 2 MAC addresses with the destination port(s) to which frames addressed to those addresses should be forwarded. The MAC table is implemented as two hash tables, each consisting of four sets of 4096 entries. One hash table is searched for the Destination MAC and the other for the Source MAC. Entries are classified by FID, derived from the VLAN, allowing for Independent VLAN Learning, Shared VLAN Learning, or any hybrid configuration.

Ports are stored in the tables as GLORTs, since they might need to identify more than a single physical port, such as for a Link Aggregation Group. The hash table used to store DMACs in combination with the egress FID produces the destination GLORT to which the frame should be forwarded. The hash table used to store SMACs in combination with the ingress FID produces a source GLORT to detect MAC movement.

Each 96-bit MA_TABLE entry includes the following fields:

- *MACAddress* (48 bits) — MAC Address.
- *FID* (12 bits) — Learning group.
- *valid* (1 bit) — Indicates if entry is in use.
- *secure* (1 bit) — Indicates frame should not be reported to TCN FIFO if source GLORT does not match destGlort, and silently dropped. Ignored on DMAC lookups.
- *glort* (16 bits) — Destination GLORT on DMAC lookups, source GLORT for SMAC lookups. During SMAC lookups, this field is compared to the canonicalized frame's source GLORT to identify MAC moves.
- *TrigId* (6 bits) — An ID associated with the entry for use in trigger matching rules.

5.7.7.5 MA Table Lookup

The MA_TABLE lookup is done by computing a 12-bit hash index from a 60-bit number constructed by concatenating the 48-bit MAC address with the 12-bit FID. An independent hash index is computed for each of the 4 sets, and each of the four entries is compared with the MAC/FID pair from the frame. If any one of the entries matches the MAC/FID pair, this is a hit, otherwise, it is a miss.

Note: If MA_TABLE_CFG_1.*HashMode* is set to 1b, then a single hash index is computed and used for all 4 sets. This mode of operation is less efficient than the normal default behavior for random or semi-random MAC addresses, and is reserved for testing purposes.

5.7.7.6 Destination GLORT Determination

The destination GLORT used to forward the frame may come from one of the following places

- An FFU set GLORT.
- FFU set flood-set.
- FFU route.
- The MA Table Lookup.
- An FTAG DGlort.

The following steps are used to determine the destination GLORT:

- Each frame can either be FFU switched, FFU flood-set or FFU routed, if the FFU takes any forwarding action at all. If multiple forwarding actions are specified in different slices, the FFU precedence is used to determine which one is acted upon.
- If a route was chosen by the FFU, the dglort either comes directly from the NextHop or comes from the MAC table. This causes FTAG DGlort to be ignored. If a dglort is specified explicitly in the route action, the result of the MA Table Lookup is ignored.
- If a switch GLORT was chosen by the FFU, this takes highest precedence, and the FTAG DGlort and MA destination lookup are ignored.
- If the FFU does not specify a GLORT, and the frame was tagged with a non-zero DGlort in an FTAG this will set the GLORT.
- Otherwise, the MA table lookup sets the destination GLORT if there is a match in the table.
- If a flood-set GLORT was chosen by the FFU, this GLORT is used if the MA table misses, thus it is lower precedence than a match in the MA table.
- In the absence of any GLORT from an FFU action, an MA table hit, or an FTAG DGlort, and if the frame is a broadcast frame, the broadcast GLORT is used. Otherwise the flood GLORT is used.

The following logic determines which destination GLORT to use after the destination MAC lookup, depending on the *DGlortFromFFU*. *DGlortFromFFU* is zero except in the case where the FFU sets a switch or route action, or in the absence of that if the frame has an FTAG DGlort. The *FloodSet* bit is zero except in the case where the route-glort action sets this bit.

```
if (DGlortFromFFU != 0) & (Flood-Set == 0)
  DGLORT = DGlortFromFFU
else if (DMAC-EgressFID-Lookup == HIT) & (DGlortFromMA_TABLE != 0)
  DGLORT = DGlortFromMA_TABLE
else if (DGlortFromFFU != 0) & (Flood-Set == 1)
  DGLORT = DGlortFromFFU // Flood-Set GloRT
else if (DMAC == Broadcast)
  DGLORT = BroadcastGlort
else
  DGLORT = FloodGlort
```

5.7.7.7 MA Table Management

5.7.7.7.1 Table Access

The MAC Table's entries are visible to software and can be read and written as atomic 3-word registers. Software must directly write entries into the table through this interface, taking care to allocate entries to the appropriate hashed index. The MA_TABLE addressing is defined as follows:

Field	Bit(s)	Description
Word	1:0	Identifies one of three words in the 96-bit entry. Word number 3 is reserved.
Index	13:2	Indexes one of 4096 hash table bins.
Set	15:14	Identifies one of four sets in the entry's hash table bin.

Entries are hashed based on the entry's FID value and MAC address. Given this 60-bit key, a folded Ethernet CRC calculation is performed to provide a set of four 12-bit hash values. The computation is as follows:

```
// Compute key
 key [63:0] = {4'd0, fid, mac_addr}; // KEY[7:0] = MAC_ADDR[7:0] (LSB)
                                     // ... ... ...
                                     // KEY[47:40] = MAC_ADDR[47:40] (MSB)
                                     // KEY[55:48] = FID[7:0]
                                     // KEY[59:56] = FID[11:8]
                                     // KEY[63:60] = 0
 hash[0] [11:0] = CRC32( key[63:0 ] );
 hash[1] [11:0] = CRC32({key[7:0], key[63:8 ]});
 hash[2] [11:0] = CRC32({key[15:0], key[63:16]});
 hash[3] [11:0] = CRC32({key[23:0], key[63:24]});

 // Search for mac address (t=0 for DMAC, t=1 for SMAC)
 match = -1;
 for (i=0;i<4;i++)
 {
     index = hash[i] + 4096 * i;
     if ( MA_TABLE[t,index].MAC == mac_addr && MA_TABLE[t,index].FID == fid )
        match = i;
 }
```

5.7.7.7.2 MAC Table Change Notification FIFO

The hardware uses the MAC Table Change Notification FIFO (MA_TCN_FIFO) to communicate SMAC table lookup miss events and source port move events to software. Each entry in the FIFO contains the reason for the notification and all information necessary for software to add a "learn" entry into the MAC Table, update the source port, or report a security violation.

The MA_TCN_FIFO has a capacity of 511 change notifications.

Relevant registers:

- MA_TCN_FIFO[0..511]:
    - — *MACAddress* (48 bits) — MAC address.
    - — *VID* (12 bits) — Ingress VLAN ID.
    - — *srcGlort* (16 bits) — The frame's source GLORT that cause the notification. For non FTAG-ed ports, the srcGlort is redundant information, as the srcGlort could be derived from ports. For FTAG-ed ports, the srcGlort is from FTAG.SGLORT (unless 0), and represents the original source GLORT associated with the port that received the packet on a multi-switch system.

- — *Index* (14 bits) — For port moves only, the MA_TABLE index for the MAC address.
- — *Port* (6 bits) — Physical source port from which the MAC address arrived.
- — *EntryType* (1 bit) — Indicates whether this is a new SMAC (lookup miss) or port move.

- MA_TCN_PTR_HEAD:
  - — *Head* (9 bits) — Entry in the MA_TCN_FIFO that software reads next via MA_TCN_DEQUEUE. RO by software.
- MA_TCN_PTR_TAIL:
  - — *Tail* (9 bits) — Entry in the MA_TCN_FIFO that hardware writes to next. Updated by hardware. RO by software.
- MA_TCN_DEQUEUE:
  - — Same bit fields as MA_TCN_FIFO.
  - — *Valid* (1 bit) — Indicates if an entry was available in MA_TCN_FIFO and successfully read.
  - — Reading from this register has the same effect as reading the entry indicated by MA_TCN_PTR_HEAD.*Head* from the MA_TCN_FIFO (it has the same data), except that it also causes MA_TCN_PTR_HEAD.*Head* to increment, and includes the Valid bit to indicate whether an entry was available.
- MA_TCN_WM[0..47] (indexed by port):
  - — *Wm* (9 bits) — The maximum number of TCN FIFO entries that may be consumed by the port.
- MA_TCN_USAGE[0..47] (indexed by port):
  - — *Usage* (9 bits) — The number of TCN FIFO entries consumed by the port. This field is RW for validation purposes.

The MA_TCN_WM register may be used to prevent a flood of new SMAC addresses on one port from preventing learning on another port. A value of 10 for each port ensures that every port has TCN FIFO space available in a 48-port configuration. A port's allocation of the TCN FIFO is considered consumed when:

```
MA_TCN_USAGE[port] >= MA_TCN_WM[port]
```

Setting a MA_TCN_WM value of zero prevents any TCN FIFO entries from that port, though per-port learning is usually controlled by PORT_CFG_3.*LearningEnable*.

The MAC Address Table Change Notification FIFO (MA_TCN_FIFO) has head and tail pointers exposed to software. A new event is written to the tail pointer location in the FIFO and the tail pointer is incremented. Software advances the head pointer by reading from MA_TCN_DEUQUEUE.

The FIFO is empty when:

```
When MA_TCN_PTR_HEAD.Head == MA_TCN_PTR_TAIL.Tail
```

The FIFO is full when:

```
(MA_TCN_PTR_TAIL.Tail+1)%512 == MA_TCN_PTR_HEAD.Head
```

Note that this simple full condition limits the usable capacity of the MA_TCN_FIFO to 511 entries. In the full condition, if hardware needs to add a 512th event to the FIFO, it instead drops the notification event and raises the Overflow interrupt bit in MA_TCN_IP.

The TCN entry types are listed in Table 5-21.

Table 5-21 TCN FIFO Entry Type

EntryType	Encoding	Description
NewSource	0	Entry was missed in the SMAC lookup table.
MACMoved	1	Non-secure entry was found in the SMAC lookup table, but the destination GLORT found in the table does not match the source port of the ingress frame.

Only one event notification may be enqueued for an individual frame. The TCN FIFO entry is not put into the FIFO if any of the following conditions are true:

- The trigger unit choses “Do Not Learn” as the learning action.
- The TCN_FIFO is full.

5.7.7.7.3 MA_TCN_FIFO Interrupts

Relevant registers:

- MA_TCN_IP:
  - *PendingEvents* (1 bit) — Indicates that an entry has been added to the TCN FIFO. May be used to notify software that the TCN FIFO needs servicing without having to poll MA_TCN_DEQUEUE.
  - *TCN_Overflow* (1 bit) — Indicates that an event could not be added to the TCN FIFO because the FIFO was full.
- MA_TCN_IM:
  - *PendingEvents* (1 bit) — Mask PendingEvents interrupts.
  - *TCN_Overflow* (1 bit) — Mask TCN_Overflow interrupts.

The *TCN* bit in the FH_TAIL_IP register is set whenever a bit is set in MA_TCN_IP and has a zero in the corresponding bit in MA_TCN_IM. The interrupt propagates to the CPU if the *TCN* bit in FH_TAIL_IM is also zero. Hardware only sets bits from 0b to 1b in MA_TCN_IP. Software must explicitly write a 1b to clear the MA_TCN_IP bits, even if the condition that originally caused the interrupt no longer applies.

5.7.7.8 Frame Error Handling

SMAC events are posted to the notification FIFO only at the end of the frame, once the frame has been determined valid by the Ethernet Port Logic or by the PCIe interface. The actual SMAC lookup is performed at the head of the frame. Therefore, if an event needs to be posted, the device keeps this event entry in a temporary location (one entry per port is sufficient) until the end of the frame is validated, and then posts into the FIFO.

The MA_USED_TABLE is updated at the head of the frame. This is done regardless of whether the frame terminates with a framing error, a CRC error, or a parsing error.

5.7.7.9 Source MAC Processing

The second hash table (MA_TABLE[1,*]) can be set to look up SMAC/ingress FID. If the SMAC/IFID misses in the table, the frame's source information (SMAC, FID, source GLORT) may be put into the MA_TCN_FIFO for handling by software.

If there is a hit in the table, the *MA_USED* bit for that entry is set. This notification bit can be used by software to perform aging of the table. When a hit is found in the table, the hardware compares the source GLORT field found in the table with a canonicalized source GLORT for the frame. If these entries do not, this condition is passed to the triggers for handling (frame drop, trap, etc.). If the entry is not "secure", and learning is enabled on the port on which this frame ingressed, a MacMoved entry is placed in the TCN_FIFO along with the MA_TABLE index and new source GLORT (not canonicalized).

MACs may move within a LAG (different ports) without triggering a MacMoved event in the TCN_FIFO. Thus a hardware mapping from LAG member GLORT to canonical LAG GLORT must be defined. This mapping function is implemented as a small sixteen-entry CAM, configured in the CANONICAL_GLORT_CAM registers. Figure 5-21 illustrates its operation.

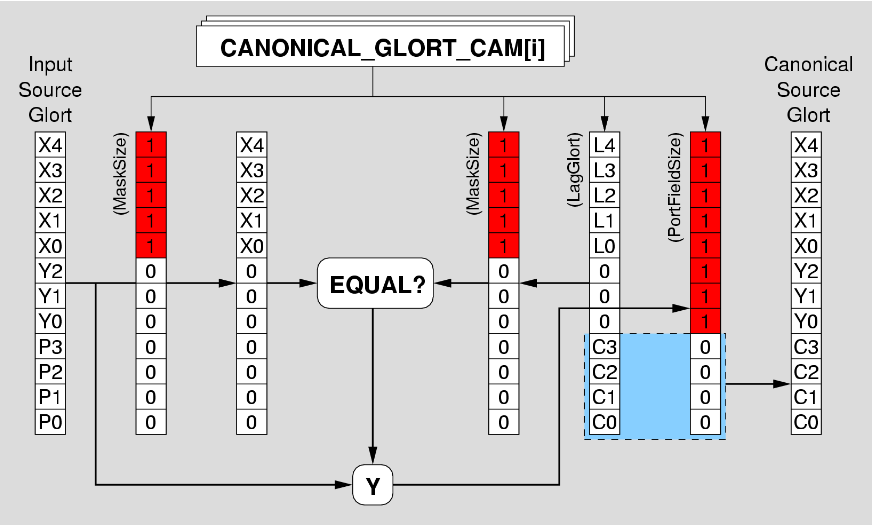


Figure 5-21 Canonical GLORT CAM

In this example, the X0..X4 portion of the GLORT identifies a subset of the overall GLORT space reserved for LAG use. Within that scope, each individual LAG is identified by the Y0..Y2 field. For each LAG, the P0..P3 field identifies both the canonical GLORT (C0..C3) as well as the port member GLORTs (P0..P3 ≠ C0..C3). Note that for a given number of PortFieldSize bits, the maximum number of members per LAG is $2^{\text{PortFieldSize}}$-1 since one value must be used for the canonical GLORT value.

Note: This canonical source GLORT is only used for detection of MAC movement (secure and non-secure), and does not affect the source GLORT used in later stages or the source GLORT placed inside an ISL tag. For a multi-chip system, the size of the canonical GLORT CAM may restrict the number of LAGs that differ in the number of bits that encode the port. Most multi-chip systems use only a couple different bit-widths for LAG port encoding (for example 3-bit for <8 port LAGs and 5-bit for <=16 port LAGs) such that the capacity of this CAM is not exceeded.

5.7.7.10 Special Packet Handling

The switch can perform special handling of certain well-known frame types without requiring additional lookup or FFU resources. Any other protocol or address not discussed below may be handled using the MA_TABLE, triggers and/or FFU resources.

5.7.7.10.1 Invalid Source MAC Addresses

An SMAC of all zeros (00:00:00:00:00:00), all ones (FF:FF:FF:FF:FF:FF) or multicast (bit 40 is a 1b) are considered invalid SMAC addresses. Such frames may be optionally dropped with no TCN FIFO entry generated, as indicated by SYS_CFG_1.dropInvalidSMAC.

5.7.7.10.2 CPU_MAC as Source MAC

If a frame is received with an SMAC that matches the value in CPU_MAC, no TCN FIFO entry is generated.

5.7.7.10.3 Reserved Multicast Destination MAC Addresses

The switch has the ability to trap (or drop) layer 2 packets identified by a DMAC of the form 01-80-C2-00-00-XX where XX may range from 00 to 3F. Such frames include:

- **BDPU** (Spanning Tree): 01-80-C2-00-00-00
- **LACP** (Link Aggregation Control Protocol): 01-80-C2-00-00-02
- **Port Authentication** (802.1X): 01-80-C2-00-00-03
- **LLDP** (Link Layer Discovery Protocol): 01-80-C2-00-00-0E
- **GARP** (both GMRP and GVRP): 01-80-C2-00-00-20 and 01-80-C2-00-00-21, respectively

How each of these packets is handled is controlled with the IEEE_RESERVED_MAC registers:

- IEEE_RESERVED_MAC_ACTION:

  Index 0: 01-80-C2-00-00-00 through 01-80-C2-00-00-1F
  Index 1: 01-80-C2-00-00-20 through 01-80-C2-00-00-3F

  — *action* (2 bits per MAC address):

  00b = Switch normally (as a general multicast address)
  01b = Trap (sent to the CPU GLORT defined in TRAP_GLORT register)
  10b = Drop
  11b = Log

  Note: For some of the reserved MAC addresses, logging or switching normally are improper behaviors for standards-compliant switches.)

- IEEE_RESERVED_MAC_CFG:

  — *trapPri* (4 bits) — The switch priority to optionally assign to a reserved MAC address packet that is trapped

- IEEE_RESERVED_MAC_TRAP_PRIORITY:
  - *select* (1 bit per MAC address):

    0b = If packet is trapped (or logged), its switch priority remains unchanged.
    1b = If packet is trapped (or logged), the trapped packet is given the switch priority specified in IEEE_RESERVED_MAC_CFG.

5.7.7.10.4 Trapping of IGMP and MLD Frames

The trapping of IGMP frames can be turned on per VLAN using the INGRESS_VID_TABLE[vid].*TrapIGMP* bit. If this bit is set and the protocol is IGMP (protocol=2) and the frame is IPv4, the frame is trapped, otherwise, the frame is switched normally.

The equivalent concept for IPv6 is MLD and is implemented as a subset of ICMPv6. It always uses a TTL of 1 and the MLD frames are thus trapped using the TTL trap option described in Section 5.7.5.

5.7.7.11 Trigger Configuration

For a regular L2/L3 switch, software may configure the following trigger rules to augment the L2 functionality:

- **Secure Port Action** — TRAP unknown SMACs received on secure ports so that software may disposition, optionally learn and possibly forward the frame.
- **Secure MAC Action** — To take action on a secure MAC seen moved to a different port, a trigger should match on the MAC table entry and *MACMoved* bit from the L2Lookup. This trigger may drop or trap the frame, depending on software configuration.
- **Security Violation Squelching** — May be used to squelch multiple TCN FIFO events for a repeat security violation by updating the SMAC table entry with the violating port and trigId for a low precedence trigger to drop on SMAC Hit and Forward Normally.
- **Unsecure to Secure Port Action** — TRAP known SMAC moving from a non-secure port to a secure port on SMAC Hit, MacMoved and secure RX port.

5.7.8 Destination Mask Generation

The destination mask processing is shown in Figure 5-22.

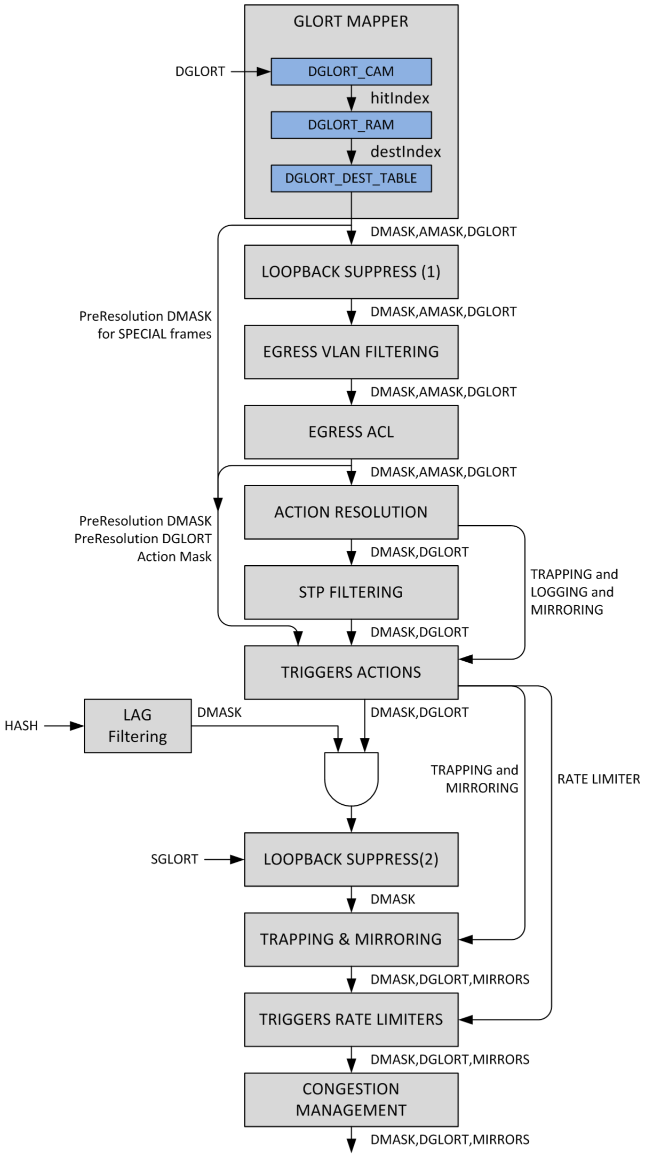


Figure 5-22 Destination Mask Generation Overview

The elements of this processing are detailed in the next sections. The triggers are detailed in Section 5.7.9.

5.7.8.1 Port Mapping Unit

The Port Mapping Unit uses the destination GLORT from earlier stages in the pipeline to retrieve the set of destination ports for the frame as well as an IP multicast table index for packet replication. The set of destination ports is encoded in a 48-bit destination mask.

The destination GLORT may come from different sources:

- The FTAG received with the packet
- The FFU unit
- The NEXTHOP unit
- The L2LOOKUP unit

A packet that is not tagged gets a default destination GLORT of 0. The FFU is instructed whether the frame has a non-zero DGLORT, and may choose to overwrite the DGLORT with an entry in the ARP table (either by setting the DGLORT directly, or by specifying a route action, which clears the DGLORT to zero so that the MAC table can set it). If the DGLORT is not set by the ISL tag or the ARP table, the MAC table's DGLORT result is applied to the frame. The DGLORT may also be changed by the trigger unit—this change to the DGLORT does not change the port mask that the frame transmits on, only the DGLORT that is put on an ISL tag or is used by the PCI sub-system.

The source GLORT may come from 2 different sources:

- The ISL tag
- The ingress port's default

A packet that is not tagged is assigned a source GLORT based on the port it came in on. When received on a PCIe port, this source GLORT is assigned as a function of the frame's virtual switch interface (VSI). Once a source GLORT is set, it cannot be changed until the frame leaves the ISL domain (having the ISL information stripped).

The destination GLORT is searched into a 256-entry 16-bit full ternary CAM. The GLORT RAM is 256x40 bits and there is a one-to-one correspondence between the GLORT CAM and GLORT RAM. If the search is successful, the highest hit line is selected and the associated GLORT RAM is read. If the destGLORT is not found in the CAM, the frame destination mask is set to zero automatically, causing the frame to be dropped (unless triggers change the decision). A single CAM entry may cover a very large set of contiguous destination GLORTs.

The GLORT RAM contains the following information:

- *ParityError* (1 bit) — Indicates the parity correctness status of each entry.
- *Strict* (2 bits) — Controls the method to compute the index into the Destination Mask Table and if the destination mask retrieved from this table is used strictly or ANDed with the destination mask from the layer 2 table or for LAG filtering.
- *DestIndex* (12 bits) — Base index address in the Destination Mask Table.
- *DestCount* (4 bits) — Number of ports in the LAG to which the GLORT entry belongs. A value of 0b represents '16'.
- *RangeSubIndexA* (8 bits) — Controls the mapping from the GLORT value to the Destination Mask entry.
    - — Divided into two 4-bit fields: Length, Offset

- *RangeSubIndexB* (8 bits) — Controls the mapping from the GLORT value to the Destination Mask entry.
  - Divided into two 4-bit fields: Length, Offset
- *HashRotation* (1 bit) — Selects one of two independent hash values for use in the Destination Mask offset calculation.

Note: If the destGlort is not found in the TCAM, then the “strict” field get set to DGLORT_RAM[0].*Strict*.

The content of the GLORT RAM is then used to compute an index in the Destination Mask Table (GLORT_DEST_TABLE, 4 K x 40) which contains the following fields:

- *DestMask* (48 bits) — ANDed with the default destination mask produced by the layer 2 lookup processing.
- *IP_MulticastIndex* (12 bits) — Index relevant for IP multicast; detailed in the IP Multicast section.

Data associated with each CAM entry. The entry defines how to compute the index for indexing the GLORT_DEST_TABLE to retrieve the final destination.

Multiple GLORT values can map to the same GLORT_DEST_TABLE index. Therefore, all 64 K possible GLORT values can map into the 4 K table. The *Strict* field indicates whether the GLORT_DEST_TABLE index is generated deterministically (strict) or by hashing. When strict is used (*Strict* is 0x3, or *Strict* is 0x0 and the ISL tag's FTYPE is either special delivery or management), the index into the GLORT_DEST_TABLE is computed as follows:

```
index = DestIndex + (GLORTB<<widthA) + GLORTA
```

where:

GLORTA = The value of the bits extracted from the GLORT according to *RangeSubIndexA*.

GLORTB = The value of the bits extracted from the GLORT according to *RangeSubIndexB*.

widthA = The number of bits extracted from the GLORT according to *RangeSubIndexA* (indicated by bits 21:18 in *RangeSubIndexA*).

The idea is to provide a single CAM entry to cover multiple LAGs with multiple ports in each LAG where the port numbers are encoded in RangeSubIndexB and the LAG numbers are encoded in RangeSubIndexA. The number of ports in each LAG does not need to be a power of 2 to make efficient use of the GLORT_DEST_TABLE. If the number of LAGs is not a power of 2, a choice can be made between wasting GLORT_DEST_TABLE entries or consuming additional GLORT_CAM/RAM entries by dividing the LAGs into multiple sets, each set containing a number of LAGs that is a power of 2.

When non-strict is used (*Strict* is 0x2, or *Strict* is 0x0 and the ISL tag's FTYPE is either routed or normal), the index into the GLORT_DEST_TABLE is computed as follows:

```
index = DestIndex + ((hash%DestCount)<<widthA) + GLORTA
```

where:

GLORTA = The value of the bits extracted from the GLORT according to *RangeSubIndexA*.

widthA = The number of bits extracted from the GLORT according to *RangeSubIndexA* (indicated by bits 22:19 of *RangeSubIndexA*).

hash%DestCount = A modulo hash over the *DestCount* number of entries in the GLORT_DEST_TABLE per LAG.

If *DestCount* is equal to the highest value that would ever be seen encoded by *RangeSubIndexB*, the difference between strict and non-strict is essentially the difference between hashing over the ports in a LAG and addressing the individual ports specifically (as required by LACP).

The use of sub index A allows for efficient packing of entries in the Destination Mask Table when the entries are shared between different LAGs. The use of the hash function in the index calculation has the effect of balancing traffic across multiple destination masks.

In case of strict routing, the frame hash is not used in the calculation and the destination GLORT is sufficient to determine where the frame goes. The use of strict routing bypasses any filtering done by layer 2 processing including VLAN ingress filtering, VLAN egress filtering, and LAG filtering. However, the trigger can still modify the destination mask if active.

5.7.8.2 Loopback Suppression

Three tables are used to prevent layer 2 packets from flowing back to the port or the link aggregation they came from.

The tables are:

- PORT_CFG_2 (48 x 48 bits)
- INGRESS_VID_TABLE.*reflect* (4096 x 1 bit)
- FH_LOOPBACK_SUPPRESS (48 x 32 bits)

The first two tables are applied early in the mask generation pipeline, while the last table is applied much later (after Triggers).

The FH_LOOPBACK_SUPPRESS table used during destination mask generation is intended to loopback suppress L2 replicated packets by removing bits from the destination mask. This mechanism is used for unicast packets or multicast packets when they are not replicated across multiple VLANs. For frames that are replicated across multiple VLANs on Ethernet ports, the MCAST unit applies a separate stage of loopback suppression for packets that stay on the same ingress VLAN, and the frame copy is dropped on transmit in the modify. Additionally if the MCAST replicates packets into PCIe port, the PCIe port is do a final stage of loopback suppression, using information passed from the MCAST unit via the FTAG.

The process is the following:

- The destination mask generated from DGLORT is ANDed with PORT_CFG_2[rxPort]. This is applied unconditionally on all packet types.
- The INGRESS_VID_TABLE.*reflect* bit is examined. If it is zero, the destination mask bit corresponding to the receive port is cleared. If the *reflect* bit is set, the destination mask bit corresponding to the receive port is left untouched.
- If the frame is not marked Special or Routed (either through the pipeline or on the FTAG FTYPE), then the switch checks every bit of the destination mask. If a bit is found set, the packet is marked to be transmitted to the port indicated by that bit, and the source GLORT is checked to determine if it belongs to the same Link Aggregation Group as the source port. If yes, the destination mask bit is cleared.

In the case of IP multicast packets, the loopback suppression is only checked if the ingress VLAN and the egress VLAN are the same.

5.7.8.3 Egress VLAN/ACL/STP Filters

The egress VLAN/STP filters depends on the egress VLAN ID, the egress Spanning Tree ID, and the following registers:

- EGRESS_VID_TABLE[egressVid].*membership* (48-bit mask)
- EGRESS_MST_TABLE[egressSTPid].*Forwarding* (48-bit mask)

The ACL filter is constructed in the FFU from application of egress ACL rules. For details, refer to Section 5.7.3, "Filtering and Forwarding Unit".

5.7.8.4 Action Resolution

The action resolution stage analyzes the various conditions encountered for this frame and uses the switch configuration to prioritize them into order. The actions are from higher precedence to lowest. For example, a frame received with FTYPE=SPECIAL escapes all followed conditions and is forwarded according to the DGLORT mapping unit. Table 5-22 also indicates if MAC learning is disabled or allowed. If allowed, the learning might still be disabled for that port as per PORT_CFG_3.

Table 5-22 Action Resolution

Action Flag	Learning	Description
SPECIAL	Blocked	Frames with FTYPE set to SPECIAL.
DROP_PARSE_ERR	Blocked	Drop due to header parse error or discard eligible.
DROP_PERR	Blocked	Uncorrectable ECC or parity error detected while doing lookup.
TRAP_MAC_CTRL	Allowed	Trap MAC control frame
TRAP_MAC_CTRL_REMAP	Allowed	Trap MAC control frame at remapped priority.
DROP_CONTROL	Blocked	Drop PAUSE frame.
DROP_MAC_CTRL	Blocked	Drop MAC control frame.
DROP_TAG	Blocked	Drop due to VLAN tagging rules violation.
DROP_SMAC	Blocked	Drop due to invalid SMAC.
DROP_SV_MOVED	Blocked	Drop due to MAC security violation (moved address).
TRAP_CPU_ADDR(15)	Allowed	Trap CPU MAC address.
DROP_IV(16)	Blocked	Drop due to VLAN ingress membership violation.
DROP_INGRESS_STP_NON_LEARN	Blocked	Drop due to ingress STP check (non-learning state).
DROP_INGRESS_STP_LEARN	Allowed	Drop due to ingress STP check (learning state).
DROP_FFU	Blocked	Drop due to FFU action.
TRAP_FFU	Allowed	Trap due to FFU action.
TRAP_ICMP_TTL	Allowed	Trap due to TTL ≤ 1 for ICMP frames.
TRAP_IP_OPTION	Allowed	Trap IP frames with options.
TRAP_MTU_VIO	Allowed	Trap due to MTU violation.
TRAP_IGMP	Allowed	Trap due to IGMP.
TRAP_TTL	Allowed	Trap due to TTL ≤ 1 for non-ICMP frames.
DROP_TTL	Allowed	Drop due to TTL ≤ 1.
DROP_DLF	Blocked	Drop due to flood control of DLF frames (null flood GLORTdest).

Table 5-22 Action Resolution [Continued]

Action Flag	Learning	Description
DROP_CAM_MISS	Dropped	Drop due to GLORT_CAM miss.
DROP_NULL_GLORTDEST	Allowed	Drop due to a null destination mask.
DROP_EV	Allowed	Drop due to VLAN egress violation.
DROP_POLICER	Allowed	Drop due to policer.
DROP_EGRESS_STP	Allowed	Drop due to egress STP check.
DROP_LOOPBACK	Allowed	Drop due to port or VLAN reflection disabled.
GLORT	Allowed	FFU/ARP DGLORT forwarded.
FLOOD	Allowed	Flood due to destination MAC miss in MA_TABLE.
SWITCH_MAC_CTRL	Allowed	Switch MAC control frame.
FORWARD_NORMAL	Allowed	Forward normally.

The following additional attributes can be added in addition to FORWARD_NORMAL action.

Table 5-23 Additional Attributes

Action Flag	Description
LOG_INGRESS_FFU	Frame copied to the CPU as a result of FFU action.
LOG_MAC_CTRL	Log MAC control frame.
LOG_ARP_REDIRECT	Log ARP redirect.
LOG_IP_ICMP	Multicast ICMP frame was copied to the CPU because it's TTL was ≤ 1.
LOG_IP_TTL	Multicast frame was copied to the CPU because it's TTL was ≤1.
HDR_TIMEOUT	Header was too long to be completely parsed.
MIRROR_INGRESS_FFU	Frame was mirrored as a result of FFU action.

All action flags are defined in TRIG_APPLY.*AMASK* enumeration and becomes the action mask to the trigger unit.

5.7.8.5 Link Aggregation

Link-Aggregation is a means of developing more throughput and redundancy between two switches by aggregating point-to-point links together to form one logical port. This aggregation is transparent to the IEEE MAC. Traffic in the same flow destined for that logical port is sent out one and only one of the physical ports.

In the FM10000, all datapath functionality, such as hashing and flood filtering are implemented in hardware. The LACP and Marker frames are trapped and sent to the CPU so that aggregation control process may be implemented in software. The chip's link aggregation operation is fully compatible with IEEE 802.3ad-2000 and IEEE 802.3-2002 clause 43.

Three concepts are used in the FM10000 to implement link aggregation:

- Link Aggregation GLORTs
- Filtering
- Pruning

5.7.8.5.1 Link Aggregation GLORTs

When ports are aggregated into a LAG, there is a need for two types of global resource address:

- **LAG member GLORTs** — These identify the individual physical port members of the LAG. A frame forwarded to such a GLORT egresses from a particular physical port, regardless of the frame's header hashing result.
- **Canonical LAG GLORTs** — These GLORTs identify the trunked port group as a whole, possibly comprising multiple physical ports from different FM10000 devices. A frame identifying this GLORT destination egresses from one of the LAG member ports depending on the result of the frame's header hashing.

Generally, the CPU sends and receives frames to/from LAG member GLORTs to implement the LACP and Marker protocols, while all other frames are addressed by canonical GLORT. When a frame is sent to the CPU its source GLORT association (configured in PORT_CFG_ISL) must be its LAG member GLORT. This allows the CPU to determine the ingress physical port when such frames are trapped. However, a frame's source GLORT must be stored into the MAC Address Table by canonical GLORT. The canonical GLORT CAM in the L2Lookup stage is used to compare only the high order bits of the incoming source GLORT with the source GLORT stored in the source MAC table. This ensures that an SMAC may be observed on any of the ingress LAG ports without triggering a MAC moved event.

5.7.8.5.2 Filtering and Pruning

There are two methods to load balance across multiple ports.

The first method is called **Filtering**. The destination mask table entry pointed to by the GLORT_RAM includes multiple possible destinations and a key in LAG_CFG defines which of these ports are going to actually transmit the frame.

The second method is called **Pruning**. The GLORT_RAM includes a base pointer, a count and a hash function which is used to select which destination mask is going to be used.

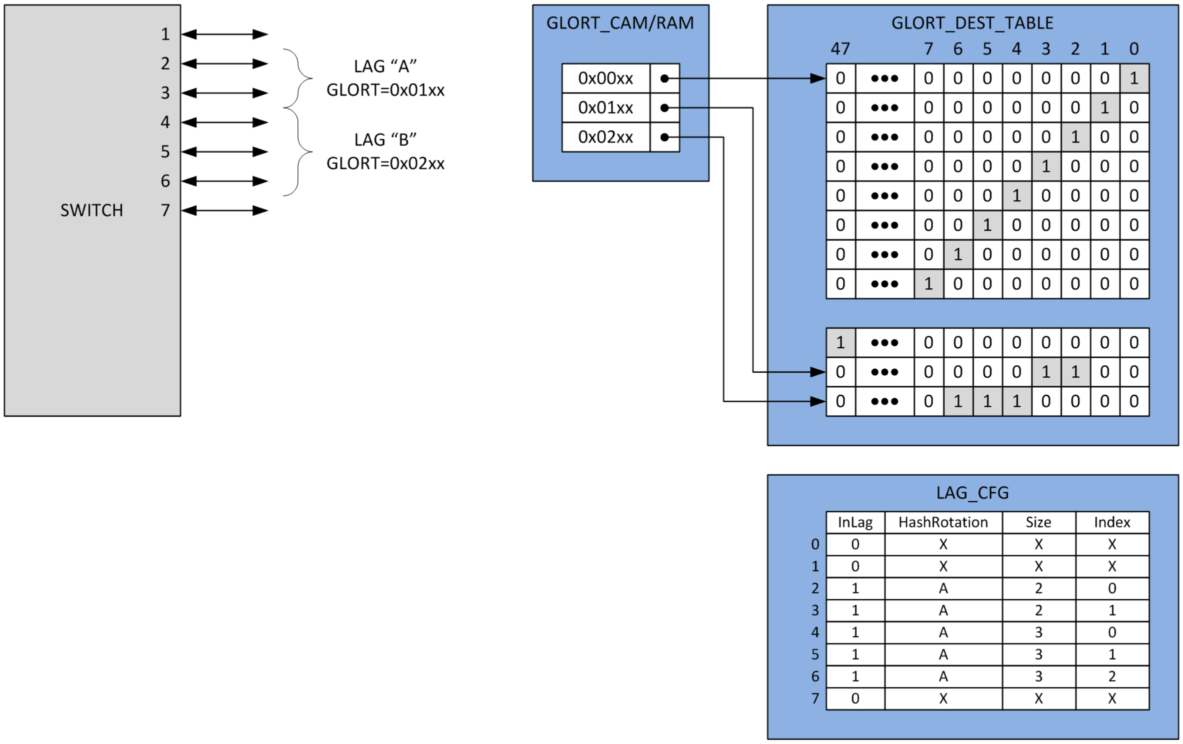
The **Filtering** method is the recommended method for single switch system while a combination of **Pruning** and **Filtering** is required for more complex systems.

The exact processing steps done by the switch are:

1. The hash module computes 2 keys (A and B) from incoming data. One is normally used for balancing traffic across multiple chips while the other one is used to balance traffic across the current chip. Note that the chip has different methods of computing A and B, and may thus load balance traffic differently across multiple stages of switches.
2. The GLORT_RAM defines if a hash function is used, which one (A or B), and which divider is selected (1 to 16). The index is computed from the base index and the remainder of the hash key computed divided by the modulo. This is the pruning step.
3. Then the per-port register set LAG_CFG[0..47] defines the hash function to use (A or B), the divider to use (1 to 16) and the remainder to watch for. The fields of this register are:
   - *LagSize* (4 bits — 0b means 16)
   - *Index* (4 bits)
   - *HashRotation* (1 bit)
   - *InLag* (1 bit)

5.7.8.5.3 Example A – LAG within One Switch

In this first example, a single switch supports 2 LAG groups, A and B. The first one has 2 ports (2 and 3) while the second has 3 ports (4, 5 and 6). All other ports are single ports and not members of any LAG.



GLORT_DEST_TABLE

47		7	6	5	4	3	2	1	0
0	•••	0	0	0	0	0	0	0	1
0	•••	0	0	0	0	0	0	1	0
0	•••	0	0	0	0	0	1	0	0
0	•••	0	0	0	0	1	0	0	0
0	•••	0	0	0	1	0	0	0	0
0	•••	0	0	1	0	0	0	0	0
0	•••	0	1	0	0	0	0	0	0
0	•••	1	0	0	0	0	0	0	0
1	•••	0	0	0	0	0	0	0	0
0	•••	0	0	0	0	1	1	0	0
0	•••	0	1	1	1	0	0	0	0

LAG_CFG

	InLag	HashRotation	Size	Index
0	0	X	X	X
1	0	X	X	X
2	1	A	2	0
3	1	A	2	1
4	1	A	3	0
5	1	A	3	1
6	1	A	3	2
7	0	X	X	X

Figure 5-23 LAG Filtering in a Single Switch System

The solution is to assign GLORTs 1 through 47 for the individual ports that are not part of a LAG and assign GLORT 0x1XX and 0x2XX for the LAG “A” and the LAG “B” respectively. This is shown in the next table.

Port	Default GLORT
1	1
2	0x101
3	0x102
4	0x201
5	0x202
6	0x203
7	7

For example, when a frame needs to be transmitted to GLORT "0x100", the GLORT_CAM matches 0x1xx and the corresponding GLORT_RAM is read, which points to an entry into the GLORT_DEST_TABLE. The destination mask at that location includes port 2 and 3 as possible destinations. The hash key computed from the content of the packet is presented to the lag filtering which is configured with the same modulo (2) but different remainder (0 or 1) causing half the flows to go to one link while the second half goes to the second link.

5.7.8.5.4 Example B – LAG within a Two-level Fat Tree

In this second example, five switches are interconnected together to form a multi-switch system. They are all managed from a single point. The system is shown with a single LAG on the external ports that spans two switches. The basic problem to solve is to load balance the traffic from W to both the inner LAG group as well the outer LAG group in such a way that the traffic is well balanced across all links.

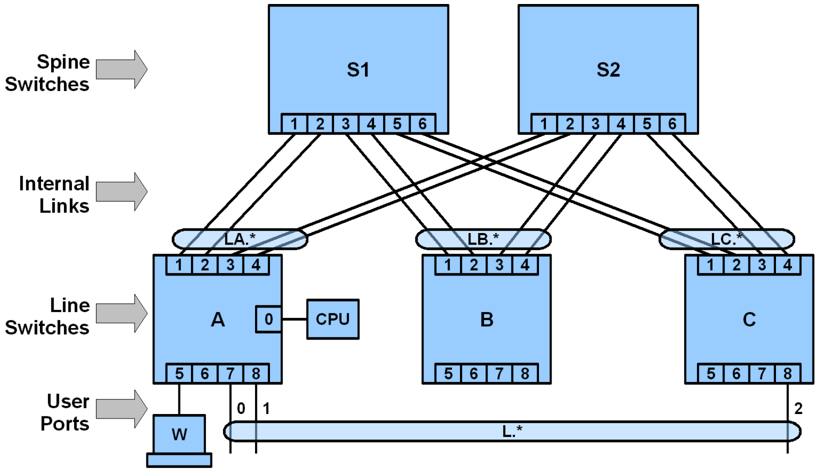


Figure 5-24 LAG Filtering in a Multi Switch System

The following nomenclature is used:

- X.Y identifies a port where "X" is the group/switch it belongs to and "Y" is the port within that group.
- X.* identifies all ports for that switch or group.
- X.0 is the canonical for LAG groups.
- The tuple X.Y can be numerically presented as X * 32 + Y and use directly as a GLORT.

The example above includes the following elements:

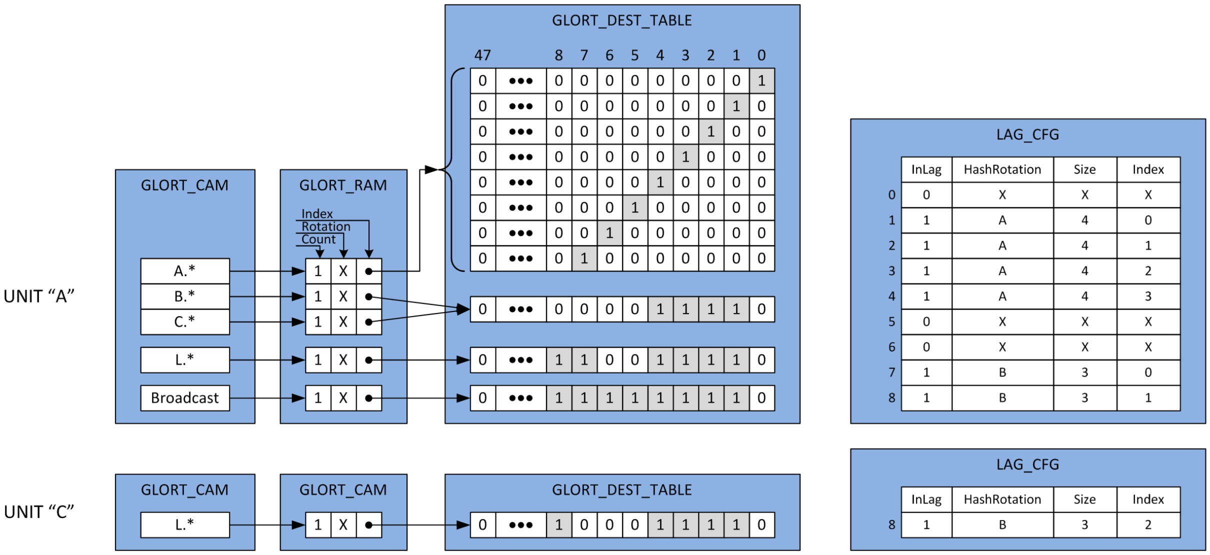
- The switch A, B, C are line switches with user ports and internal links.
- The switch S1 and S2 are spine switches and do not include any user ports.
- The LAG L is a link aggregation group from a user group.
- The LAGs LA, LB, LC are link aggregation groups used to balance the traffic across the multiple spine chips.
- The workstation W is sending traffic.

The requirement is to balance the traffic from workstation W to the different destinations.

The simplest method for implementing this topology would be the following:

- **Switch "A" and "C"** — Two different hash functions are used and configured to generate different keys. The hash function "A" is used to balance toward the spine chip (LAG LA.*) while the hash function "B" is used to balance toward the LAG L.*
- **Switch "A" and "C"** — The GLORT CAM includes the following entries:
    - A.*
    - B.*
    - C.*
    - L.*
    - Broadcast GLORT
- **Switch "A"** — The corresponding GLORT_RAM includes the following entries:
    - A.* points to a block of 48 entries in the GLORT_DEST_MASK using a strict indexing method. Each of the GLORT_DEST_MASK entry points to a specific port.
    - B.* points to a single entry in the GLORT_DEST_MASK which include ports 1,2,3,4 as possible destinations. Any packet going to GLORT B.* are hashed across ports 1,2,3,4 via the LAG filter, which uses a different index for each port.
    - C.* points to the same entry as B.*
    - L.* points to a single entry in the GLORT_DEST_MASK which include ports 1,2,3,4,7,8 as possible destinations. Any packet going to GLORT L.* are hashed across ports 1,2,3,4,7,8.
    - Broadcast GLORT points to a single entry in the GLORT_DEST_MASK which includes all ports, including CPU.
- **Switch "C"** — The corresponding GLORT_RAM includes the following entries:
    - A.* points to a single entry in the GLORT_DEST_MASK which includes ports 1,2,3,4 as possible destinations.
    - B.* points to the same entry as A.*.
    - C.* points to a block of 48 entries in the GLORT_DEST_MASK using a strict indexing method. Each entry of the GLORT_DEST_MASK points to a specific port.
    - L.* points to a single entry in the GLORT_DEST_MASK which includes ports 1,2,3,4,8 as possible destinations.
    - Broadcast GLORT points to a single entry in the GLORT_DEST_MASK which includes all ports except CPU.

The configuration is illustrated in Figure 5-25 (only pertinent ports are shown).



GLORT_DEST_TABLE (UNIT "A")

47		8	7	6	5	4	3	2	1	0
0	•••	0	0	0	0	0	0	0	0	1
0	•••	0	0	0	0	0	0	0	1	0
0	•••	0	0	0	0	0	0	1	0	0
0	•••	0	0	0	0	0	1	0	0	0
0	•••	0	0	0	0	1	0	0	0	0
0	•••	0	0	0	1	0	0	0	0	0
0	•••	0	0	1	0	0	0	0	0	0
0	•••	0	1	0	0	0	0	0	0	0
0	•••	0	0	0	0	1	1	1	1	0
0	•••	1	1	0	0	1	1	1	1	0
0	•••	1	1	1	1	1	1	1	1	0

LAG_CFG (UNIT "A")

	InLag	HashRotation	Size	Index
0	0	X	X	X
1	1	A	4	0
2	1	A	4	1
3	1	A	4	2
4	1	A	4	3
5	0	X	X	X
6	0	X	X	X
7	1	B	3	0
8	1	B	3	1

GLORT_DEST_TABLE (UNIT "C")

0	•••	1	0	0	0	1	1	1	1	0

LAG_CFG (UNIT "C")

	InLag	HashRotation	Size	Index
8	1	B	3	2

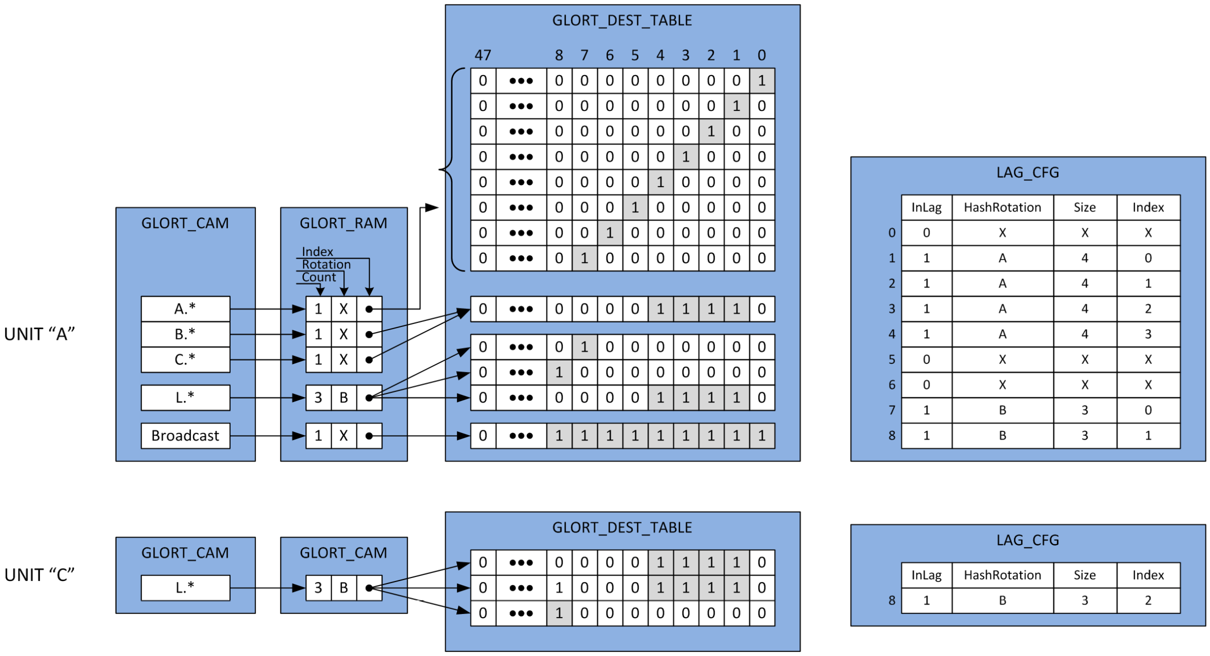
Figure 5-25 Configuration Example for LAG Filtering in a Multi-Switch System

This solution is simple and may be extended easily to support a very large configuration but has the inconvenient characteristic of duplicating some of the packets going from W to L.*. The packets going to L.2 are properly filtered out from L.0 and L.1 and are hashed across LA.* and finally reach switch "C" and then L.2. However, the packets going to L.0 and L.1 are not filtered out from LA.* and thus are sent twice, once on either L.0 or L.1 and once on the LA.* group. The extra packets sent on LA.* are received by the switch C, which filters them out, avoiding duplication on L.*, but extra bandwidth is lost on the way to spine chips transporting packets that are filtered out anyway.

An alternative approach is to use pruning to avoid sending the packets going to L.2 to the inner links (LA.*). The set up becomes the following:

- **Switch "A" and "C"** — Hashing function usage is not changed.
- **Switch "A" and "C"** — The GLORT CAM configuration is not changed.
- **Switch "A"** — The corresponding GLORT_RAM for L.* is changed:
  - — L.* points to three entries in the GLORT_DEST_MASK, one for each port in L.*. The ports L.0 and L.2 points to ports 7 and 8 respectively while the third entry for port L.2 includes the LA.* links. A first level of hashing is used to distribute the traffic across the three entries while a second level of hashing (filtering) is used across the LA.* links.
  - — BROADCAST points to one entry in the GLORT_DEST_MASK. The filtering is used to ensure that only one packet gets eventually transmitted to the L.* group.
- **Switch "C"** — Similarly, the GLORT_RAM for L.* is changed:
  - — L.* points to three entries in the GLORT_DEST_MASK. Two are identical and include LA.* only, and one will include L.2. The hash function is used to distribute the traffic equally among the three entries.
  - — BROADCAST points to one entry in the GLORT_DEST_MASK. The broadcast is transmitted on the spine switch and all other ports. The LAG receives only one copy due to the lag filtering still applied in the leaf switches (A,B).
  - — Note that the LAG filtering for ports A.7, A.8 and C.8 are not superfluous for unicast (as the L.* is already pruned), but must remain in place for broadcast.

This new configuration is shown in Figure 5-26.



Unit "A" GLORT_DEST_TABLE:

47		8	7	6	5	4	3	2	1	0
0	•••	0	0	0	0	0	0	0	0	1
0	•••	0	0	0	0	0	0	0	1	0
0	•••	0	0	0	0	0	0	1	0	0
0	•••	0	0	0	0	0	1	0	0	0
0	•••	0	0	0	0	1	0	0	0	0
0	•••	0	0	0	1	0	0	0	0	0
0	•••	0	0	1	0	0	0	0	0	0
0	•••	0	1	0	0	0	0	0	0	0
0	•••	0	0	0	0	1	1	1	1	0
0	•••	0	1	0	0	0	0	0	0	0
0	•••	1	0	0	0	0	0	0	0	0
0	•••	0	0	0	0	1	1	1	1	0
0	•••	1	1	1	1	1	1	1	1	1

Unit "A" LAG_CFG:

	InLag	HashRotation	Size	Index
0	0	X	X	X
1	1	A	4	0
2	1	A	4	1
3	1	A	4	2
4	1	A	4	3
5	0	X	X	X
6	0	X	X	X
7	1	B	3	0
8	1	B	3	1

Unit "C" GLORT_DEST_TABLE:

0	•••	0	0	0	0	1	1	1	1	0
0	•••	1	0	0	0	1	1	1	1	0
0	•••	1	0	0	0	0	0	0	0	0

Unit "C" LAG_CFG:

	InLag	HashRotation	Size	Index
8	1	B	3	2

Figure 5-26 Configuration Example for LAG Pruning in a Multi-Switch System

5.7.8.6 Trapping and Mirroring

The Action Resolution and Trigger stages might cause the frame to be trapped, logged and/or mirrored.

If a frame is trapped, the destination mask is changed to CPU_TRAP_MASK_FH and the DGLORT is changed to one of the following:

- If the frame is trapped due to the action resolution stage, the DGLORT is changed to TRAP_GLORT with the lower 8 bits changed to reflect the trap cause (Section 5.7.1.5 defines the trap codes).
- If the frame is trapped by a trigger, the DGLORT is changed to TRAP_GLORT with the lower 8 bits replaced with the index of the trigger that caused the action.

If a frame is trapped by action resolution and trapped again by a trigger, the trap code from the trigger unit takes precedence.

The logging and mirroring actions are orthogonal to the normal frame processing. There are two mirrors (mirror 0 and mirror 1) that can be asserted for a given frame. The "log" function as defined in FFU and action resolution stage and trigger stages use mirror 0. The "mirror" function defined in FFU uses mirror 1.

The triggers could also force an action to use a specific mirror (mirror 0 or mirror 1).

Each mirror can use one of the 64 available profiles. The mirror profile defines:

- The expected egress port for this mirror command (defined in FH_MIRROR_PROFILE_TABLE[profile].*port*).

- The tagging information when the frame exit the device (defined in MOD_MIRROR_PROFILE_TABLE[profile].*port*).

If a given frame has two mirrors enabled and they both point to the same profile, two copies are sent out of the device, and both exit the same port and have the same egress modification. Thus, they are indistinguishable from each other.

Note: The parameter SYS_CFG_1.*enableTrapPlusLog* defines if a frame can be both trapped and logged at action resolution stage. If this flag is set to 0b, only trap or log can be enabled, and trapping takes precedence. However, the trigger stage can override this flag by forcing a trap, log or mirror on any frames that match the trigger conditions.

The mirror profile used for a given log or mirror action is defined in the following registers:

Table 5-24 Mirror Profile

Register	Description
RX_MIRROR_CFG.*MirrorProfileIdx*	Mirror 1 profile for FFU mirror command.
LOG_MIRROR_PROFILE.*FFU*	Mirror 0 profile for FFU log command.
LOG_MIRROR_PROFILE.*ReservedMAC*	Mirror 0 profile for reserved MAC logging.
LOG_MIRROR_PROFILE.*ARPRedirect*	Mirror 0 profile index on an ARP log (unicast frame routed back to ingress VLAN).
LOG_MIRROR_PROFILE.*ICMP*	Mirror 0 profile index when a multicast ICMP frame is logged because TTL ≤ 1.
LOG_MIRROR_PROFILE.*TTL*	Mirror 0 profile index when any multicast frame is logged because TTL ≤ 1.

If a mirror is activated by the Trigger unit, the mirror profile is defined as part of the Trigger action. For details, refer to the TRIGGER_ACTION_MIRROR[0..63] register and Section 5.7.9, “Triggers”.

5.7.8.7 SPECIAL Frames

Frames received as FTYPE=SPECIAL on an FTAGed port are processed as follows:

- DGLORT MAPPING: executed
- ACTION RESOLUTION: executed, top priority, thus:
  - LOOPBACK SUPPRESS(1): skipped
  - LOOPBACK SUPPRESS(2): skipped
  - INGRESS/EGRESS VLAN FILTERING: skipped
  - INGRESS/EGRESS STP FILTERING: skipped
  - EGRESS ACL: skipped
  - Refer to Section 5.7.8.4 for priority.
- TRIGGER ACTIONS: executed
- LAG FILTERING: executed
- TRAPPING AND MIRRORING: executed
- TRIGGERS RATE LIMITERS: executed
- CONGESTION MANAGEMENT: executed

Frames received as normal frames but marked as trapped become special frames but go through all stages.

5.7.9 Triggers

5.7.9.1 Overview

In addition to the trapping, ACLs, routing, discarding, and forwarding rules described above that implement various protocols, the switch also contains a general set of rules for modifying frames at the last stage of the pipeline. These rules are user programmable and are referred to as “triggers.” A total of 64 triggers are supported.

A trigger is defined by two parts: a Match Condition and an Action Specification. The Match Condition is a programmable Boolean expression. If all of the conditions defined in the expression are true, the trigger “fires” and the Action Specification is applied to the packet. For a given packet, any number of triggers may fire. In the case of conflicting Action Specifications, an Action Resolution step determines exactly how the packet is handled.

While triggers are very general, their placement in the overall frame processing pipeline imposes a fundamental limit to their use. Specifically, the triggers are evaluated after the GLORT mapping stage, egress ACLs, and layer 2 trapping, but before link aggregation filtering and congestion management. The implications of this placement in the pipeline are detailed in the Trigger Actions section below.

5.7.9.2 Trigger Match Conditions

Each trigger specifies a list of conjunctive match conditions involving properties of the frame and corresponding configured trigger parameters. If all of the match conditions evaluate true, the entire trigger matches and is said to ''fire.” The programmable match conditions are listed in Table 5-25.

Table 5-25 Programmable Trigger Match Conditions

Condition	Frame Property	Trigger Parameter(s)
MatchSA (2 bits)	TrigID (6 bits) from the Layer 2 lookup of the frame's source MAC Address. This field is 00x0 if there is no match in the MA Table.	SA_ID (6 bits)
MatchDA (2 bits)	TrigID (6 bits) from the Layer 2 lookup of the frame's destination MAC Address. This field is 0x3F if there is a destination lookup miss.	DA_ID (6 bits)
MatchHitSA (2 bits)	Source MAC Address match in the Layer 2 MA Table. If a source lookup was performed (learning was not disabled on the ingress port) and the address was found in the table, this condition evaluates to true.	—
MatchHitDA (2 bits)	Destination MAC Address match in the Layer 2 MA Table. If the address was found in the table, this condition evaluates to true.	—
MatchHitSADA (2 bits)	Source or Destination MAC Address match in the Layer 2 MA Table or Source or Destination MAC Address miss. The match condition supports following modes: 00b = True if either (or both) addresses are found in the Layer 2 MA Table. 01b = True if either (or both) addresses are not found in the Layer 2 MA Table. 10b = Always true. 11b = Reserved	—
MatchVlan (2 bits)	TrigID (6 bits) from the EGRESS_VID_TABLE lookup.	VID_ID (6 bits)
MatchFFU (2 bits)	TrigID (8 bits) set from the FFU's SET_TRIG action. If no FFU rule hit specifies a SET_TRIG action, this condition cannot match for any configured value of FFU_ID and FFU_Mask. The default FFU_ID produced by the FFU is 0.	FFU_ID (8 bits) FFU_Mask (8 bits)

Table 5-25 Programmable Trigger Match Conditions [Continued]

Condition	Frame Property	Trigger Parameter(s)
MatchSwitchPri (2 bits)	Associated switch priority (4 bits).	SwitchPri (4 bits)
MatchEtherType (2 bits)	First non-VLAN EtherType (16 bits).	EtherType (16 bits) EtherTypeMask (16 bits)
MatchDestGlort (2 bits)	Destination GLORT (16 bits) associated with the frame.	DestGlort (16 bits) DestGlortMask (16 bits)
MatchFrameClass (3 bits)	The frame class as defined by the Layer 2 Destination MAC Address: 00b = Unicast 01b = Broadcast 10b = Multicast 11b = Reserved If the corresponding bit in the configured FrameClassMask is set to 1b, this trigger condition evaluates to true.	FrameClassMask (3 bits)
MatchSrcPort	Ingress port number (5 bits). If the corresponding source port bit in the configured SrcPortMask is set to 1b, this trigger condition evaluates to true.	SrcPortMask (48 bits)
MatchTX (2 bits)	Compare frame destination mask to the configured DestPortMask for this frame. The control is a 2-bit field with the following choices: 00b = Match if (DMASK & TRIGGER_CONDITION_TX[i].*DestPortMask*) == 0. Used to check if a frame is not going to any port in a given set 01b = Match if (DMASK & TRIGGER_CONDITION_TX[i].*DestPortMask*) != 0. Used to check if a frame is going to any port in a given set 10b = Match if (DMASK == TRIGGER_CONDITION_TX[i].*DestPortMask*). Used to check if a frame is going to a set of ports that is identical to a given set 11b = Match if (DMASK != TRIGGER_CONDITION_TX[i].*DestPortMask*). Used to check if a frame is going to a set of ports that is not identical to a given set Selecting 0 and using a DestPortMask of 0 is always a match regardless if the frame is forwarded on any port or dropped.	DestPortMask (48 bits)
MatchRouted	Value of the mark_routed bit from the FFU ROUTE action (0b if no such action was applied to the frame).	RoutedMask (2 bits)
MatchFTYPE	ISL tag FTYPE field.	FtypeMask (4 bits)
MatchHandlerAction	The frame processing pipeline maintains a bit vector of all actions applicable to each frame that it handles. For example, if a frame arrives on a port in the Disabled spanning tree state, a corresponding action bit is set, indicating that the frame should be dropped. At the end of the pipeline, a precedence resolution is performed over all applicable actions to determine the final disposition of the frame. By matching against arbitrary bits in this action bit vector, the triggers have the capability to overrule nearly any handling decision made by the processing pipeline. All of the action codes, log codes and additional trigger codes apply (see TRIG_APPLY.*AMASK* enumerated data type in Section 11.17.2).	HandlerActionMask (64 bits)
MatchRandom	One of two 24-bit pseudo-random numbers is associated with the frame. The trigger matches if the number is less than or equal to 2^RandomThreshold.	RandomNumber (1 bits) RandomThreshold (5 bits)
MatchByPrecedence	—	MatchByPrecedence (1 bits)

Most of the match conditions require a 2-bit Match Case value specifying the sense of the match test:

0 = Match if the frame property does not equal the trigger parameter.

1 = Match if the frame property does equal the trigger parameter.

2 = Match unconditionally.

The *MatchTx* condition uses 2 bits as well, but with slightly different meaning. See the *MatchTx* description in Table 5-25.

Since the *MatchFrameClass*, *MatchSrcPort*, and *MatchHandlerAction* conditions already provide these matching semantics by taking a mask as their configured parameters, they do not require a Match Case assignment. All triggers have default Match Case values of 2 for all conditions. The *SrcPortMask* and *HandlerAction* mask are 0 by default, disabling the triggers from matching any frames by default. To activate a trigger, the *SrcPortMask* must be programed to a non-zero value (selecting which set of ports the frame must come from) and the *HandlerAction* must also be set to non-null to match at least one action.

All triggers are disabled by default.

5.7.9.2.1 Configurable Trigger Precedence

The *MatchByPrecedence* condition controls whether otherwise matching triggers are applied to the frame in parallel or in a precedence fashion based on their constant trigger number. If *MatchByPrecedence* is set to 1b, the trigger can only fire if no other lower-numbered trigger matches since (and including) the last trigger with *MatchByPrecedence*==0. If *MatchByPrecedence* is set to 0b, the trigger fires unconditionally, assuming all other match conditions are satisfied. Table 5-26 illustrates this behavior:

Table 5-26 Trigger Precedence

Trigger Number	MatchByPrecedence	Matches?	Fires?	Explanation
0	0	Y	y	Evaluated in parallel.
1	0	N	N	Begins a new precedence block.
2	1	N	N	
3	1	Y	Y	First trigger in precedence block to fire.
4	1	Y	N	Excluded by Trigger 3 firing.
5	0	Y	Y	Begins a new precedence block.
6	1	Y	N	Excluded by Trigger 5.
7	1	N	N	
8	1	Y	N	Excluded by Trigger 5
...				

Note: Hardware always processes trigger #0 as if its *MatchByPrecedence* is zero, regardless of what is specified in the register configuration.

5.7.9.2.2 Random Matching

The triggers support statistical rate-based firing with the *MatchRandom* condition. This condition matches based on a comparison between a 24-bit pseudo-random number and a *RandomThreshold* rate parameter configured per trigger. Two random numbers are calculated globally, with each trigger specifying which of the two is to be used in its *RandomThreshold* comparison. Additionally, the *MatchRandomIfLess* parameter determines whether the condition matches based on a less-than-or-equal-to comparison or a greater-than-or-equal-to comparison.

The match condition is determined by the following expression:

```
If (MatchRandomIfLess)
      Match = (RandomNumberGenerator\[Trigi.RandomNumber\] <= 2^Trigi.RandomThreshold)
Else
      Match = (RandomNumberGenerator\[Trigi.RandomNumber\] >= 2^Trigi.RandomThreshold
```

The trigger unit computes two new 24-bit *RandomNumberGenerator* every clock cycle calculated by doing 24 iterations of a 31-bit LFSR. The two LFSRs are seeded differently (hardware defined seed), providing two statistically independent random numbers. The pattern replicates every 2^^31 cycles (every 3 seconds at 700 MHz).

The *MatchRandom* condition does not require a Match Case (equal/not-equal/unconditional) configuration. To disable rate-constrained matching, *RandomThreshold* should be set to the maximum value (or any value equal or larger than 24); otherwise, some value between 0 and 23 should be selected.

5.7.9.3 Trigger Actions

Each trigger specifies a list of actions to apply to the frame. Nine sets of mutually exclusive actions are defined in Table 5-27.

Table 5-27 Trigger Actions

Trigger Action Set	Trigger Action	Associated Configuration	Description
ForwardingAction	0 - Leave as-is	—	No change in forwarding.
	1 - Forward	NewDestGlort (16 bits) NewDestGlortMask (16 bits)	Forward the frame with a new destination GLORT. Destination GLORT bits are overridden from *NewDestGlort* whose corresponding *NewDestGlortMask* bits are 1b. This action can also be used to undo a drop decision from earlier in the frame processing pipeline (refer to Section 5.7.1.4.)
	2 - Redirect	NewDestMask (48 bits) FilterNewDestMask (1 bits) NewDestGlort (16 bits) NewDestGlortMask (16 bits)	Override destination mask to *NewDestMask*. If *FilterNewDestMask* is set to 0b, link aggregation filtering is not applied to this new destination mask. If *FilterNewDestMask* is set to 1b, LAG filtering is applied, even if the frame's FTYPE equals 2 or 3. *NewDestGlort* is applied as for the Forward case.
	3 - Drop	DropMask (48 bits)	Do not forward to any ports i with *DropMask*[i]==1.

Table 5-27 Trigger Actions [Continued]

Trigger Action Set	Trigger Action	Associated Configuration	Description
TrapAction	0 - Leave as-is	—	No change in CPU trapping.
	1 - Trap	—	Trap to CPU. This function has higher precedence than SYS_CFG_1.*enableTrapPlusLog* setting. Enabling trapping on a logged frame causes this frame to be logged and trapped even if SYS_CFG_1.*enableTrapPlusLog* is set to 0b.
	2 - Log	—	Log to CPU. This function has higher precedence than SYS_CFG_1.*enableTrapPlusLog* setting. Enabling logging on a trapped frame causes this frame to be logged and trapped even if SYS_CFG_1.*enableTrapPlusLog* is set to 0b.
	3 – Do not Trap or Log	—	Overrides a trap or log action specified from an earlier point in the frame processing pipeline.
MirroringAction	0 - Leave as-is	—	No change in mirroring disposition.
	1 - Mirror	MirrorSelect (1 bit) MirrorProfileIndex (6 bits)	Mirror the frame to the specified port with the specified GLORT. This action overrides the FFU's mirror action. Truncation of the mirrored frame is enabled by setting *MirrorTruncate* to 1. In order for the mirrored frame to be truncated, the corresponding *MirrorTruncationEnable* bit in the egress port must also be set.
SwitchPriAction	0 - Leave as-is	—	No change in the frame's associated switch priority.
	1 - Reassign Switch Priority	NewSwitchPri (4 bits)	Associate *NewSwitchPri* with the frame, for purposes of congestion management and egress queuing. **Note:** This has no effect on the FFU's
VlanAction	0 - Leave as-is	—	No change in VLAN.
	1 - Reassign egress VLAN ID	NewVlan (12 bits)	Override egress VLAN to the specified value. **Note:** This only affects L2-switched and L3-unicast frames. IP multicast replication, if applicable, still produces frames with the same egress VLANs as if this action was not specified.
LearningAction	0 - Leave as-is	—	No change in learning.
	1 - Do Not TCN Update	—	Do not learn the source L2 address into the MAC Address Table.
	2 - Reserved	—	Learn this frame's source L2 address, even if it will be dropped.
RateLimitAction	0 - Leave as-is	—	Do not apply a Trigger Rate Limiter to this frame.
	1 - Rate Limit	RateLimitNum (4 bits)	Assign this frame (and any mirrored copies) to one of 16 Trigger Rate Limiters. A drop mask is applied to the frame if the specified rate limiter's bandwidth limit is exceeded.

As illustrated in Section 5-27, triggers are applied after GLORT mapping, egress ACLs, and action resolution, but before LAG filtering and congestion management.

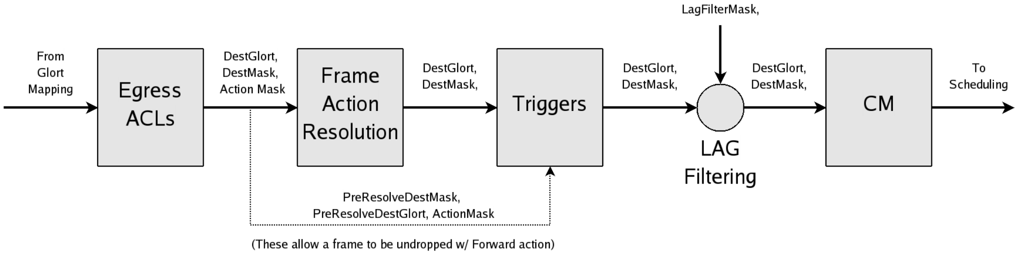


Figure 5-27 Trigger Application

This location in the frame processing pipeline imposes the following constraints on the trigger actions:

- Port numbers and destination mask bits apply to pre-filtered LAG membership ports. Therefore, the *MatchTx* condition cannot match on individual physical ports belonging to a LAG. The Redirect ForwardAction offers a FilterNewDestMask configuration, which, when set to 0b, bypasses LAG filtering and allows frames to be directed to specific physical ports.
- Any assigned destination GLORTs and port masks are applied to the frame directly and must be mutually consistent. There is no further mapping of the specified GLORT.
- Frames are still subject to dropping due to congestion even if they are trapped, redirected, mirrored or otherwise forwarded normally by the triggers.

5.7.9.3.1 Using Triggers to Undrop Frames

The Frame Action Resolution step in the Frame Processing Pipeline applies various hard-coded precedence rules to determine a final handling action for a given frame. For example, a frame might ingress on an invalid (port, VLAN) pair, yet it might also match a Trap FFU rule. The action resolution step determines that the security violation in this case has higher precedence than the Trap rule, causing the frame to be dropped. Since these precedence rules are hard-coded (see the list of Action Codes in the TRIG_APPLY.*AMASK* enumerated data type in Section 11.17.2), there may be a need to fine-tune these precedence decisions in certain applications. The triggers provide this capability by supporting the MatchHandlerAction match condition in conjunction with the Forwarding action.

For purposes of overriding the action resolution decision, the most important operation the triggers provide is “undropping” a frame. This behavior is specified by selecting *ForwardingAction*==1. In this case, if the frame was dropped, the destination mask and GLORT prior to the Frame Action Resolution step is restored. The frame's action handling code is also restored to its original value (this matters only for Group 6 packet counter classification.)

5.7.9.3.2 Rate Limiting

The triggers implement 16 drop-based rate limiters. The rate limiter selected for a given trigger is configurable in TRIGGER_ACTION_CFG_2[i].*RateLimitNum* and enabled in TRIGGER_ACTION_CFG_1[i].*RateLimitAction*=0. It is possible to have multiple triggers asserted for a given frame, and thus have multiple rate limiters enabled. However, hardware only enables one rate limiter for a given frame, ignoring all others. The rate limiter selected is the one pointed by the trigger with the lowest index with a rate limiter enabled.

Each rate limiter consists of a single token bucket with configured Capacity and Rate parameters. These token buckets are:

- Refilled periodically, updating token bucket state if appropriate, empty to non-empty.
- Decremented at frame reception, updating token bucket state if appropriate, non-empty to empty.
- Used to apply drop mask if token bucket is empty.

Each token bucket is refilled every 16 clocks cycles using following expression:

```
refillAmount = (RateMantissa * 32) >> (RateExponent + 2)
```

where *refillAmount* is in units of 1/16 bytes, *RateMantissa* is a 12-bit quantity and *RateExponent* is a 2-bit quantity. For a 700 MHz frame handler clock rate, these parameter definitions give a rate limit range between 2.73 Mb/s and 89.58 Gb/s.

At frame reception, the winning rate limiter token bucket (if any) is decremented by the receive frame size except under the following circumstances:

- An action resolution causes the frame to be dropped.
- A trigger causes the frame to be dropped.
- The enabled rate limiter state is empty (drop mask is applied).
- Congestion Management causes the frame to be dropped entirely.

The status of the token bucket is then reported back as a single bit in the register TRIGGER_RATE_LIM_EMPTY.*Empty*[r]. This register is set to 1b if the token bucket is zero or negative.

Follow on frames check for the *Empty* bit of the enabled rate limiter, and, if found set, applies the drop mask defined in the TRIGGER_RATE_LIM_CFG_2 against the current frame destination mask. The drop mask is also applied to the monitor ports for any mirror copies of the frame. The drop mask is applied (ANDed) after LAG filtering, loopback suppress and trapping operations have been performed, but before congestion management masking.

Since mirrored copies of a frame (whether mirroring is invoked by a trigger or FFU rule) are affected by the rate limiter along with the original frame, the rate limiter's drop mask may be used to selectively apply the rate limiter only to the mirrored copies, only to the original frame or to both, as long as the original frame is not normally forwarded to the mirror's monitor port.

The hardware has a delay between updates to the token bucket (replenishing or decrementing) and its application. This delay is up to 16 frames. So, a given token bucket could go negative by up to 16 x MAX_FRAME_SIZE before it stops being decremented, thus allowing to 16 frames to go through without seeing the drop mask applied.

Here are few examples of possible utilization of rate limiters:

- Limit chip-wide proportion of flooding traffic to protect against denial-of-service attacks.
- Limit low-priority traffic sent or trapped to the CPU in a rate-controlled manner rather than in a watermark-controlled manner.
- L2 policing: Limit bandwidth directed to a particular DMAC, VLAN, or egress port.
- Limit bandwidth of mirrored copies (e.g., for oversubscribed monitor ports).

5.7.9.3.3 Trigger Action Resolution

Note: "Trigger Action Resolution" is not to be confused with the "Frame Action Resolution" step in the Frame Processing Pipeline described in Section 5.7.9.3.1.

When multiple matching triggers fire in parallel (due to the use of *MatchByPrecedence*==0), the set of actions specified by these triggers must be resolved to a single non-conflicting set. There are three general rules governing this resolution process:

1. Whenever possible, non-conflicting actions are all applied.
2. Otherwise, higher precedence trigger actions override lower precedence trigger actions. The action values are defined such that a higher value indicates higher precedence.
3. In all other cases, the actions of lower-numbered triggers override the actions of higher-numbered triggers.

Rule 1 applies to the case of matching triggers with *ForwardingAction*==3 (Drop). Each trigger's DropMask is applied:

```
FinalDropMask = 0
 foreach matching trigger i
    FinalDropMask = FinalDropMask | DropMask[i]
 NewDestMask = ~FinalDropMask & DestMask
```

DropMasks are ORed together for all matching triggers with *ForwardingAction*==3 (Drop) and then applied. The ORed DropMasks are applied before LAG, loopback suppress or trapping, while the Rate Limiter drop mask is applied after all three of those elements.

Rule 2 applies for any other unequal trigger action values. For any two matching triggers i and j, ActionValue[i] is given precedence over ActionValue[j] if ActionValue[i] > ActionValue[j]. For example, the "Reassign egress VLAN" case of *VlanAction*==1 would override the "Leave as-is" case of *VlanAction*==0.

Rule 3 applies for any other equal trigger action values. For any two matching triggers i and j, ActionValue[i] is given precedence over ActionValue[j] if i < j.

In the case that a frame is completely dropped to all destinations due to trigger actions, the frame's action code is set to "TriggerDrop" and is classified accordingly by the Group 6 packet counters.

5.7.9.4 Trigger Counters and Interrupts

Whenever a trigger fires the count associated with that trigger is incremented.

Each trigger is associated with a bit in one of the TRIGGER_IP registers. Whenever a trigger fires, the corresponding IP bit is set.

5.7.10 Congestion Management

5.7.10.1 Overview

The Congestion Management (CM) stage is responsible for tracking memory usage and applying different watermarks for flow control or drop decisions with the goal of providing fair access to the network during congestion.

The FM10000 provides the following mechanisms for managing congestion:

Accounting of memory usage globally, by Shared Memory Partition (SMP), and by RX and TX ports

Each frame is tracked globally, in one of the two Shared Memory Partitions and by the RX port (exactly one RX port association per frame received) and TX port (0 or more associations per frame received). The Shared Memory Partitions allows to partition memory to support two class of services; by example lossy versus lossless. The SMP selection is mapped from the switch priority.

Private and drop watermarks defined per-RX port per SMP

As frames arriving from a particular (Rx and SMP) pair begin to consume too much of the shared frame memory, this mechanism enables subsequent frames to be head-dropped before they are queued. Conversely, any (Rx and SMP) consuming less than its private watermark enables is guaranteed space in the frame memory regardless of the ingress congestion due to other ports.

Global per-SMP drop watermarks

These watermarks are applied to any frames whose SMP usage is above their respective private watermarks.

Global drop watermark

This watermark is applied to any frame.

Per-port, per-SMP soft drop watermarks

These watermarks probabilistically drop frames queued to (Tx and SMP) destinations that are consuming too much of the shared memory. A probabilistic mechanism is employed on Tx to avoid reserving excessive amounts of memory to maintain fair egress scheduling.

Generation of PAUSE frames in response to rising Rx memory consumption

Two styles of pause frames can be generated, one for standard IEEE 802.3 per-port pause, the other for IEEE 802.1Qbb PFC. Port-based pause frames are generated based on Xon/Xoff watermarks referenced to total Rx port and total shared memory usage. Class-based pause frames are controlled by Xon/Xoff watermarks referenced to per-(Rx and RXMP) and per shared SMP memory levels. As long as any Xoff watermark is exceeded, the corresponding port's pause frame is repeated at some timed resend interval configured per port.

Tx per-port and per-class reaction to ingress PAUSE frames

Pause frames identified by the parser might provide the egress scheduler with timed per-port or per-TC pause times. The mechanism supports Xon messages with pause payload-specified IEEE-standard time values, as well as immediate-response Xoff messages. Each pause frame can specify these parameters over a vector of up to eight traffic classes. The FM10000 provides support for standard IEEE 802.3 PAUSE frames and IEEE 802.1Qbb PFC (Priority Flow Control) PAUSE frames.

5.7.10.2 Congestion Management Register Set

The registers set is defined as follows:

Maps:

- CM_APPLY_SWITCH_PRI_TO_TC, CM_SWEEPER_SWITCH_PRI_TO_TC — Maps switch priority (SWPRI) to traffic class (TC). Those two tables must be configured the same way for the unit to function properly.
- CM_APPLY_TC_TO_SMP, CM_SWEEPER_TC_TO_SMP — Maps traffic class (TC) to shared memory partition (SMP). Those two tables must be configured the same way for the unit to function properly.
- CM_TC_PC_MAP[0..47] — Maps traffic class to pause class for each port.
- CM_PC_SMP_MAP[0..47] — Maps pause class to SMP for each port.

Counters:

- CM_APPLY_DROP_COUNT[0..47] — Counts TX drops due to congestion management.

Tracking Memory Usage:

- CM_TX_TC_USAGE[0..47,0..7] — Tracks memory usage per transmit queue (48 ports x 8 traffic class).
- CM_RX_SMP_USAGE[0..47,0..1] — Tracks memory usage per port and per SMP.
- CM_SMP_USAGE[0..1] — Tracks memory usage per SMP.
- CM_SHARED_SMP_USAGE[0..1] — Tracks memory usage per SMP exclusion of RX private area.
- CM_GLOBAL_USAGE — Tracks global memory usage.

Note: USAGE registers might go negative temporarily when near zero.

Current States Applied to Head of Frame:

- CM_APPLY_STATE
- CM_APPLY_SOFTDROP_STATE[0..15]
- CM_APPLY_RX_SMP_STATE[0..47,0..1
- CM_APPLY_TX_TC_STATE[0..47,0..7]
- CM_APPLY_TX_SWPRI_STATE[0..47,0..15]
- CM_APPLY_SOFTDROP_CFG[0..15]
- CM_APPLY_TX_SOFTDROP_CFG[0..47,0..15]

Watermarks:

- CM_GLOBAL_WM
- CM_SOFTDROP_WM[0..15]
- CM_SHARED_WM[0..15]
- CM_SHARED_SMP_PAUSE_WM[0..1]
- CM_RX_SMP_PRIVATE_WM[0..47,0..1]
- CM_RX_SMP_HOG_WM[0..47,0..1]
- CM_RX_SMP_PAUSE_WM[0..47,0..1]

- CM_TX_TC_HOG_WM[0..47,0..7]
- CM_TX_TC_PRIVATE_WM[0..47,0..7]

Pause/RateLim Configuration and Control:

- CM_PAUSE_RESEND_INTERVAL[0..47]
- CM_PAUSE_DECIMATION[0..7]
- CM_SHARED_SMP_PAUSE_CFG[0..1]
- TX_RATE_LIM_CFG[0..47,0..7]
- TX_RATE_LIM_USAGE[0..47,0..7]
- CM_BSG_MAP[0..47]
- CM_EGRESS_PAUSE_COUNT[0..47]
- CM_SMP_PAUSE_EX[0..47,0..1]

Other Configuration:

- CM_PORT_CFG[0..47]
- CM_GLOBAL_CFG

Epoch Tracking:

- MCAST_EPOCH_USAGE[0..1]
- MCAST_EPOCH

5.7.10.3 Congestion Management Blocks

The Congestion Management consists of two basic units:

CM_APPLY

Applies congestion management states at reception of head of the frame. If a frame is accepted, all follow-on segments of the same frame are accepted. If a frame is dropped, all follow-on segments of the same frame are also dropped.

CM_USAGE

Tracks memory usage as segments are received or freed per port. For congestion management purpose, the unit is informed when a segment has been fully transmitted on a given queue on a given port. The unit computes the current states by comparing usage and watermarks, and sends those states to the CM_APPLY unit. There is a delay between memory tracking and state reporting to the APPLY unit equal to a maximum of 200 clock cycles.

The two units are shown in Figure 5-28.

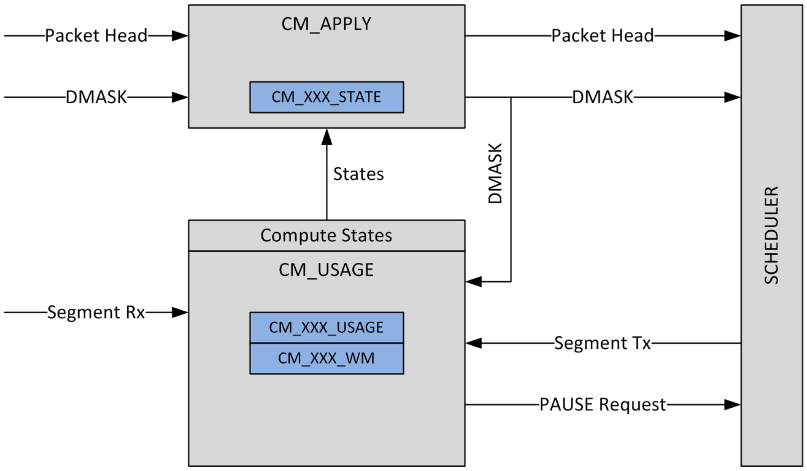


Figure 5-28 Congestion Management Unit Overview

5.7.10.4 Memory Usage Tracking and Watermarks Application

Figure 5-29 shows the global usage tracking and watermarks. Any frame received is counted in the CM_GLOBAL_USAGE. At the start of each frame, the CM_GLOBAL_USAGE is compared to the CM_GLOBAL_WM, which is used to limit the total number of frames allowed enter the device. Normally, the CM_GLOBAL_WM should be set to the maximum number of segments possible (24,576), minus the number of ports enabled x (numbers of segments per maximum size frame), minus 256 reserved segments.

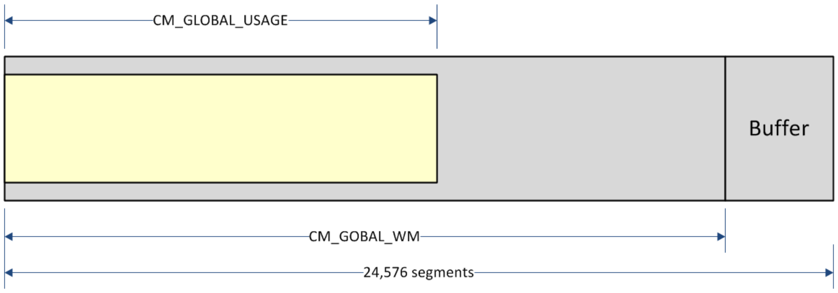


Figure 5-29 Global Memory Tracking and Watermarks

Figure 5-30 shows the per-RX port usage tracking and watermarks. The packets are part of the private area until the CM_RX_SMP_PRIVATE_WM and the excess are part of the shared memory area. Size of queues could be limited per port and per SMP. Pause is controllable per port and per SMP.

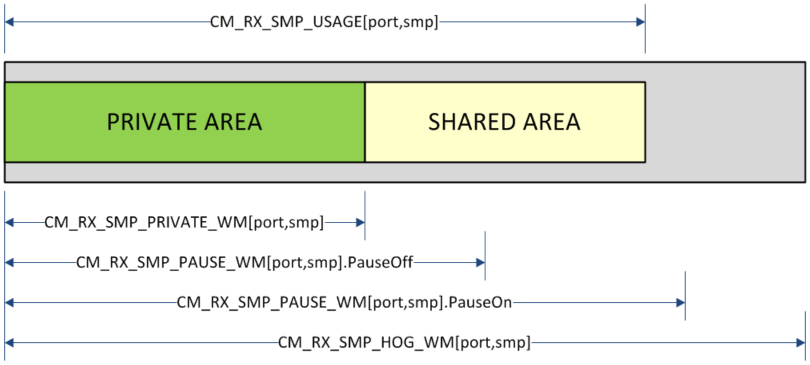

CM_RX_SMP_USAGE[port,smp]
PRIVATE AREA
SHARED AREA
CM_RX_SMP_PRIVATE_WM[port,smp]
CM_RX_SMP_PAUSE_WM[port,smp].PauseOff
CM_RX_SMP_PAUSE_WM[port,smp].PauseOn
CM_RX_SMP_HOG_WM[port,smp]

Figure 5-30 Received Memory Usage Tracking and Watermarks

Figure 5-31 shows the per-SMP usage tracking and watermark. The usage is tracked per SMP and per shared part of SMP. The size of the shared area is limited by CM_SHARED_WM, and the PAUSE level is controllable per SMP. Also, the unit tracks what is left (smpFree) until the shared memory is full for a given switch priority. This is used to apply a probabilistic drop above a soft drop watermark to allow fairness when the partition gets really full. See soft drop probability on page 172 for details.

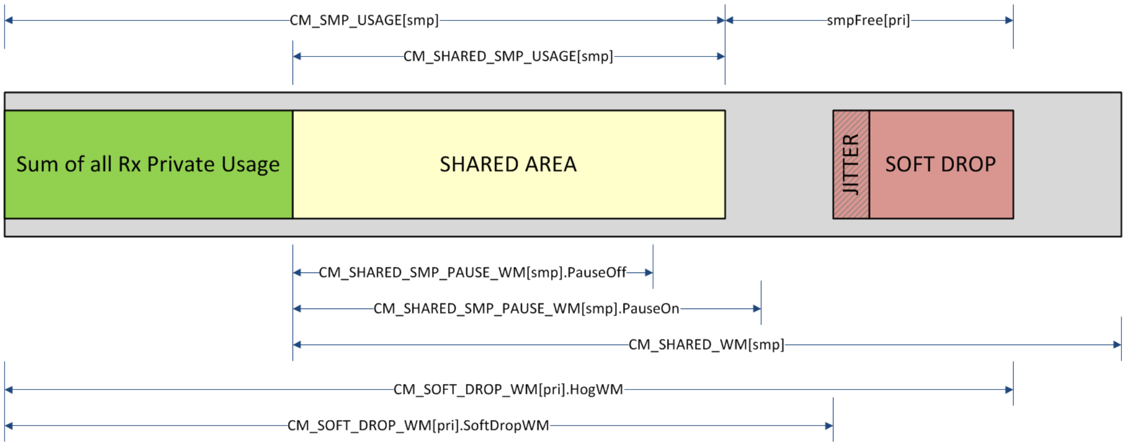

CM_SMP_USAGE[smp]
smpFree[pri]
CM_SHARED_SMP_USAGE[smp]
Sum of all Rx Private Usage
SHARED AREA
JITTER
SOFT DROP
CM_SHARED_SMP_PAUSE_WM[smp].PauseOff
CM_SHARED_SMP_PAUSE_WM[smp].PauseOn
CM_SHARED_WM[smp]
CM_SOFT_DROP_WM[pri].HogWM
CM_SOFT_DROP_WM[pri].SoftDropWM

Figure 5-31 Shared Memory Partition Usage Tracking and Watermarks

Figure 5-32 shows the per-TX port and per-Queue usage tracking and watermarks. Each queue could be set for a private area under which no frames are dropped. Past this level the frame considered using the non-private area, and implements a maximum size (CM_TX_TC_HOG_WM) to limit the size of a queue. If the frame is not part of the private area in RX or TX, it is susceptible to probabilistic drop.

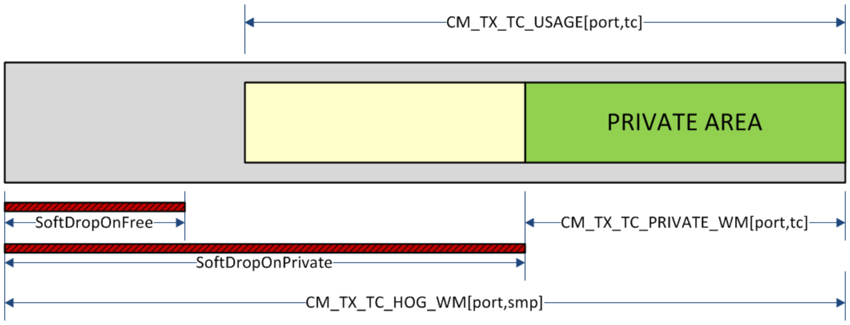


Figure 5-32 Transmit Memory Tracking and Watermarks per Port per Traffic Class

CM_USAGE updates CM_*XXX*_USAGE using the ingress and egress segment updates. CM_USAGE also periodically computes the watermark states and sends updates to CM_APPLY.

The first step is computing the shared SMP usage by summing the difference between the CM_RX_SMP_USAGE and CMP_RX_PRIVATE_WM.

```
# Compute the shared SMP usage
foreach smp (0..1)
    sharedSmpUsage = 0;
    foreach i (0..47)
        sharedSmpUsage += min(0,CM_RX_SMP_USAGE[i][smp]-CM_RX_PRIVATE_WM[i][smp]);
    CM_SHARED_SMP_USAGE[smp] = sharedSmpUsage;
```

The second step is to scan all ports and switch priority and compare the actual usage to the watermarks defined.

```
# For each port
foreach port (0..47)
{
   # For each SWPRI
   foreach swpri (0..15)
   {
      # Using maps
      tc = CM_SWEEPER_SWITCH_PRI_TO_TC[swpri]
      smp = CM_SWEEPER_TC_TO_SMP[tc]

      # Collect Usage
      globalUsage = CM_GLOBAL_USAGE
      smpUsage = CM_SMP_USAGE[smp]
      sharedSmpUsage = CM_SHARED_SMP_USAGE[smp]
      rxSmpUsage = CM_RX_SMP_USAGE[port][smp]
      txTcUsage = CM_TX_TC_USAGE[port][tc]

      # Collect Watermarks
      globalWM = CM_GLOBAL_WM
      sharedWM = CM_SHARED_WM[swpri]
      rxPrivateWM = CM_RX_PRIVATE_WM[port][smp]
      rxSmpHogWM = CM_RX_SMP_HOG_WM[port][smp]
      softDropHogWM = CM_SOFTDROP_WM[swpri].HogSegmentLimit
      softDropWM = CM_SOFTDROP_WM[swpri].SoftDropSegmentLimit
      txTcHogWM = CM_TX_TC_HOG_WM[port,tc]
      txTcPrivateWM = CM_TX_TC_PRIVATE_WM[port,tc]

      # Space left until hard drop limit
      smpFree = min(0,softDropHogWM-smpUsage)

      # Compute States
      # Note: Usage might be negative at moment, all comparisons are signed.
      overGlobal = globalUsage >= globalWM
      overShared = sharedSmpUsage >= sharedWM
```

```
        overRxPrivate = rxSmpUsage >= rxPrivateWM
        overRxHog = rxSmpUsage >= rxSmpHogWM
        overTxPrivate = txTcUsage >= txTcPrivateWM
        overTxHog = txTcUsage >= txTcHogWm
        overFreeSmp = txTcUsage >= smpFree
        overFreeSmp2 = txTcUsage >= smpFree/2
        overSoftDrop = smpUsage > softDropWM
        usageOverLimit = min(-1,max(255,smpUsage - softDropWM))

        # Send to CM_APPLY
        send(overGlobal,CM_APPLY_STATE.global_ex)
        send(overShared,CM_APPLY_STATE.rxs[swpri])
        send(overRxPrivate,CM_APPLY_RX_SMP_STATE[port,smp])
        send(overRxHog,CM_APPLY_RX_SMP_STATE[port,smp].rxHog)
        send(overTxPrivate,CM_APPLY_TX_TC_STATE[port,tc].txPrivate)
        send(overTxHog,CM_APPLY_TX_TC_STATE[port,tc].txHog)
        send(overSmpFree,CM_APPLY_TX_SWPRI_STATE[port,swpri].overSmpFree)
        send(overSmpFree2,CM_APPLY_TX_SWPRI_STATE[port,swpri].overSmpFree2)
        send(usageOverLimit,CM_APPLY_SOFTDROP_STATE[swpri].usageOverLimit)

        # Check for pause ON/OFF
        rxPauseOnWM = CM_RX_SMP_PAUSE_WM[port,smp].PauseOn
        rxPauseOffWM = CM_RX_SMP_PAUSE_WM[port,smp].PauseOff
        sharedPauseOnWM = CM_SHARED_SMP_PAUSE_WM[smp].PauseOn
        sharedPauseOffWM = CM_SHARED_SMP_PAUSE_WM[smp].PauseOff
        sharedPauseEn = CM_SHARED_SMP_PAUSE_CFG[smp].EnableMask[port]

        # Update pause state to OFF if applicable and send PAUSE(OFF) if needed
        # Note: Usage might be negative at moment, all comparisons are signed.
        if ( rxSmpUsage < rxPauseOffWM ||
            (sharedPauseEn && sharedSmpUsage < sharedPauseOffWM) )
             # Request sending one PAUSE OFF packet
             if (pause_state[port,smp]==1) sendPause(OFF)
             # Remember that the link-partner is not paused
             pause_state[port,smp] = 0;

        # Update pause state to ON if applicable and send PAUSE(ON) if needed
        if ( rxSmpUsage >= rxPauseOnWM &&
            (!sharedPauseEn || sharedSmpUsage >= sharedPauseOnWM) )
             # Request sending PAUSE ON packets
             if (pause_state[port,smp]==0) sendPause(ON)
             # Remember that the link-partner is paused
             pause_state[port,smp] = 1;
        }
     }
```

Note: The shared SMP usage only counts what is over the RX private watermark. If a port usage is strictly equal to the RX private watermark, the port is deemed not contributory to the shared SMP. However, the port is considered out of its private area in this case. The following example details the case:

smpUsage	rxPrivateWM	contributionToSharedSMP	overRxPrivate
9	10	+0	0
10	10	+0	1
11	10	+1	1
12	10	+2	1

The CM_APPLY applies the CM_*XXX*_STATEs to incoming frames. The application follows the following pseudo-code. The mirrors are added to the DMASKs if mirrors are requested.

```
# Add mirrors to destination mask
dmask1 = dmask
if (hasMirror0) dmask1 |= shift(1,MirrorDest0)
if (hasMirror1) dmask1 |= shift(1,MirrorDest1)
```

The current states are recovered for the frame received.

```
# port = receiving port for this frame
# swpri = switch priority assigned to frame
tc = CM_APPLY_SWITCH_PRI_TO_TC[swpri]
smp = CM_APPLY_TC_TO_SMP[tc]

# Recover states
overGlobal = CM_APPLY_STATE.global_ex
overShared = CM_APPLY_STATE.rxs[swpri]
overRxPrivate = CM_APPLY_RX_SMP_STATE[port,smp].rxPrivate
overRxHog = CM_APPLY_RX_SMP_STATE[port,smp].rxHog
overSoftDrop = CM_APPLY_SOFTDROP_STATE[swpri].overSoftDrop
usageOverLimit = CM_APPLY_SOFTDROP_STATE[swpri].usageOverLimit
```

The frame is dropped for all destinations if it is over the allowed global space or the allowed maximum space for this port and this SMP, or if the shared area is exceeded and private is exceeded.

```
# Drop if conditions applied
if ( overGlobal || overRxHog || (overRxPrivate && overShared) )
{
   dmask1 = 0
}
```

If the frame is not dropped, the unit checks each possible TX port destination and verifies if the frame could be forwarded on that port. A frame cannot be forwarded to a particular port if the transmit queue is consuming too much memory. A (private and hog) watermark pair is defined per (Tx and SWPRI) queue, which controls this filtering.

As long as the queue's memory usage is below the private watermark, the frame is queued to the specified destination. If memory use is above the hog watermark, the frame is not queued. Between the private and hog watermarks, two soft drop evaluations are performed to determine one of three drop probabilities (0%, 50%, or 100%).

The pseudo code is as follows:

```
# Compute adjusted jitter
jitter = random(CM_APPLY_SOFTDROP_CFG[swpri].JitterBits)
softDropEnabled = overSoftDrop && (usageOverLimit >= jitter)

# Check for TX drop
foreach i (0..47)
{
   if ( dmask1[i] )
   {
      # Recover state
      overTxPrivate = CM_APPLY_TX_TC_STATE[port,tc].txPrivate
      overTxHog = CM_APPLY_TX_TC_STATE[port,tc].txHog
      overSmpFree = CM_APPLY_TX_SWPRI_STATE[port,swpri].overSmpFree
      overSmpFree2 = CM_APPLY_TX_SWPRI_STATE[port,swpri].overSmpFree2
      softDropOnPrivate = CM_APPLY_TX_SOFTDROP_CFG[tx,swpri].SoftDropOnPrivate
      softDropOnFree = CM_APPLY_TX_SOFTDROP_CFG[tx,swpri].SoftDropOnFree

      # Equations
      if ( softDropOnFree && softDropEnabled )
          overSoft = overSmpFree || (overSmpFree2 && random(1));
      else if ( softDropOnPrivate )
          overSoft = random(1)

      # Drop if applicable
      if ( overRxPrivate && (overTxHog || (overTxPrivate && overSoft)))
      {
          dmask1[i] = 0
          CM_APPLY_DROP_COUNT[port]++
      }
   }
}
```

If the frame is completely dropped due to TX drop, it is counted in the CMTxHog0Drops (SMP 0) or CMTxHog1Drops counters (SMP 1), regardless if the frame is dropped due to an over soft drop watermark excess or a transmit hog watermark excess.

The drop probability is calculated as follows:

When SoftDropOnPrivate=1

Regime	Drop Probability
txTcUsage >= txTcHog	100%
txTcUsage >= txTcPrivate && txTcUsage < txTcHog	50%
txTcUsage < txTcPrivate	0%

When SoftDropOnFree=1

Regime	Drop Probability
txTcUsage >= txTcHog	100%
txTcUsage >= txTcPrivate && txTcUsage >= smpFree	100% * jitterProb
txTcUsage >= txTcPrivate && txTcUsage >= smpFree/2 && txTcUsage < smpFree	50% * jitterProb
txTcUsage >= txTcPrivate && txTcUsage < smpFree/2	0%
txTcUsage < txTcPrivate	0%

Note: jitterProb = smpUsage > (softDropWM + random(JitterBits))

5.7.10.5 Congestion Management Register Changes

CM registers can be changed at any time without causing malfunction of this unit.

5.7.10.6 PAUSE Flow Control

The FM10000 supports two types of PAUSE frames:

- IEEE 802.3 PAUSE frames
  - — PAUSE frames which contains a single pause interval applicable to all traffic classes
- Priority Based PAUSE frames (802.1Qbb)
  - — PAUSE frames which contain a bit vector to identify which class to pause and a 16-bit pause time for each class paused.

5.7.10.6.1 PAUSE Generation

The FM10000 monitors the amount of data accumulated in the shared memory, and enters a pause ON state if the data accumulated exceeds programmable limits and pause is enabled. The switch exits the pause ON state when data goes below certain levels. The limits are programmable independently for going into pause state (*PauseOn* thresholds) or out of pause state (*PauseOff* thresholds). For the device to work properly, the software programs the *PauseOff* thresholds to a lower value than the *PauseOn* thresholds.

If programmed to do so, the FM10000 starts transmitting PAUSE(Quanta) frames as it enters into a pause ON state, and continues to transmit those frames periodically as long as the pause ON state persists. The FM10000 transmits a single PAUSE(0) frame if the pause ON state clears.

The repeat interval for pause frames is programmable.

The following registers control the pause transmission behavior:

- CM_RX_SMP_PAUSE_WM (48 x 2 x 32-bits) — Defines the limit in number of segments that each RX can used per SMP before starting or stopping sending PAUSE frames. The register contains an ON limit and an OFF limit. The hardware assumes that the ON limit is greater than the OFF limit.
- CM_SHARED_SMP_PAUSE_WM (2 x 32-bits) — Defines the limit in number of segments of the shared pool that can be used per SMP before starting or stopping PAUSE frames. The register contains an ON limit and an OFF limit. The hardware assumes that the ON limit is greater than the OFF limit.
- CM_SHARED_SMP_PAUSE_CFG[smp].*EnableMask[port]* — Indicates which Pause Class is activated for each SMP.
- CM_PC_SMP_MAP (48 x 8-bit) — Used for mapping pause class priority from traffic class during class-based pause transmission.
- CM_PAUSE_RESEND_INTERVAL (48 x 16-bits) — Defines the PAUSE resend interval in quanta for each port.
- MOD_PER_PORT_CFG_2[0..47] — Defines the pause type and pause quanta for each port.

5.7.10.6.2 PAUSE Reception

The FM10000 can be programmed to react to pause frames received. When a PAUSE frame is received on a port and this pause class is enabled, the FM10000 immediately updates the state of each pause class (pc) of that port according to the incoming frame. If it is an standard pause frame, all pause classes are stopped for the prescribed time, or immediately un-paused if the specified time is zero. If it is a class based (PFC) frame, only those pause classes marked in the PFC frame priority_enable_vector are updated, with each paused or un-paused according to whether that class's specified time is zero.

There is also a per-traffic-class mask available (CM_PAUSE_CFG.*PauseMask*) to disable pausing on specific traffic classes. The CM_TC_PC_MAP is used to map pause class to traffic class.

A pause class given a nonzero pause time counts down that time at a rate determined by the port speed, until the time reaches zero, at which point the class is un-paused.

The following registers control the pause reception behavior:

- CM_PAUSE_CFG[0..47].*PauseMask* — Defines whether a TC can be paused or not, and defines Quanta setting.
- CM_TC_PC_MAP[0..47] — Defines how Traffic Class are mapped to Pause Class.
- CM_PAUSE_BASE_FREQ — Refer to Section 5.7.10.6.2.1.

5.7.10.6.2.1 Pause Timer Rate Configuration

There are two steps to configuring the rate of each port's pause timers:

1. The frequency of the FC clock relative to the 156.25 MHz ETH_REFCLK is specified as an M/N ratio. The values used in the formula below are taken from the PLL_FABRIC_CTRL register:
    - CM_PAUSE_BASE_FREQ
        - M: FbDiv255 * (2*(FbDiv4+1))
        - N: RefDiv * OutDiv

    Note: Some SKUs of the FM10000 only support a limited set of FC frequencies, as specified by PLL_FABRIC_LOCK.*FeatureCode*. In this case the values in PLL_FABRIC_CTRL are not directly writable, and are forced to the correct values for that SKU. The above formulas for CM_PAUSE_BASE_FREQ can still be used.

2. The speed of each port should be configured in CM_PAUSE_CFG[port]. The fields *PauseQuantaMultiplier* and *PauseQuantaDivisor* define the duration of a 512-byte pause quantum, relative to a 1.28 ns base interval. The *PauseQuantaMultiplier* has a minimum value for correct operation that depends on the exact FC frequency. The following settings should be used for different port speeds:

Port Speed	PauseQuantaMultiplier	PauseQuantaDivisor
100 Gb/s	64	16
40 Gb/s	160	16
25 Gb/s	160	10
10 Gb/s	160	4
1 Gb/s	400	1
100 Mb/s	4000	1
10 Mb/s	40000	1

5.7.10.6.2.2 Pause Reaction Time and Precision

The FM10000 stops scheduling frames on a port at the time the pause frame is received and fully processed by the frame processing pipeline. However, this only controls the TX scheduling; committed data is still in the TX pipeline and is terminated before the pause takes effect. The committed data is equal to a frame size plus up to 1500 bytes.

The pause requested is affected two ways:

- Delayed due to frame packet processing (from 200 ns to 500 ns).
- Delayed due to already committed data (depend on packet size).

The actual time is:

```
Given folllowing definition:

  T = PAUSE arriving to FM10000
  T1 = PAUSE processing completed and TX scheduler stopped
  T2 = DATA stops at FM10000 output
  T3 = DATA restarted at FM10000 output

Then time relations are:

  T1 = T + [200..500ns] (variable, depends on actual load)
  T2 = T1 + [0..CommittedData/Speed] (variable, depends on committed data)
  T3 = T1 + Quanta*512/Speed (fix)
```

The actual pause time observed by the link partner is in the range T3-T2, which can be expressed in Quanta as [Q-Qc..Q], where Q is the quanta received and Qc is the amount of data already committed by the transmitter (in Quanta units).

```
For 40G/100G:   Qc = MaxFrameLength / 512 + 120
For all others: Qc = MaxFrameLength / 512 + 30

Example:

  MaxFrameLength=10K, SPEED=25G/10G/1G   ==> pause time in range [Q-190..Q]
  MaxFrameLength=1518B, SPEED=25G/10G/1G ==> pause time in range [Q-53..Q]
  MaxFrameLength=10K, SPEED=100G/40G     ==> pause time in range [Q-276..Q]
  MaxFrameLength=1518B, SPEED=100G/40G   ==> pause time in range [Q-143..Q]
```

As an example, if a link partner is requesting a pause of 1000 quanta, the maximum frame size is 1500 bytes, and the port is at 10 GbE, the pause time is actually somewhere in the range [48.5 µs..51.2 µs], depending of activity at that time, [947..1000] in Quanta.

5.7.11 Statistics and Monitoring

5.7.11.1 Frame Counters

The switch logic maintains numerous per-port and switch-wide frame counters that provide management with statistical information about the state of the switch and of the network in general. The majority of these counters are provided for compatibility with network monitoring-related IETF RFCs: 2819 (RMON), 3273 (High-Capacity RMON), 2613 (SMON), and 4502 (RMON2).

The switch counters are organized into a collection of counter groups. Most groups are defined such that only one of its counters is updated per ingress or egress frame. Table 5-28 summarizes the counter groups. When a frame is sent or received exactly one counter in each group is incremented, with the exception of group 4 & 5, which are considered as one group.

Table 5-28 List of Counter Banks

Group (Bank)	Name	Description	Type	Unit	Number of Counters	Standard
1	RX_STATS_BANK[0]	RX Type	Per-Port, RX	Byte & Frame	NumPorts x 6	RMON
2	RX_STATS_BANK[1]	RX Length Bin	Per-Port, RX	Byte & Frame	NumPorts x 16	RMON
3	RX_STATS_BANK[2]	RX Per Priority	Per-Port, RX	Byte & Frame	NumPorts x 8	SMON
4	RX_STATS_BANK[3]	RX Forwarding A	Per-Port, RX	Byte & Frame	NumPorts x 16	—
5	RX_STATS_BANK[4]	RX Forwarding B	Per-Port, RX	Byte & Frame	NumPorts x 16	—
6	RX_STATS_BANK[5]	RX VLAN	Per-Port, RX	Byte & Frame	64 x 3	SMON
7	MOD_STATS_BANK[0]	TX Type	Per-Port, TX	Byte & Frame	NumPorts x 7	RMON
8	MOD_STATS_BANK[1]	TX Length Bin	Per-Port, TX	Byte & Frame	NumPorts x 12	RMON
9	CM_APPLY_DROP_COUNT	TX CM Drops	Per-Port, TX	Frames	NumPorts x 1	RMON
10	TRIGGER_STATS	Trigger counters	Per-Trigger, RX	Frames	64	–

Note: A variety of layer 3/4 packet header and forwarding properties can be counted in a configurable way by using the FFU counters. Each FFU counter tallies both the packet count and octet count of each frame counted.

5.7.11.1.1 Ingress Counters

Bank	Type	Description
1	Per Port	IP/nonIP, L2 unicast/multicast/broadcast, pause, frame errors.
2	Per Port	Length binning, length errors.
3	Per Port	Priority, TC or SWITCH_PRI depending on STATS_CFG.PrioritySelect.
4	Per Port	Forwarding, drop classification.
5	Per Port	Drop classification.
6	Per VLAN counter index	VLAN statistics.

5.7.11.1.1.1 Bank 1: Frame Type Classification

In Frame type classification, the precedence is from lowest to highest. As an example, an IPv6 multicast frame with an FCS error is counted as an FCS error.

Bank	Index	Class	Description
1	0	Non IP, L2 unicast	
1	1	Non IP, L2 multicast	
1	2	Non IP, L2 broadcast	
1	3	IPv4, L2 unicast DMAC	
1	4	IPv4, L2 multicast DMAC	
1	5	IPv4, L2 broadcast DMAC	
1	6	IPv6, L2 unicast DMAC	
1	7	IPv6, L2 multicast DMAC	
1	8	IPv6, L2 broadcast DMAC	
1	9	IEEE 802.3 pause	
1	10	Class based pause	
1	11	Framing error	Any frame with an error other than FCS error.
1	12	FCS error	Frame with FCS Error but no other type of error.
1	13	Unused	
1	14	Unused	
1	15	Unused	

Frames received on Ethernet ports might be erroneous for various reasons; overrun, bad encoding, FCS errors, etc. The Ethernet Port Logic marks the frame as having an FCS error if and only if the frame has an FCS error but no other type of errors detected. Otherwise, the frame is counted as having a framing error.

The next table (Section 5.7.11.1.1.2) shows the length binning according to actual frame length received after adjustment as per RX_STATS_CFG.*PerFrameAdjustment*. Note that the EPL might be configured to enforce a maximum frame length, and if this happens, the actual binning is for the truncated length, not the actual frame length received by the EPL.

5.7.11.1.1.2 Bank 2: Frame Length Classification

Bank	Index	Class	Description
2	0	Len < 64	
2	1	Len = 64	
2	2	64 < len < 128	
2	3	128 ≤ len < 256	
2	4	256 ≤ len < 512	
2	5	512 ≤ len < 1024	
2	6	1024 ≤ len < 1523	
2	7	1523 ≤ len < 2048	
2	8	2048 ≤ len < 4096	
2	9	4096 ≤ len < 8192	
2	10	8192 ≤ len < 10240	
2	11	len ≥ 10240	
2	12	Unused	
2	13	Unused	
2	14	Unused	
2	15	Unused	

5.7.11.1.1.3 Bank 3: Frame Priority Classification

Bank	Index	Class	Description
3	0	Priority = 0 (TC or SWPRI, selected by STATS_CFG)	
3	1	Priority = 1 (TC or SWPRI, selected by STATS_CFG)	
3	2	Priority = 2 (TC or SWPRI, selected by STATS_CFG)	
3	3	Priority = 3 (TC or SWPRI, selected by STATS_CFG)	
3	4	Priority = 4 (TC or SWPRI, selected by STATS_CFG)	
3	5	Priority = 5 (TC or SWPRI, selected by STATS_CFG)	
3	6	Priority = 6 (TC or SWPRI, selected by STATS_CFG)	
3	7	Priority = 7 (TC or SWPRI, selected by STATS_CFG)	
3	8	Priority = 8 (0 or SWPRI, selected by STATS_CFG)	
3	9	Priority = 9 (0 or SWPRI, selected by STATS_CFG)	
3	10	Priority = 10 (0 or SWPRI, selected by STATS_CFG)	
3	11	Priority = 11 (0 or SWPRI, selected by STATS_CFG)	
3	12	Priority = 12 (0 or SWPRI, selected by STATS_CFG)	
3	13	Priority = 13 (0 or SWPRI, selected by STATS_CFG)	
3	14	Priority = 14 (0 or SWPRI, selected by STATS_CFG)	
3	15	Priority = 15 (0 or SWPRI, selected by STATS_CFG)	

5.7.11.1.1.4 Bank 4, 5: Action Code Classification

Banks 4 and 5 are combined to track the 5-bit action code which has more than 16 bins.

Bank	Index	Class	Description
4	0	FIDForwarded	Forwarded normally, unicast or multicast, valid MA_TABLE lookup, also L2 broadcast.
4	1	FloodForwarded	MA_TABLE miss
4	2	SpeciallyHandled	FTYPE==0x2 (special delivery)
4	3	ParserErrorDrops	Would be uncorrectable ecc error drops in the FM10000.
4	4	ParityErrorDrops	
4	5	Trapped	
4	6	PauseDrops	Dropped due to SYS_CFG_1.*dropPause* or SYS_CFG_1.*dropMacCtrlEthertype.*
4	7	STPDrops	Fully dropped by STP, multicast frames partially dropped not counted.
4	8	SecurityViolations	Security violation.
4	9	VlanTagDrops	
4	10	VlanIngressDrops	Ingress boundary violation.
4	11	VlanEgressDrops	
4	12	GlortMissDrops	
4	13	FFUDrops	
4	14	TriggerDrops	Frame dropped due to a trigger drop action or a trigger rate limiter action.
4	15	Reserved	
5	0	PolicerDrops	
5	1	TTLDrops	
5	2	CMPrivDrops	CM global watermark.
5	3	CMSMP0Drops	Insufficient memory in SMP 0.
5	4	CMSMP1Drops	Insufficient memory in SMP 1 *5.
5	5	CMRxHog0Drops	SMP0, incremented if frame is dropped due to an RX hog watermark.
5	6	CMRxHog1Drops	SMP1, incremented if frame is dropped due to an RX hog watermark.
5	7	CMTxHog0Drops	SMP0, incremented if frame is fully dropped due to a TX condition (hog or softdrop).
5	8	CMTxHog1Drops	SMP1, incremented if frame is fully dropped due to a TX condition (hog or softdrop).
5	9	Reserved	
5	10	TriggerRedirects	
5	11	FloodControlDrops	Dropped due to DLF flood control.
5	12	GlortForwarded	Forwarded using a DGLORT coming from FFU or ARP.
5	13	LoopbackSuppDrops	Dropped frame due to loopback suppress.
5	14	OtherDrops	Dropped frame due to one of the following: • Reserved multicast DMAC (01-80-C2-00-00-XX). • Invalid SMAC. • Trapped but CPU_TRAP_DMASK is 0. • Triggers forward but LAG filtering then cause a drop.
5	15	*Reserved*	

5.7.11.1.1.5 Bank 6: VLAN Classification

VLAN counters (not per port):

Bank	Index	Class	Description
6	0	INGRESS_VID_TABLE.*CounterIndex*==0, Ucast	
...			
6	63	INGRESS_VID_TABLE.*CounterIndex*==63, Ucast	
6	64	INGRESS_VID_TABLE.*CounterIndex*==0, Multicast	
...			
6	127	INGRESS_VID_TABLE.*CounterIndex*==63, Multicast	
6	128	INGRESS_VID_TABLE.*CounterIndex*==0, Broadcast	
...			
6	191	INGRESS_VID_TABLE.*CounterIndex*==63, Broadcast	

5.7.11.1.2 Egress Counters

Groups 7 and 8 are implemented in the modify. They increment one bin each per transmitted frame on the specified port, and can be enabled/disabled for each port. For each port, minimum frame size can be configured to either 0 bytes or 64 bytes. If the minimum frame size is 64 bytes, a frame less than 64 bytes is padded to 64 bytes. When multiple error conditions pertain, the hardware counts the Group 7 event with the highest priority (1).

Per-port frame and byte counters:

Group (Bank)	Index	Priority	Description
7	0	7	L2 unicast, good FCS
7	1	8	L2 broadcast, good FCS
7	2	9	L2 multicast, good FCS
7	3	6	Tx error sent: frames transmitted with bad FCS.
7	4	3	Timeout drop: dropped due to timeout in switch memory.
7	5	2	Tx error drop: dropped due to Tx error (SAF error from FC).
7	6	1	Tx ecc drop: dropped due to uncorrectable error (incl scheduler memories).
7	7	4	Loopback drop: dropped due to MCAST loopback suppression (in the scheduler).
7	8	5	TTL1 drop: dropped due to routed frame having TTL ≤ 1.
7	9	10	Pause: PAUSE frame sent.
7	10	11	CB-pause: Class-Based (PFC) PAUSE frame sent.
7	11		Unused.
7	12		Unused.
7	13		Unused.
7	14		Unused.
7	15		Unused.

Group (Bank)	Index	Description
8	0	Len < 64
8	1	Len = 64
8	2	64 < len < 128
8	3	128 <= len < 256
8	4	256 <= len < 512
8	5	512 <= len < 1024
8	6	1024 <= len < 1523
8	7	1523 <= len < 2048
8	8	2048 <= len < 4096
8	8	4096 <= len < 8192
8	10	8192 <= len < 10240
8	11	10240 <= len
8	12	Unused.
8	13	Unused.
8	14	Unused.
8	15	Unused.

5.7.11.2 Frame Monitoring

The FM10000 contains two mirrors, commonly one is used for programmed mirrors while the other is used for CPU logging. Each mirror has a single configured monitor port where a single RX copy (identical to how the frame was received by the switch) is sent if a flow triggers a mirror. If a regular frame is also going out the monitor port, the mirrored copy is generated in addition to the regular frame copy (or copies).

There are three ways for a flow to cause a mirror copy to be generated:

- The FFU hits with a mirror action, in which case mirror 1 is used.
- A logging action is invoked (FFU log action, ICMP, TTL<=1, ARP redirect, IEEE Reserved MAC action), in which case mirror 0 is used.
- The trigger unit specifies a mirror action, in which case either mirror 0 or 1 can be specified by the trigger.

Note: The configurable match conditions of the FFU and a trigger can be combined to determine whether a frame is mirrored (the FFU must specify a trigId and the trigger must match on the FFU's trigId).

The two mirrors share a set of 64 profiles. Each profile specifies:

- The monitor port.
- Whether to truncate the mirror copy; the maximum frame size is set in the MODIFY unit, per-port (MirrorTruncationLen).
- DGLORT to place into the FTAG (if present) of the mirror copy.

- VID and VPRI to optionally add to the mirror copy in a new VLAN tag using the monitor port's configured Ethertype (the new tag is added on top of any existing VLAN tag in the frame) and to overwrite the corresponding values in the FTAG (if present) of the mirror copy.

The profile is selected either by the trigger configuration, or in the case of FFU mirrored frames or logging action frames, by a register dedicated for that purpose.

Each mirror can generate one copy, and these copies are completely independent from the regular egress traffic. If both mirrors send copies to the same egress port, that port gets two independent copies, each potentially with its own modification profile. If at the same time regular traffic from the same ingress frame is sent out that port, the regular traffic and the mirror traffic are sent together.

When frames that are to be mirrored are also congestion managed, they are treated the same way as multicast- the mirrored copy is treated as at most one Tx copy in the CM tracking. If the mirror copy goes out the same port as the regularly forwarded traffic, it is not counted for congestion management, a similar case to how IP multicast copies on the same port are counted.

Further modification of the copied frames by downstream FTAG tagged devices is prevented by setting the mirror copy's FTYPE field to 0x2 (Special Delivery) on egress.

Mirrors and traps are very similar- however traps use the regular forwarding mechanism (setting the forwarding mask to the CPU_MASK), while the mirroring logic creates additional frame copies:

- **TRAP** — Sets the dmask = CPU_TRAP_MASK, CPU_CODE set depending on trap reason or trigger number.
- **LOG** — Uses Mirror0, sets CPU_CODE equal to mirror profile or trigger number.

If you TRAP and LOG, two copies of the same frame are generated (identical behavior from previous devices, like the FM4000). The CPU_CODE/DGLORT may differentiate between the two copies, if these two copies end up going out the same port.

5.8 Egress Modifications

5.8.1 Overview

The egress modifications are performed by the MODIFY unit. The relationship of this unit to the surrounding units is shown in Figure 5-33.

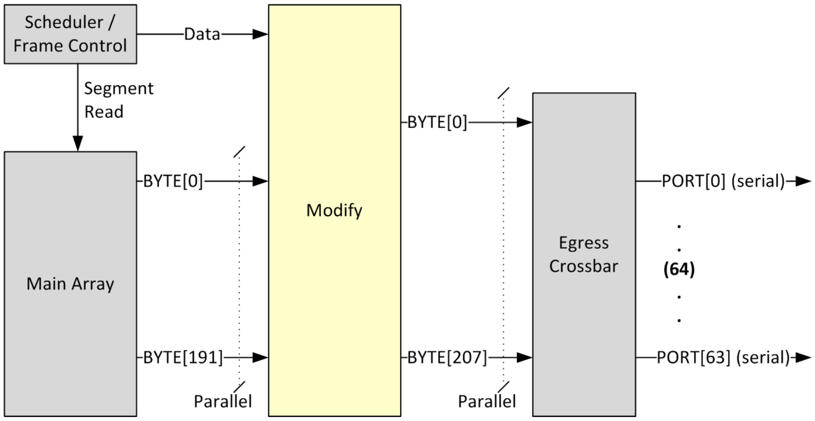


Figure 5-33 Egress Modification Overview

The scheduler is responsible for scheduling frames, one segment at a time.

Maximum size of each segment is 192 bytes. Segments are read from main memory and supplied to the MODIFY unit for modification. Only the first segment is truly modified; all successive segments are passed through or deleted (in case of truncation). The final size of the segment after modification is sent to statistics.

The data supplied depends on the type of segment. For the first segment of the frame, the data includes the following information.

- Port
- New DMAC
  - Used if the frame is routed.
- New DGLORT, SGLORT, SWPRI, USER
  - Used to create an FTAG if required.
- New VPRI, VID1
  - Used to create an FTAG if required, or update VLAN tags on egress.
- New IP TOS (DSCP)
- Tail information
  - Not last segment.
  - Last segment, good frame.
  - Last segment, bad frame.

- Flags
  - Packet type (normal or special).
  - Time tag request.
  - Pause frame.
- Two-bit tagging request from FC (TXTAG)
  - Apply normal VLAN tagging.
  - Force adding VLAN.
  - Force removing VLAN.
  - Force adding/replacing VLAN.
- Pause vector
  - If the frame is a pause frame, Indicates which traffic classes are marked pause ON.
- Pointer to MCAST table
  - Only used if lookup in MCAST table required.
- Mirror command and mirror indexes.
- multicast pointer.

For the non-first segments, the data includes:

- Port
- Tail information
  - Not last segment.
  - Last segment, good frame.
  - Last segment, bad frame.

The MODIFY unit details are shown in Figure 5-34.

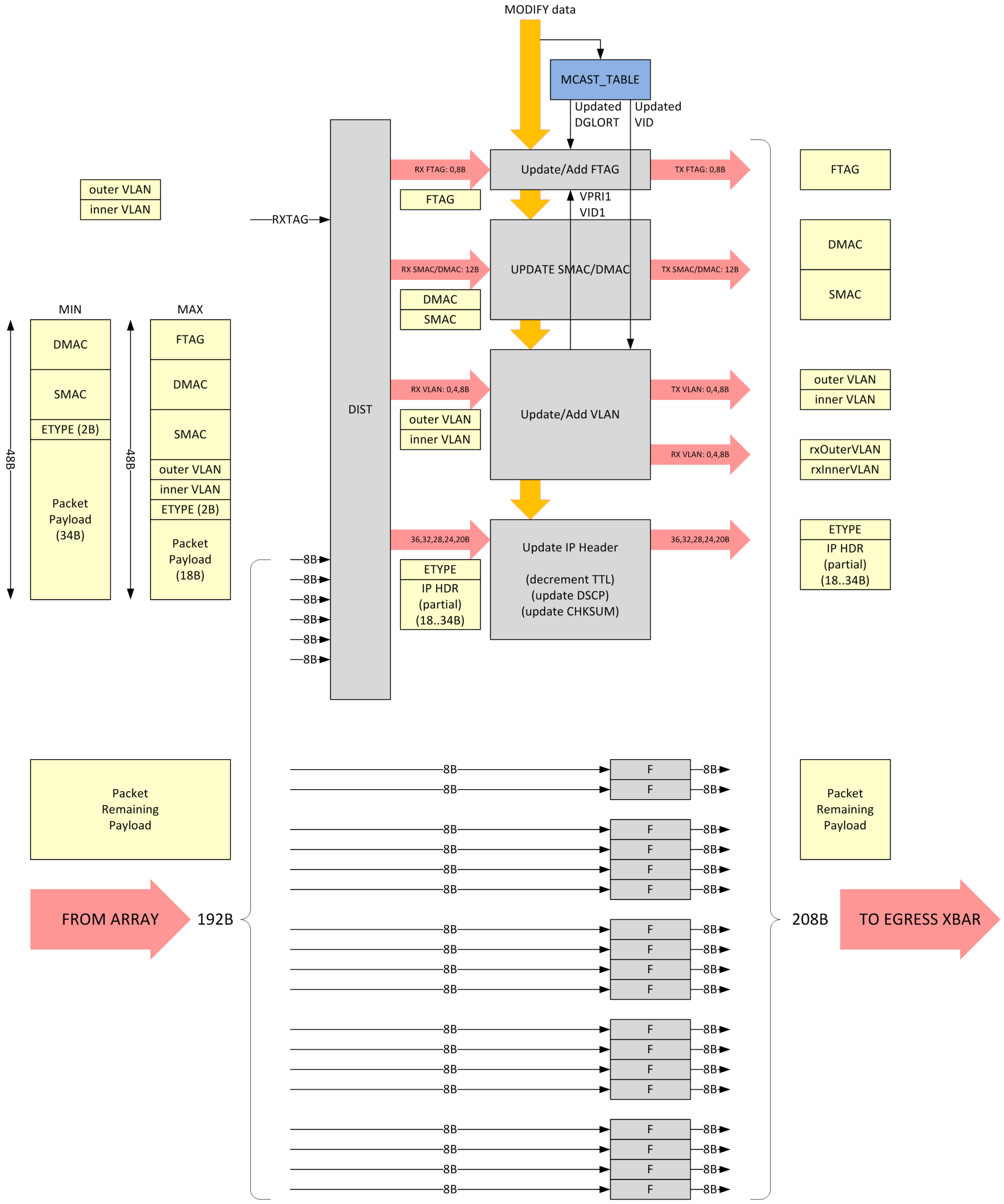


Figure 5-34 MODIFY Unit Block Diagram

The MODIFY unit has a 192-byte interface to main memory and a 208-byte interface to the egress crossbar, all split into 8-byte channels, each channel having a 4-bit length field indicating the number of bytes enabled on this channel. The MODIFY unit can add up to 16 bytes of new data on a given frame.

The MODIFY unit uses information from the parser (RX TAG) to determine the length of the header it needs to process. The header, from a modify perspective, consists of FTAG, DMAC, SMAC, two VLANs, ETYPE, and at least 18 bytes from IP header. (IP headers are longer than 18 bytes, but the MODIFY unit does not need further information past 14 bytes. The extra 4 bytes are used to align to 8-byte channel multiple.)

The RX tag encoding reflects what the parser has parsed for layer 2 (refer to Section 5.7.2):

- Bit 0: FTAG present
- Bit 1: VLAN1 tag present
- Bit 2: VLAN2 tag present
- Bit 3: VLAN2 tag before VLAN1 tag
- Bit 4: IPV4 header present
- Bit 5: IPV6 header present
- Bit 6: MPLS present
- Bit 7: Custom tag present

The RX Tag is used to re-align the first 48 bytes of the segment such that the FTAG, DMAC/SMAC, VLANs and IP are aligned with their respective specialized units.

Note: Routing modifications and DSCP updates, with the exception of VLAN tag updates (that is, any changes to DMAC, SMAC, TTL, or DSCP) are only supported on frames with no MPLS or CUSTOM tags, and for which TXTAG==0. Refer to Section 5.8.4.2 for TXTAG definitions.

5.8.2 FTAG Generation or Modification

The FM10000 supports adding, removing or updating the FTAG. The presence of the FTAG on egress depends on a per-egress port configuration in the modify. If set, all frames going to the egress port have an FTAG.

MODIFY always uses a 64-bit FTAG for enabled ports regardless of the type of ports targeted (Ethernet or others). The format is defined in Section 2.3.

Note: For Ethernet ports, the FTAG is usually transmitted in the preamble, which has a useful payload of only 56 bits. The USER field is sacrificed in this case and not transmitted on the interface.

The VLAN field in the FTAG is always VPRI1/VID1. The section on VLAN modifications details how the unit

derives the VPRI1/VID1.

The FTAG contains:

- **User** (8 bits) — Defined by the frame handler (settable via ACL). The USER bit0 gets updated to reflect if the ingress VID (IVID) is equal to egress VID (EVIDa), after MCAST substitution.

  0b = Not-equal)
  1b = Equal

- **FTYPE** (2 bits):

  00b = NORMAL — Normal delivery. frame modifications applied.
  10b = SPECIAL — Special delivery. Cancel frame modifications.

- **VTYPE** (2 bits) — Not used, set to 0b.
- **SWPRI** (4 bits) — Defined by the frame handler.
- **VPRI/VID1** (16 bits) — Equal to VID1/VPRI1 after VLAN updates.
- **SGLORT** (16 bits) — Defined by the frame handler.

- **DGLORT** (16 bits) — Defined by the frame handler, might potentially be updated via MCAST_TABLE.

Except for SPECIAL packets (refer to Section 5.8.3), VLAN tagging manipulation is independent of FTAG setting (on or off). In other words, the ability to add/remove/update a VLAN1/VLAN2 is not affected by the presence of an FTAG.

It is expected that software does not normally request VLAN1 tag to be present on FTAG-ed ports on interconnecting switches operating in a multi-chip environment, since this information is already encoded in the FTAG. However, this is not required. The MODIFY unit may be potentially configured by software to enable FTAG and enable VLAN1 tag presence on a given port. This is potentially used on PCIe ports.

5.8.3 Special Packets

A frame may be designated as SPECIAL either by the frame processing pipeline (for example, if it is trapped or received as special), or because it is mirrored. The handling is as follows:

- Packet designated SPECIAL from frame processing pipeline:
  - If the egress port is FTAG-ed, update/add FTAG normally (FTAG.VID/VPRI is derived from EVID as regular for normal frames, except that =DGLORT/SGLORT/SWPRI/USER are updated, and FTYPE is set SPECIAL), but the rest of the packet payload is exactly as it was received.
  - If the egress port not is FTAG-ed, use MOD_VLAN_TAG_VID1_MAP[EVIDb].*Tag*[port] to determine if a packet is VLAN tagged or not. If yes, add a VLAN1 tag, no further changes to the packet (refer to Section 5.8.4.2 section below for definition of EVIDb).
- Packet has a mirror attached to it:
  - If the egress port is FTAG-ed, construct FTAG normally (FTAG.VID/VPRI is from MIRROR_PROFILE, DGLORT/SGLORT/SWPRI/USER updated, and FTYPE set to SPECIAL), but the rest of the packet payload is exactly as it was received.
  - If the port is not FTAG, use MIRROR_PROFILE.VID to determine if adding a new VLAN1 TAG is needed or not. Otherwise all further modifications are canceled.

Note: In all cases, unmodified packet payload means up to but excluding CRC. The EPL updates the CRC to reflect if the packet is a good packet or not.

5.8.4 L2 Modification

5.8.4.1 SMAC/DMAC Updates

The FM10000 supports the following L2 SMAC/DMAC frame header modifications:

- Update DMAC with DMAC received from scheduler if the frame is to be routed.
- Update SMAC with SMAC mapped from the 4-bit *RouterId* associated with the frame if the frame is to be routed.

The modifications are canceled if the Egress VID after MCAST update (EVIDa) is equal to Ingress VID for multicast frame. For details, refer to Section 5.8.4.4, “Loopback Suppress and Route Flag Update”. The modifications are also canceled if the frame contains MPLS or CUSTOM tags, or if TXTAG!=0.

Updating of DMAC and SMAC can be disabled independently via the following settings, respectively:

- MOD_PER_PORT_CFG_2[port].*EnableDMACRouting*
- MOD_PER_PORT_CFG_2[port].*EnableSMACRouting*

5.8.4.2 VLAN Updates

The VLAN transformation is more complex and serves multiple possible applications. The FM10000 supports two VLAN tags (VLAN1/VLAN2), ordering is arbitrary on ingress and egress. Each VLAN tag has a format compatible with 802.1Q, shown in Figure 5-35.

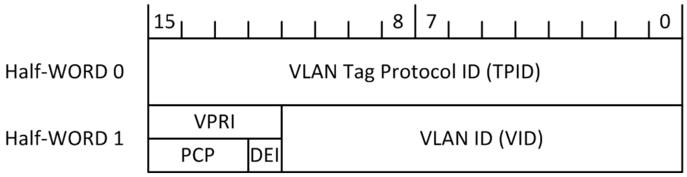


Figure 5-35 VLAN Tag Format

The VLAN tag fields are:

- VLAN Tag Protocol Identification (16 bits) —Also named VLAN Type.
- VLAN Identification (12 bits) — Also named VID.
- VLAN Priority (4 bits) — Also named VPRI. Contains the following fields:
  - — Bit 0 = Discard Eligible Indication (DEI) or Canonical Format Indicator (CFI) in 802.1Q-2008).
  - — Bit 1..3 = Priority Code Point (PCP).

The PCP and DEI/CFI are normally handled as a single 4-bit VPRI through the FM10000 frame processing pipeline, the egress modification module has an extra feature to manage the PCP and DEI/ CFI updates independently.

The VLAN manipulation is shown in Figure 5-36.

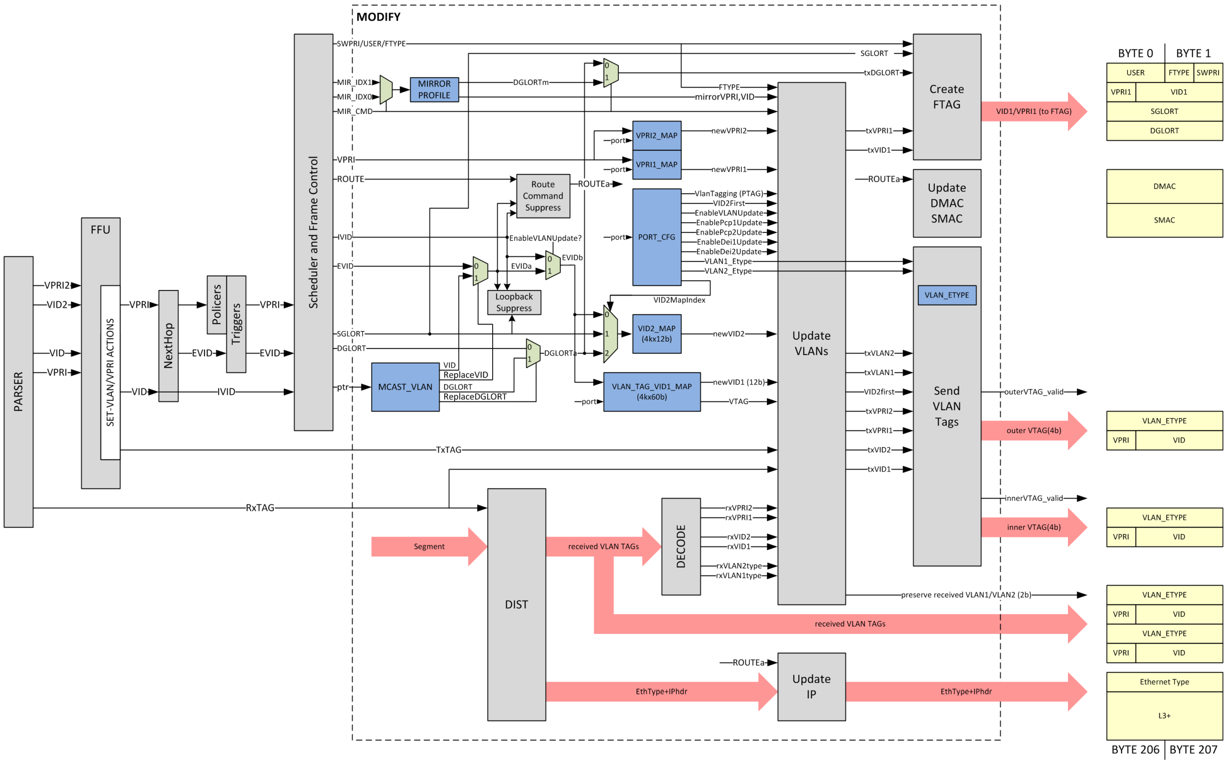


Figure 5-36 VLAN Egress Modifications

The VLAN modification block uses the following information:

- RXTAG (8 bits) — Parser information about VLAN tags and FTAGs presence.
- TXTAG (2 bits) — Tag command coming from FFU.
- VTAG (1 bit) — Port tagging configuration (MOD_VLAN_TAG_VID1_MAP[vid].*Tag*[port]).
- PTAG (3 bits) — Port tagging configuration (MOD_PER_PORT_CFG_2[port].*VlanTagging*).
- *EnableVlanUpdate* (1 bit) — Indicates if VLAN updated when routing (from MOD_PER_PORT_CFG_1).
- *EnablePcp1Update* (1 bit) — Indicates if PCP1 updated or not (from MOD_PER_PORT_CFG_2).
- *EnableDei1Update* (1 bit) — Indicates if DEI1 updated or not (from MOD_PER_PORT_CFG_2).
- *EnablePcp2Update* (1 bit) — Indicates if PCP2 updated or not (from MOD_PER_PORT_CFG_2).
- *EnableDei2Update* (1 bit) — Indicates if DEI2 updated or not (from MOD_PER_PORT_CFG_2).
- *Vid2First* (1 bit) — Egress VLAN1/VLAN2 ordering (from MOD_PER_PORT_CFG_2).
- rxVPRI1/rxVID1 — The ingress VPRI1/VID1 from received packet (if present).
- rxVPRI2/rxVID2 — The ingress VPRI2/VID2 from received packet (if present).
- txVPRI1/txVID1 — The egress VPRI1/VID2 for egress packet.
- txVLAN1 — Indicates if the VLAN1 tag shall be sent or not.
- txVPRI2/txVID2 — The egress VPRI1/VID2 for egress packet.

- txVLAN2 — Indicates if the VLAN2 tag shall be sent or not.
- newVPRI1/newVID1 — The new value for VPRI1/VID1 requested.
- newVPRI2/newVID2 — The new value for VPRI2/VID2 requested.
- EVID — Generic egress VLAN ID from Frame Processing Pipeline.
- IVID — Generic ingress VLAN ID from Frame Processing Pipeline.
- VLAN_Etype1 — Select which tag (from MOD_PER_PORT_CFG_2).
- MIR_CMD — From processing pipeline.
- MIR_IDX0 — Mirror index 0.
- MIR_IDX1 — Mirror index 1.
- mirrorVPRI/mirrorVID — From MOD_MIRROR_PROFILE_TABLE.
- FTYPE — The frame type from frame processing pipeline.
- ROUTE — Indicates if the frame is to be routed or not.
- MCAST_PTR — Indicates replication entry to use (0 = no replication).

The following tables are used during VLAN updates:

- MOD_PER_PORT_CFG_{1,2} — Defines attributes per port.
- MOD_MCAST_VLAN_TABLE — Defines a replacement for EVID or DGLORT, if desired.
  - — The scheduler indicates whether this frame is part of a replication list using a pointer into MOD_MCAST_VLAN_TABLE. If the pointer is non-zero, this frame is part of a replication list, and the table provides an updated DGLORT or EVID or both. MCAST_VLAN_TABLE is not used for mirror frames.
- MOD_VID2_MAP — Provides new VID2 from index (EVIDb, DGLORT[11:0] or SGLORT[11:0]).
  - — This table provides an updated VID2 from various fields. The index source selection is configurable per port.
- MOD_VLAN_TAG_VID1_MAP — Provides a 1-bit per-port tagging control (48-bit x 4096) and a new VID1 from egress VID.
- MOD_VPRI1_MAP — Provides a new VPRI1 from VPRI received from pipeline.
- MOD_VPRI2_MAP — Provides a new VPRI2 from VPRI received from pipeline.

The VLAN modification proceeds in three steps:

1. Retrieve current state, and apply basic changes and configuration.
2. Update VLAN fields (VPRI1,VID1,VPRI2,VID2).
   a. Non-mirrored and non-special frames.
   b. Mirrored frames.
   c. Special frames.
3. Construct VLAN tags.

Step 1: Retrieve current state and configuration

The RXTAG is used to recover the received VID1 and VID2 (rxVid1, rxVid2) from the received packet VLAN Tags. Those VLAN tags, if present, are also copied out, as they are possibly left as is at transmission time.

The multicast pointer from scheduler is used to index the MCAST_VLAN table which possibly requests an update to the DGLORT and/or the EVID. The entry 0 is reserved to indicate no change (*ReplaceVID*=0, *ReplaceDGLORT*=0). The updated EVID (or unchanged EVID), EVIDa, is then forwarded to loopback suppress unit and to another mux. For mirror frame, the multicast pointer is undefined and the MCAST_VLAN table indexing is skipped; the DGLORT and EVID are instead updated using the MOD_MIRROR_PROFILE. See “Step 2b: Mirrored Frames” below for further details on mirror frames.

A second EVID mux replaces the modified egress VID (EVIDa) with the ingress VID (IVID) if EnableUpdateVLAN is 0. The output of this second VID mux (EVIDb) is then used to index the MOD_VLAN_TAG_VID1_MAP to recover the VTAG and the new VID1 and also one of the choices to index VID2_MAP. The updated DGLORT is also one of the choices to index VID2_MAP.

MODIFY recovers a few configuration bits from MOD_PER_PORT_CFG_{1,2} and the *VID2MapIndex* which defines how to index the MOD_VID2_MAP. The MOD_VID2_MAP is used to recover a new VID2. There are three possible index selections:

0 = Use EVID
1 = Use SGLORT{0..11}
2 = Use DGLORT{0..11}

The purpose of the selection detailed later.

MODIFY recovers a new VPRI1/VPRI2 from VPRI provided by the frame processing pipeline. The application of VPRI1/VPRI2 is controlled by the port setting.

Step 2a: Update VLAN Fields (non-mirrored and non-special frames)

There are four operations supported per VLAN tag.

- UPDATE-or-ADD:
  - Update VLAN tag if present. Add if absent. Set txVLAN to 1b.
- DELETE:
  - Delete VLAN tag if present. Set txVLAN to 0b.
- INSERT:
  - Add new VLAN tags regardless of whether there are VLAN tags present or not on receive. The received VLAN Tags in the original packet are preserved and pushed further into the frame with the entire VLAN tag received unchanged. Set txVLAN to 1b.
- LEAVE-AS-RX:
  - If VLAN tag is present on receive, leave rxVID unchanged, but update VLAN Tag type and VPRI (for VPRI, follows the rule in Section 5.8.4.3; the PCP and DEI are only optionally changed depending on port configuration). If rxVID is zero, the tag is removed (step 3 below). Set txVLAN set to 1b.
  - If VLAN tag is absent on receive, do not add. Set txVLAN to 0b.

Table 5-29 shows the behavior for non-mirrored frames and normal frames.

Table 5-29 VLAN Updating Rules (not mirrored frames)

TXTAG	PTAG	CTAG	VID1	VID2	Application
0	0	0	DELETE	LEAVE-AS-RX	Basic 802.1Q
0	0	1	UPDATE-or-ADD	LEAVE-AS-RX	Basic 802.1Q
0	1	0	DELETE	LEAVE-AS-RX	Customer port in 802.1ad
0	1	1	DELETE	UPDATE-or-ADD	Customer port in 802.1ad
0	2	0	UPDATE-or-ADD	LEAVE-AS-RX	Provider port in 802.1ad
0	2	1	UPDATE-or-ADD	UPDATE-or-ADD	Provider port in 802.1ad
0	3	0	DELETE	UPDATE-or-ADD	Pseudo Port
0	3	1	UPDATE-or-ADD	UPDATE-or-ADD	Pseudo Port
0	4	0	DELETE	DELETE	Pseudo Port
0	4	1	UPDATE-or-ADD	DELETE	Pseudo Port
1	x	x	INSERT	INSERT	OpenFlow: Add VLAN
2	x	x	DELETE	DELETE	OpenFlow: Delete VLAN
3	x	x	UPDATE-or-ADD	UPDATE-or-ADD	OpenFlow: Replace VLAN

Step 2b: Mirrored Frames

The scheduler provides a mirror command (with possible values “do-not-mirror”, “mirror-0”, “mirror-1”), while the frame processing pipeline provides mirror index 0 and index 1 information. MODIFY selects between index 0 and 1 based on the mirror command provided by the scheduler, and looks up a mirror profile in MOD_MIRROR_PROFILE_TABLE. The mirror profile indicates how the mirror frame is to be constructed, it includes a mirror VPRI, a mirror VID, and a mirror DGLORT.

If the egress port is FTAG-ed, the FTAG DGLORT/VPRI/VID are set to the values recovered from the mirror profile. The FTYPE is set to SPECIAL. Rest of the frame must remained unchanged.

If the egress port is not FTAG-ed, this mirror VID is checked to be non-zero. If the mirror VID is zero, the packet is sent exactly as it was received. If the mirror VID is not zero, then a VLAN1 tag is added with VPRI/VID set to the values provided by the mirror profile. Except for the addition of a VLAN tag, the rest of the frame remains unchanged.

Step 2c: Special Frames

Special frames rules are defined in Section 5.8.2.

Step 3: Construct VLAN Tags

The VLAN tags are constructed and transmitted only if corresponding VID is non-zero and txVLAN is 1. The egress VLAN tag is constructed as follows:

- VLAN Tag 1
  - VLAN Tag Protocol ID (type) = MOD_VLAN_ETYPE[VLAN1_Etype]
  - VLAN ID = txVID1
  - VLAN VPRI = 010: txVPRI1
- VLAN Tag 2
  - VLAN Tag Protocol ID (type) = MOD_VLAN_ETYPE[VLAN2_Etype]
  - VLAN ID = txVID2

— VLAN VPRI = txVPRI2

The final stage is ordering. If the Vid2First is 1, VLAN Tag2 is sent first (outer), then VLAN Tag 1 (inner). If Vlan2First is 0, VLAN Tag 1 is sent first (outer), then VLAN Tag 2 (inner).

5.8.4.3 VPRI Updates

The VPRI coming from the frame pipeline is used to recover a new VPRI1 and new VPRI2. The table MOD_VPRI1_MAP[port].*VPRI*[vpri] is used to retrieve VPRI1 and the table MOD_VPRI2_MAP[port].*VPRI*[vpri] is used to retrieve VPRI2.

The *EnablePcp1Update*, *EnablePcp2Update*, *EnableDei1Update*, *EnableDei2Update* flags controls which part of these new fields are potentially applied. If a flag is set to 0b for corresponding VLAN tag and that VLAN tag was present on the ingress frame and the operation was not INSERT, MODIFY uses the corresponding field from the ingress frame. Otherwise it uses the values derived from the per-port VPRI maps.

```
vpri1_map = MOD_VPRI1_MAP[port].VPRI[vpri]
vpri2_map = MOD_VPRI2_MAP[port].VPRI[vpri]
enablePcp1Update = MOD_PER_PORT_CFG_2[port].EnablePcp1Update
enablePcp2Update = MOD_PER_PORT_CFG_2[port].EnablePcp2Update
enableDei1Update = MOD_PER_PORT_CFG_2[port].EnableDei1Update
enableDei2Update = MOD_PER_PORT_CFG_2[port].EnableDei2Update

if ( rxVlanTag1Present && op1 != INSERT && enablePcp1Update == 0 )
    txVpri1.pcp = rxVpri1.pcp;
else if ( rxFTagPresent && enablePcp1Update == 0 )
    txVpri1.pcp = rxFTAG.pcp;
else
    txVpri1.pcp = vpri1_map.pcp;

if ( rxVlanTag1Present && op1 != INSERT && enableDei1Update == 0 )
    txVpri1.dei = rxVpri1.dei;
else if ( rxFTagPresent && enableDei1Update == 0 )
    txVpri1.dei = rxFTAG.dei;
else
    txVpri1.dei = vpri1_map.dei;

if ( rxVlanTag2Present && op2 != INSERT && enableDei2Update == 0 )
    txVpri2.dei = rxVpri2.dei;
else
    txVpri2.dei = vpri2_map.dei;

if ( rxVlanTag2Present && op2 != INSERT && enableDei2Update == 0 )
    txVpri2.dei = rxVpri2.dei;
else
    txVpri2.dei = vpri2_map.dei;
```

5.8.4.4 Loopback Suppress and Route Flag Update

The MODIFY unit also includes a generic multicast loopback suppress. This avoids transmitting a frame back to the port it came from for the same VLAN. The loopback suppress requires the following information:

- The 16-bit source GLORT attached to select which tag the frame.
- The canonical GLORT of the egress port.
- The ingress VID (IVID) and the new egress VID (VIDa, after MCAST_VLAN update).

The goal of loopback suppression is to compare the egress VID to the ingress VID supplied. If they match and frame is multicast, suppress sending the frame to the canonicalized port on which the frame was received.

Also, if the frame is multicast and the egress VID is the same as the ingress VID, the MODIFY unit cancels the routing by clearing the route flag.

The expressions are:

```
glort-mask = MOD_PER_PORT_CFG_1[egress_port].LoopbackSuppressMask; csglortX =
MOD_PER_PORT_CFG_1[egress_port].LoopbackSuppressGlort; csglortF = SGLORT & glort-mask;
if ( MOD_MCAST_VLAN_TABLE[ptr].ReplaceVID )
   // Compare to new VID from MCAST_VLAN_TABLE
   EVIDa = MOD_MCAST_VLAN_TABLE[ptr].VID
else
   // Compare to VID from scheduler.
   EVIDa = EVID
# Drop packet for multicast if IVID=EVIDa and same port
if ( ptr!= 0)
{
  // L2 switching is when egress VID equals ingress VID
vlanSwitched = EVIDa == IVID;
  // Cancel routing for multicast frames and L2 switching
if ( vlanSwitched ) route = 0;
  // Drop if same port and L2 switching
drop = (csglortX == csglortF) & vlanSwitched;
  // Report l2 switching to connected device
FTAG.USER[0] = vlanSwitched;
}
 else
 {
  FTAG.USER[0] = 0
  drop = 0
}
```

5.8.4.5 VLAN Update Applications

The VLAN update mechanism enables the following applications to be supported:

- 802.1Q (standard VLAN)
- 802.1ad (provider bridging)
- 802.1bg (EVB)
- OpenFlow
- Pseudo-port Tagging

801.1Q - Virtual LAN:

A single VLAN tag is present (VLAN1).

- VID1_MAP[VID] = VID1
- VID2_MAP[VID] = unused (should not normally be present)
- PTAG = 0
- VTAG controls presence of VLAN 1 Tag
- VLAN 2 Tag always deleted

802.1ad - Provider Bridging:

Also, frequently referred as Q-in-Q. There are two modes of operation; stacked and mapped.

- Stacked Customer VLANs (VID=outer, VID2=inner)
  - FFU rule maps {CustPort} to VID, TXTAG=00
  - VID1_MAP[VID] = SVID(VID1)
  - VID2_MAP[VID] = 0
  - Provider Port PTAG = 2 (VTAG=0)
  - Customer Port PTAG = 1 (VTAG=0)
- Mapped Customer VLANS (VID=outer, VID2=inner)
  - FFU rule maps {CustPort,VID} to VID, TXTAG=00
  - VID1_MAP[VID] = CVID(VID2)
  - VID2_MAP[VID] = SVID(VID1)
  - Provider Port PTAG = 2 (VTAG=1)
  - Customer Port PTAG = 1 (VTAG=1)

OpenFlow:

- Add VLAN
  - FFU rule maps tuple to VID, TXTAG=01b.
  - VID1_MAP[VID] = VID1 (set to 0b to not add VID1)
  - VID2_MAP[VID] = VID2 (set to 0b to not add VID2)
- Delete VLAN
  - FFU rule maps tuple to VID, TXTAG=10b.

- — Replace VLAN
  - — FFU rule maps tuple to VID, TXTAG=11b.
  - — VID1_MAP[VID] = VID1
  - — VID2_MAP[VID] = VID2

802.1bg (EVB):

- Learning
  - — Unknown SMAC trapped to local CPU.
  - — CPU maps MAC table to DGLORT.
- EVB ports set to VID2_MAP indexed by DGLORT, PTAG = 3
- non-EVB ports set to PTAG = 0

Pseudo-Port Tagging:

- Export Source Information
  - — Transmit port VID2 map index set to SGLORT.
  - — Transmit PTAG set to 3.
- Export Destination information
  - — Transmit port VID2 map index set to DGLORT.
  - — Transmit PTAG set to 3.

A per-port configuration may disable the following for this port:

- DMAC Update: Do not do DMAC updates for routed frames on this port.
- SMAC Update: Do not do the SMAC updates for routed frames on this port.

5.8.5 L3 Modification

The FM10000 supports the following L3 frame header modifications:

- TTL Decrement the TTL on an IPv4 or IPv6 header and update the IPv4 checksum accordingly if the frame is routed.
  - — Cancel TTL decrement if the route flag gets cleared due to EVIDa (after MCAST update) equal to IVID.
  - — Cancel TTL decrement if TTL update is disabled for this port.
  - — Cancel TTL decrement if TXTAG different than 0.
  - — Cancel TTL decrement if MPLS or CUSTOM tags are present or if the total number of VLAN tags after modification is greater than 2.
- DSCP Update — Change the DSCP of the frame on an IPv4 or IPv6 header with the value sent from the pipeline and update the IPv4 checksum accordingly. This is applicable if the frame is routed or not.
  - — Cancel DSCP update if DSCP updates is disabled for this port.
  - — Cancel DSCP update if TXTAG different than 0.

- — Cancel DSCP update if MPLS or CUSTOM tags are present or if the total number of VLAN tags after modification is greater than 2.

5.8.6 Truncation

The egress modification unit supports truncation of mirrored frames. Truncation is determined by a single bit in the MOD_MIRROR_PROFILE_TABLE, which is looked-up, as explained in Section 5.8.6. The truncation length is defined on a per-port basis, in the MOD_PER_PORT_CFG_2 register. Frames can only be truncated to lengths between 64 bytes to 192 bytes, in increments of 4 bytes. Truncation on a port can be disabled by setting the truncated length to 63.

Truncated frames are sent with a good CRC, regardless of whether they were received with a good CRC or not, if no uncorrectable errors where detected while switching the frame. If truncation is disabled, the egress frame’s CRC follows the ingress frame’s CRC, or has bad CRC if uncorrectable errors where encountered while switching the frame.

5.8.7 Pause Generation

The Congestion Management sends a pause frame generation request through the Scheduler if the pause generation watermarks are exceeded. The request is presented to MODIFY as 1 bit for whether or not to generate a pause frame, and 8 bits of PAUSE_TC_XOFF pause information.

Each port is configured as either supporting classless or class-based pause via MOD_PER_PORT_CFG_2.*TxPauseType*. If a port is configured for classless pause, a PAUSE ON frame is generated if PAUSE_TC_XOFF is non-zero. Otherwise a PAUSE OFF frame is generated. If the port is configured for class-based pause, using the time expressions described below, the PAUSE_TC_XOFF vector is combined with MOD_PER_PORT_CFG_2.*TxPausePriEnVec* and MOD_PER_PORT_CFG_2.*TxPauseValue* to set the pause time values. MOD_PER_PORT_CFG_2.*TxPauseValue* allows the user to configure the pause timers per port.

- Classless PAUSE Frame:
    - — FTAG (Optional)
    - — DMAC = 01:80:c2:00:00:01
    - — SMAC = MOD_PAUSE_SMAC
    - — Ethertype = 0x8808
    - — Opcode = 0x0001 (PAUSE)
    - — Quanta = 0 for PAUSE OFF, MOD_PER_PORT_CFG_2.*TxPauseValue* for PAUSE ON
    - — Pad with zeros to 60 bytes (64-byte frame with CRC)
- Class-based PAUSE Frame:
    - — FTAG (Optional)
    - — DMAC = 01:80:c2:00:00:01
    - — SMAC = MOD_PAUSE_SMAC
    - — Ethertype = 0x8808
    - — Opcode = 0x0101 (Class based pause)
    - — priority Enable Vector = MOD_PER_PORT_CFG_2.*TxPausePriEnVec*

- Time[0] = zero or MOD_PER_PORT_CFG_2.*TxPauseValue*
- Time[1]
- Time[2]
- Time[3]
- Time[4]
- Time[5]
- Time[6]
- Time[7]
- Pad to 60 bytes (64-byte frame with CRC)

If MOD_PER_PORT_CFG_2.*TxPausePriEnVec*[tc] & PAUSE_TC_XOFF[tc] then Time[tc] is set to MOD_PER_PORT_CFG_2.*TxPauseValue*. Otherwise, Time[tc] is set to zero.

On FTAG-ed links, PAUSE frames are sent with an FTAG with all-zero fields.

5.8.8 Frame Sizes and Statistics

MODIFY is designed to handle good frames and bad frames. The minimum size for a good frame is 15 bytes, while the maximum size is 15 KB.

For bad frames, the minimum size is 4 bytes and the maximum size is 15 KB. For bad frames smaller than 64 bytes, the MODIFY unit might alter the bytes or frame size in an unpredictable manner while still forwarding the frame as bad.

The final frame size after modification is reported back to statistics for accounting and packet scheduler for traffic shaping. MODIFY includes an optional parameter per port to indicate if the port pads to 64 bytes or not. This is enabled on Ethernet ports so that the correct frame size is reported to statistics and frame scheduler.

The frame size reported does not include the FTAG.

6.0 PCIe Host Interface

6.1 Host Interface Overview

The FM10000 supports up to 9 PCIe End Points (PEPs). They are organized into two types of host interfaces:

- Four Dual PEP Host Interfaces, each supporting two modes of operation:
  - — Mode x8 — 1 x 8 GEN3 SerDes (64 Gb/s interface, 50 Gb/s packet throughput)
  - — Mode x4 — 2 x 4 GEN3 SerDes (32 Gb/s interface each, 25 Gb/s packet throughput each)
- One Single PEP Host Interface:
  - — 1 x 1 GEN3 SerDes (8 Gb/s interface, 5 Gb/s throughput)

All PEPs are completely independent of each other except that they share a common 100 MHz reference clock and a common 500 MHz clock. They can be reset independently of each other, and are seen as distinct interfaces from a host perspective. Each has the same feature set except for the number of PCIe SerDes and the speed of packet interface. Each host interface can be set into the single or dual PEP mode independently.

PEPs are numbered 0 through 8. When a Host Interface is set to operate in x8 mode, the second PEP in the same Host Interface is not used.

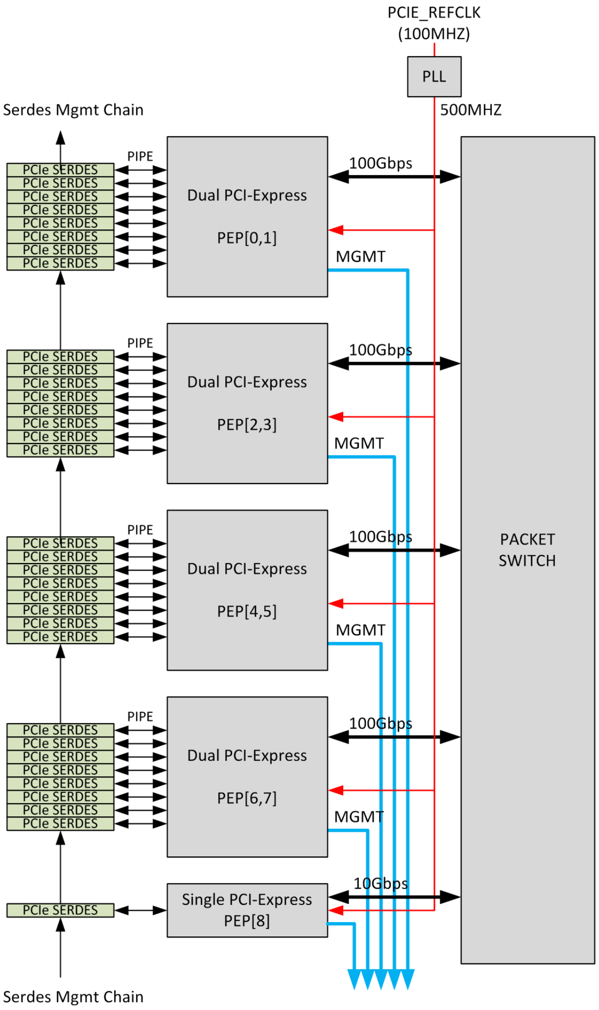


Figure 6-1 PCIe Overview

Each PEP supports the following features:

- PCIe 3.0 Compliance with the following optional and extended features:
  - — One physical function.
  - — A single traffic class.
  - — A single virtual channel.
  - — MSI-X interrupts only, does not support MSI or INTx.
  - — Function Level Reset.
  - — Advanced Error Reporting (AER).
  - — Device Serial Number (SPD).
  - — Alternative Requester ID (ARI).
  - — Single Root IO Virtualization (SR-IOV). Exposes up to 64 Virtual Functions (VF).

  - Spread Spectrum Clocking (SSC), with five clock sources for the PEPs.
  - Lane reversal support (Under certain restrictions. Refer to Section 6.2).
  - Lane polarity statically configured.
  - Maximum link width of X8 with support for X1, X2 and X4 in Gen1, Gen2 or Gen3.
  - Maximum Read Request Size (MRRS) of 4 KB.
  - Maximum Payload Size (MPS) of 512 bytes.
  - Advertised Flow Control Credits:
    - Posted Header: 64
    - Posted Data: 32
    - Non-Posted Header: 64
    - Non-Posted Data: 32
    - Completion Header and Completion Data have infinite credits (accepted immediately).
- Optional support for TLP Processing Hints (TPH)
- Host access to any internal registers of the switch.
- DMA controller for packet transfer:
  - Scattered-gathered DMA packet transfer.
  - Interleave DMA to CPU between RX and TX.
  - Packets optionally fragmented into multiple buffers.
  - TCP/UDP checksum offload.
  - TCP Segmentation Offload (TSO).
  - Receive Side Scaling (RSS).
  - TLP Processing Hints (TPH).
  - Split header/data buffers.
  - 256 queues.
  - Flow control from fabric.
  - Traffic shaping per VF in TX.
  - Performance in x8 mode:
    - 50 Gb/s at 256 B+ packets for GEN3 (25 GbE for GEN2, 12 GbE for GEN1).
  - Performance in x4 mode:
    - 25 Gb/s at 256 B+ packets for GEN3 (12.5 GbE for GEN2, 6.25 GbE for GEN1).
  - Performance in x1 mode:
    - 5 Gb/s at 256 B+ packets for GEN3 (2.5 GbE for GEN2, 1.25 GbE for GEN1).
  - The FM10000 supports only one reference clock for entire device, not one per PCIe interface.
- Jumbo packets (up to 15 KB per packet).

Each PEP supports the following transactions:

- From hosts:
  - Configuration read/write requests.
  - Memory read/write requests:
    - As being an end point, the FM10000 does not support memory read lock, which is handled as an Unsupported Request
    - Read/write requests ordering preserved.
  - Read completion to read requests from end point.
  - PCIe Messages (Message Code[7:0]; Routing r2r1r0):
    - Supported messages:
      - PM_Active_State_NAK (0x14; 100b)
      - PME_Turn_Off (0x19; 011b)
    - Messages handled as Malformed TLP:
      - Assert_INTx/Deassert_INTx
    - Silently dropped messages:
      - Unlock (0x00; 011b)
      - Hot-plug messages (0x40, 0x41, 0x43, 0x44, 0x45, 0x47, 0x48; 100b)
      - Slot power limit (0x50; 100b)
      - Vendor-defined type 1 (0x7F; 000b, 010b, 011b, 100b)
    - Dropped messages handled as Unsupported Request:
      - Vendor-defined type 0 (0x7E, 000b, 010b, 011b, 100b)
      - All other unlisted messages
- From endpoints:
  - Read completion to read requests from host.
  - Configuration read/write completion
  - MSIx interrupts.
  - Non-posted reads from the DMA engine.
  - Posted Writes from the DMA engine.
  - PCIe Messages:
    - PME_TO_Ack
    - COR_ERR
    - ERR_NONFATAL
    - ERR_FATAL

And the following internal transactions:

- Packet receive and transmit (up to 15 KB).
- Read/write to other internal devices.

- Read/write from other internal devices to local registers.

The PEP does not support:

- I/O transactions (return UR).
- Dynamic Link Width (DLW) of the PCIe interface.

Note: Refer to Section 6.6, "PCI Express Error Events and Error Reporting".

6.2 PCIe Lane Reversal Support

The PEP can negotiate a PCIe link in one of the following modes, as shown in Figure 6-2 and Figure 6-3:

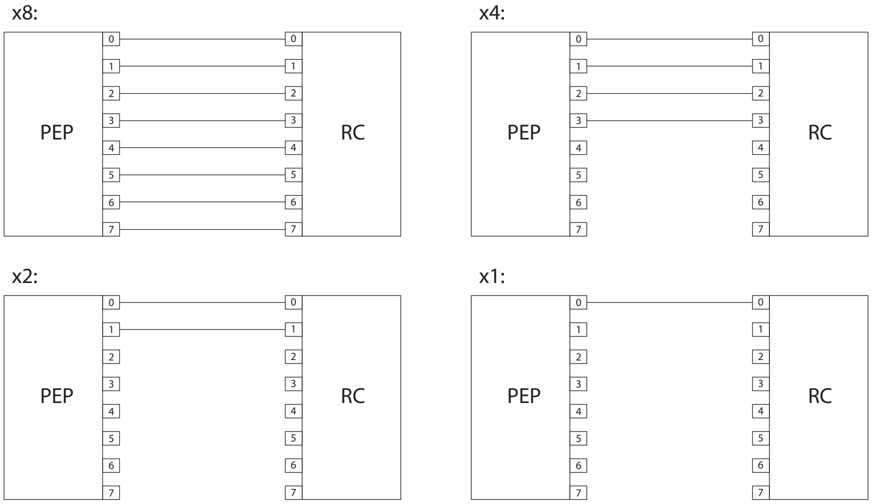


Figure 6-2 PCIe Lanes in Order

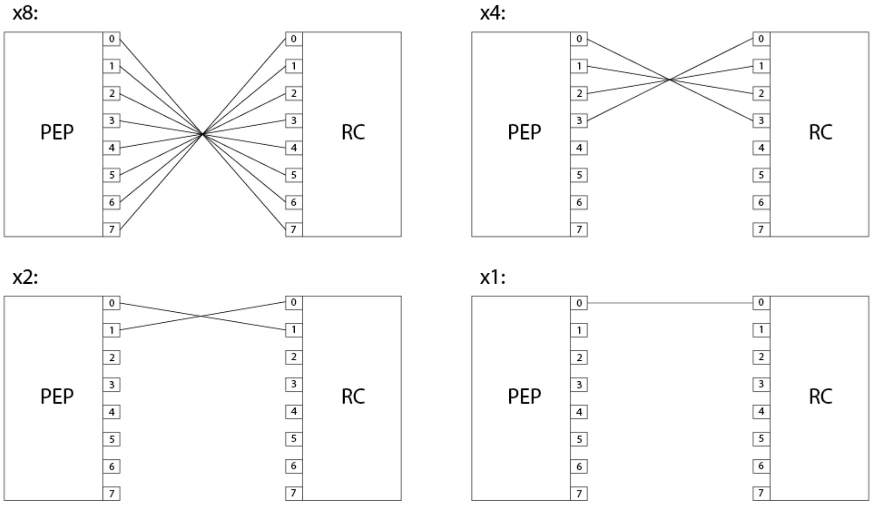


Figure 6-3 PCIe Lanes Reversed

Table 6-1 Swapping, Reversal and Polarity

Feature	Description	Example	EPL	PEP
Lane Swapping	Sequence order of all lanes in an interface can be swapped as long as the TX and RX are kept the same.	For a group of lanes 0123, the physical layout could be 3210 or 0213, or some other combination for both the RX and the TX in the same way.	Yes	No
Lane Reversal	Sequence order of all lanes in an interface can be reversed.	For a group of lanes 0123, the physical layout could be 3210.	N/A	Yes
Polarity Reversal	The P/N pair in a RX or RX lane can be inverted. TX and RX are independent.	TX P/N can be physically inverted as N/P. This is independent of the RX P/N in the same interface.	Yes By API configuration file.	Yes Automatic as part of PCIe protocol.

The SerDes are centrally managed (refer to Section 6.9) as well as locally managed. The central management is used for manufacturing and at boot time for initial configuration, the local management is used during normal operation.

Figure 6-4 shows the block diagram for the dual PEP blocks. If the BIFURCATION signal is de-asserted, all SerDes are attached to PEP[0] and the packet interface is routed to PEP[0], the PEP[1] remains unused and maintained in reset to save power. If the BIFURCATION signal is asserted, the bottom four SerDes are attached to PEP[0], the top four SerDes to PEP[1] and the packet interface operates in 2x25 GbE mode, one 25 GbE for each PEP.

Note: The BIFURCATION signal is driven by the overall device Management and is defined in DEVICE_CFG.*PCIeMode*[0..3].

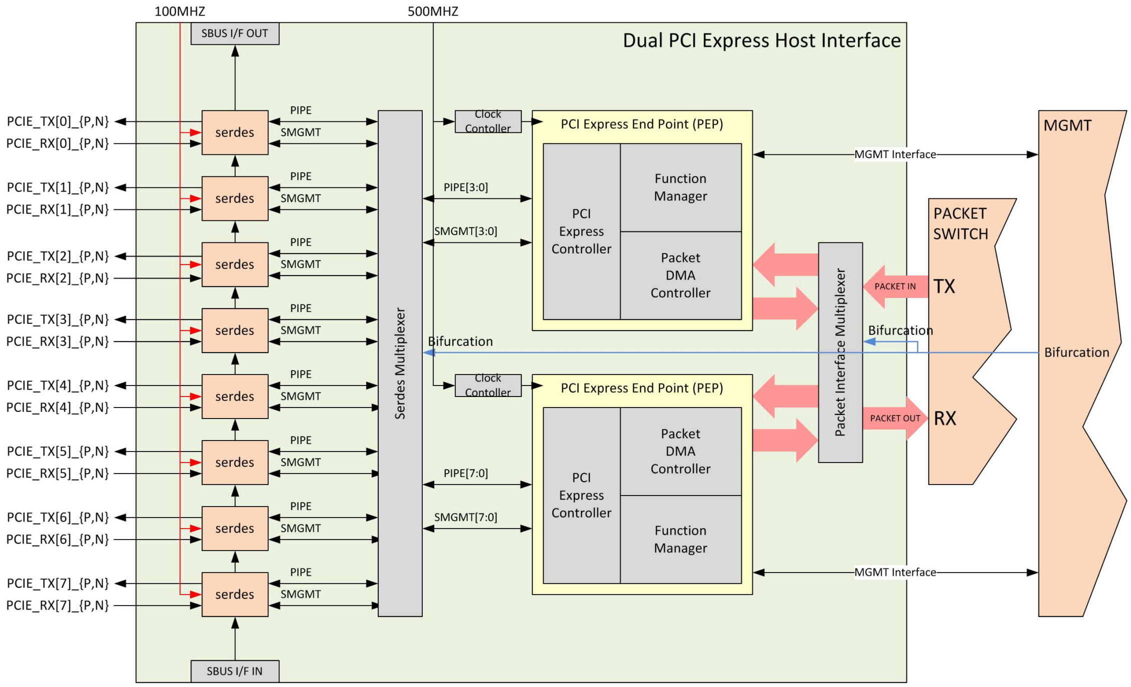


Figure 6-4 PCIe Interface with Dual PEP

Figure 6-5 shows the block diagram for the single PEP block. There are no bifuraction signals in this case. This block supports a single SerDes.

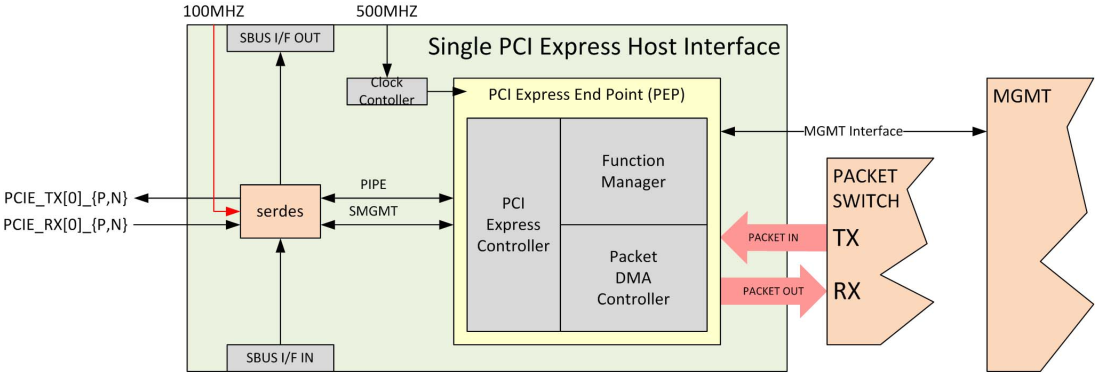


Figure 6-5 PCIe Interface with Single PEP

6.3 End Point Overview

Figure 6-6 shows the PEP block diagram itself.

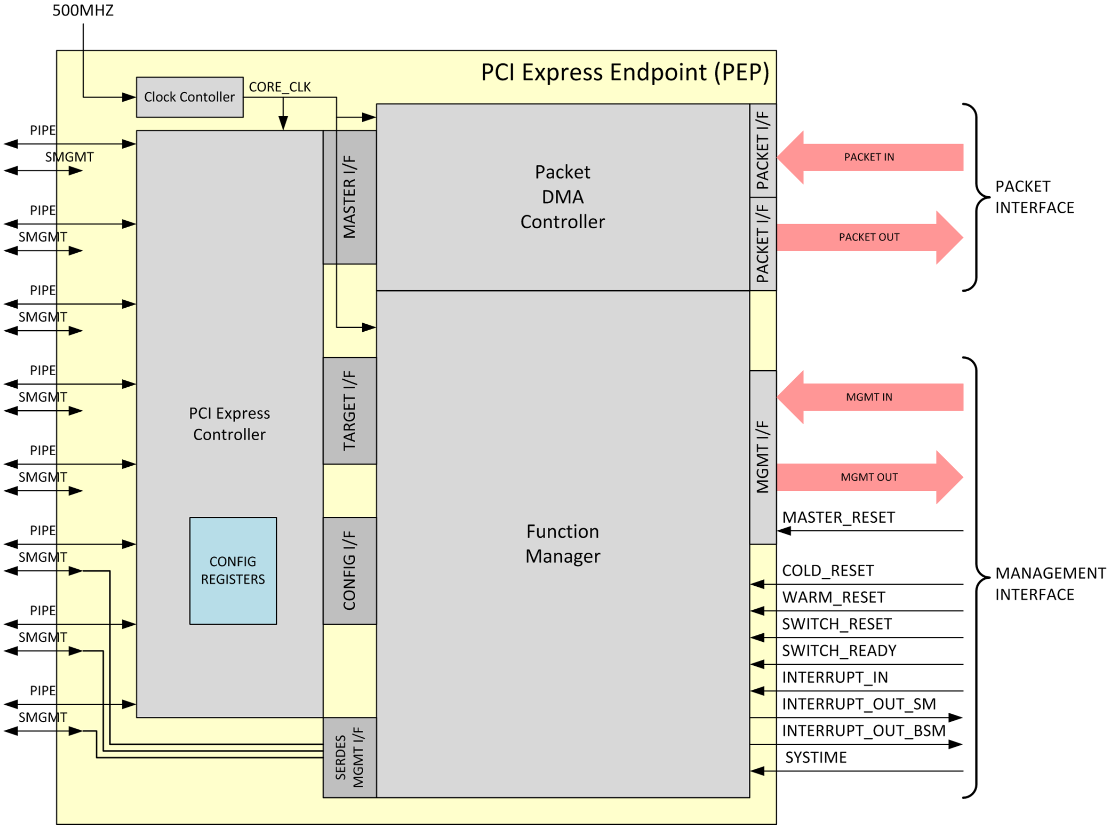


Figure 6-6 PCIe End Point Block Diagram

PEP contains the following blocks:

- PCIe controller.
- Clock controller.
- Packet DMA Controller.
- Function Manager.
- Reset and Power Manager.

The main interfaces in an endpoint are:

- Packet interface to switch fabric.
- Management interface to overall device management.
- Serial interface:
  - — Direct differential input/output signals.

- Serial bus (SBUS) SerDes management interface.
  - This interface ripples from one PCIe module to the next.
- SerDes management interface.
- PCIe slave interface.
  - For host initiated transactions (request and response).
- PCIe master interface.
  - For DMA controller initiated transactions (request and response).
- PCIe configuration interface.
  - Access to internal config registers of PCIe controller.
- DMA controller management interface.
  - Controls the DMA.
- PCIE reference clock (500 MHz).
- Module management:
  - Enable — Indicates if this PEP is enabled or not. Asserted according to SKU.
  - Management reset — De-asserted when device management is up.
  - Core reset — De-asserted when 500 MHz stable.
  - Warm reset — Indicates a fundamental reset to the PCIe.
  - Switch ready — When de-asserted, it indicates a reset of the Ethernet switch and switch reboot has not completed.
  - Interrupt in — Indicates a switch interrupt toward host.
  - Interrupt out to SM — Indicates a host/PCIe event to the Switch Manager.
  - Interrupt out to BSM — Indicates a host/PCIe event to the Boot State Machine.
- System reference time.

The PCIe controller interfaces to the SerDes via a PIPE 4.0 interface and implements the full link layer and transport layer of PCIe as defined in PCI-Express gen 3 specification. The master bus is used by the DMA controller to access to host resources via memory read or memory write request, the read response follows the same path. The slave interface is used by the host to manage this interface or any other component of the device. The PCIe controller includes all the configuration registers for this unit, those registers follows standard PCIe configuration space and are accessible via configuration read/write cycles from a PCIe host. The configuration registers are also available via a back door to allow initialization of configuration registers at power-up or for general debugging purpose.

The function manager manages the entire PCIe endpoint. It arbitrates between hosts access and internal registers access from other masters in the switch to access to local registers in the unit. Local registers are either configuration registers inside the cores, global registers of the interface or a bridge to DMA controller registers. The register controller can receive read/write requests from a locally attached host or from the switch internal management crossbar, and processes them in the order they were received. There are no hardware support for lock accesses. Therefore, if two hosts access perform a read-write sequence on the same register at about the same time, the ordering might cause one write to be lost. The Low Speed Management block contains a TEST_AND_SET register, which can be used to implement a software semaphore or, alternatively, the system software could use PCIe mailboxes. A sub-unit of the function manager is the reset and power manager. This sub-unit manages reset states, power states and interrupts to or from central FM10000 management.

The DMA controller is responsible for transferring incoming packets or outgoing packets from the internal port to the host memory. The controller initiates read/write accesses to host memory on its own to either retrieve/update buffer descriptors or to transfer packets. The controller contains FIFOs that are used to buffer packet payloads and buffer descriptor caches to hold descriptors expected to be processed soon.

6.4 Power Up and Fundamental Reset Handling

The FM10000 can support multiple hosts via multiple independent PCIe endpoints and, as such, provides extra functions to better control reset requests.

- **ENABLE** — Controls if a particular PEP is enabled or not.
- **MASTER_RESET** — General device reset. Asserted with MASTER_RESET; de-asserted only when power and PCIE_REFCLK stable.
- **COLD_RESET** — General PCIe host interfaces reset. De-asserted only when PCIE core clocks (500 MHz) are stable.
- **WARM_RESET** — PCIe PEP specific reset. Generally follows the external pin PCIE_RESET_N (one per host) except at initial bring-up when this signal is delayed until the SerDes have been downloaded.
- **Hot Reset** — A reset propagated in-band across a Link using a Physical Layer mechanism and targeting a single PCIe endpoint. This is mostly processed the same way as a WARM_RESET.
- **PFLR** — Function-Level Reset of the Physical Function. It is initiated by the OS setting the InitiateFLR bit in the PCIe endpoint configuration space register PCIE_CFG_PCIE_DEV_CTRL. It resets the PCIe configuration space and the PEP internal data path, excluding the PCIe interface.
- **SWITCH_RESET** — A reset targeting the Ethernet switch function only.
- **Data Path Reset** — A reset of the internal blocks of the PEP, excluding the PCIe interface.

Each PCIe endpoint supports the following device states:

- D3 cold (D3c) — No power.
- D3 disabled (D3d) — Power is applied but the host interface is disabled.
- D3 hot (D3h) — Hot Reset asserted (via in-band message).
- D0 reset (D0r) — Out of reset but initial setup not completed.
- D0 uninitialized (D0u) — Out of reset and firmware load complete.
- D0 active (D0a) — Fully active state.

The D3d is entered by default on certain PEPs depending on SKU (see Section 2.0, “Architecture Overview”), and this cannot be changed. Furthermore, the Switch Manager could choose to disable a specific PEP by software using the DEVICE_CONFIG.*PCIeEnable*[i] to 0, disabling PEPs reduce the power to the minimum. The Switch Manager can disable a PEP only while the PEP is in reset (D3h, warm reset asserted), not before.

Note: Accessing any register in a disabled PEP is not allowed, as this can cause a lock up of the management infrastructure.

The D0r state is not one defined by PCI SIG, this is a state where the end point has power and is out of reset but some initial configuration through the serial boot or other hosts has not happened yet. This state is not reflected on the physical link to the host and thus un-differentiable from an end point in reset.

The endpoint also supports the following link states:

- L0
- L1
- L3

ASPM (active state power management) is supported (L1 supported). However, the device states D1,D2 and the link states L2, L0s are not supported.

The management generates four reset-related signals to each PCIE endpoint:

- COLD_RESET
- PCIE_WARM_RESET
- SWITCH_RESET
- SWITCH_READY

The COLD_RESET is asserted whenever MASTER_RESET (the general device reset) is asserted, and only gets de-asserted after MASTER_RESET is de-asserted and after the 500 MHz core clock is stable.

The COLD_RESET is interpreted by the PCIe endpoint as a command to reset all circuitry of the endpoint to its initial state.

The PCIE_WARM_RESET is asserted after a COLD_RESET or via assertion of the PCIE_RESET_N[i] pin or via software. While in this state, any management access from any other master in the FM10000 to this unit is ignored (write flushed, read returned with data 0). Also, packets from the switch are drained, and in-flight packet to the switch are terminated immediately with an error code. The PCIE_WARM_RESET is de-asserted after COLD_RESET is de-asserted, the SerDes firmware has been download, PCIE_RESET_N is de-asserted, and management PCIE reset override is cleared.

The SWITCH_READY is de-asserted immediately after a MASTER_RESET gets asserted, but could also be de-asserted by software at any time. De-assertion of SWITCH_READY indicates that the switch fabric is reset. While SWITCH_READY is de-asserted, an in-flight packet to the switch is drained internally in the PEP and an in-flight packet from the switch is terminated immediately with an error code.

The SWITCH_RESET is asserted immediately after a MASTER_RESET or asserted shortly after any SWITCH_READY de-assertion. The signal causes the switch fabric and its direct interfaces (including the PCIE fabric packet interface) to be reset. Every de-assertion of SWITCH_READY is followed by an assertion of SWITCH_RESET. Every de-assertion of SWITCH_RESET is followed (generally after a delay for switch configuration) by assertion of SWITCH_READY.

Figure 6-7 shows the timing diagram for the initial power up.

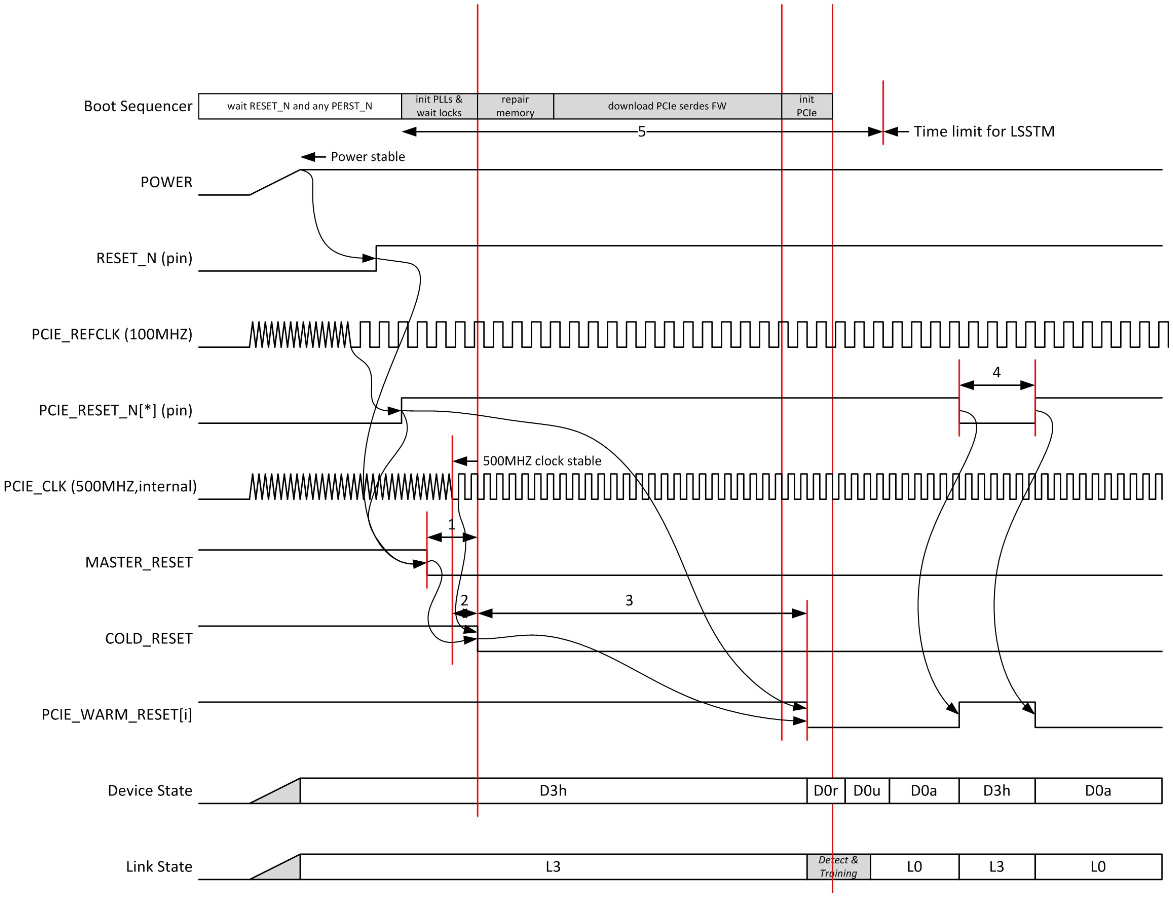


Figure 6-7 PCIe Power Up Timing Diagram

The timing constraints are listed in Table 6-2.

Table 6-2 Power Up Timing Constraints

Reference	Min	Typical	Max	Description
1	500 µs			MASTER_RESET de-assertion to COLD_RESET
2	100 µs			500 MHz clock stable to COLD_RESET de-assertion
3	5 ms	6 ms	18 ms	COLD_RESET de-assertion to PCIE_WARM_RESET de-assertion
4	100 µs			Minimum PCIE_RESET_N Pulse Width

The diagram shows the relationship between the external RESET_N pin and PCIE_RESET_N pins and the internal MASTER_RESET, PCIE_CLK, COLD_RESET and PCIE_WARM_RESET. The COLD_RESET and PCIE_WARM_RESET force a D3h state on this endpoint and physical links are in L3 state, the endpoint remains in this state as long as the serial boot or external host does not take it out of this state by de-asserting the resets. This time is not defined, it could be as long as desired.

Once the PCIE_WARM_RESET is de-asserted, the device enters a D0r state where an external host or serial boot is expected to do minimal re-configuration of the config registers and also minimal SerDes initialization. The physical links are maintained in an L3 state during that time. The serial boot or external host terminates this state by writing into the PCIE_CTRL to transit the physical link out of its L3 state and start initialization. The device enters D0u until the physical link is up and the host has completed the device configuration.

Figure 6-8 shows timing diagrams for switch or end point resets.

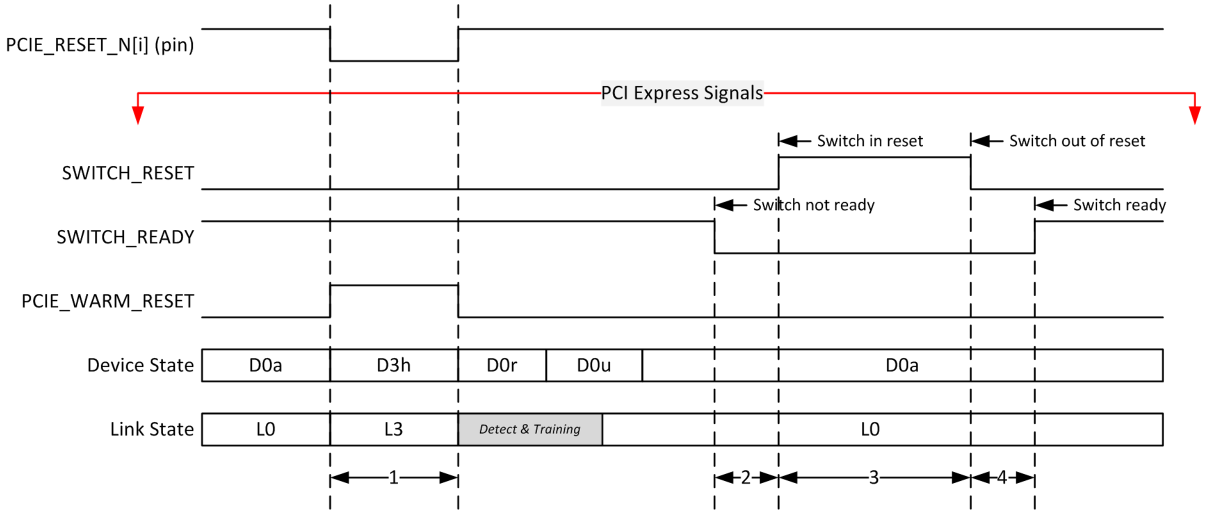


Figure 6-8 PCIe Reset Timing Diagrams

The timing constraints are listed in Table 6-3.

Table 6-3 Reset Timing Constraints

Reference	Min	Description
1	100 µs	Minimum PCIE_RESET_N Pulse Width.
2	100 µs	SWITCH_READY de-assertion to SWITCH_RESET assertion.
3	>1 ms	SWITCH_RESET pulse width.
4	100 µs	SWITCH_RESET de-assertion to SWITCH_READY assertion.

The PCIE_WARM_RESET could be asserted via external PCIE_RESET_N[i] pin or via internal software register write and remains asserted for at least 1ms.

The SWITCH_RESET or SWITCH_READY are asserted by software and there is a strict timing relationship between SWITCH_READY and SWITCH_RESET as shown in Figure 6-8.

6.4.1 Data Path Reset

This reset is initiated by setting the *DataPathReset* bit in PCIE_DMA_CTRL register. It has the effect of resetting internal blocks (RHI, RPP, and THI) and their associated memories and registers. On the PEP interface with the switch, it is handled like a WARM_RESET. It has no effect on the PCIe interface and PCIe block and on the interface with the management block. The PEP is kept internally in this reset state also while it is in D3hot state.

6.5 PEP Reset Sequences

Reset sequences have different priorities. A higher priority reset preempts a lower priority reset sequence being currently executed internally, starting the highest sequence from its beginning. Resets with the same priority do not preempt each others, but the internal reset sequence duration may be extended so that the broadest resetting effect of the two will occur.

Reset events are listed below from highest to lowest priority:

1. WARM reset (PCIE_RESET_N assertion or SOFT_RESET.*PCIeReset*[i] bit set), PCIe reset (PCIe in-band reset, or in case of surprise, PCIe link down).
2. PFLR, a D3hot to D0 transition, Data Path reset.
3. VFLR.

6.5.1 WARM and PCIE Reset

1. In case SM needs to reset to the PEP, it should first ask the Host to initiate a PCIe reset.
2. Reset initiated by PCIE_RESET_N de-assertion (system event), or by SM writing SOFT_RESET.*PCIeReset*[i], or by host root complex issuing a PCIe in-band reset, or in case of surprise PCIe link down.
3. In case of PCIe reset, the global *HotReset* interrupt is issued to SM/BSM.
4. PEP flip flops and memories are reset. *NotInReset* bit in PCIE_IP register is cleared.
    a. WARM reset case — Entire PEP is reset excepted to BAR4 access.
    b. Other cases — Entire PEP is reset except PCIe sticky bits and CSR access logic (including BAR4 access).
5. PEP re-asserts the *NotInReset* bit once the PCIE_WARM_RESET signal is de-asserted and memory initialization sequence is complete.
6. BSM either polls the *NotInReset* bit until it is read as 1b, or waits for the *OutOfReset* interrupt from the PEP.
    a. If PEP is still not out of reset after 150 μs, this is a fatal HW error. In such a case, a COLD reset is required.
7. BSM re-loads PCIe configuration space items and other settings from NVM.
8. BSM sets *LTSSM_ENABLE* bit in the PCIE_CTRL register to initiate the PCIe link training sequence.
9. When PCIe link is up, and power state is D0u, BIOS/Kernel initializes the PCIE configuration space and invokes the SW driver.
10. SW driver re-configures the PEP DMA.

6.5.2 Physical Function FLR (PFLR) and D3hot to D0 Transitions

The FLR bit is required for the Physical Function (PF) and per Virtual Function (VF). Setting this bit for a VF only resets the part of the logic dedicated to the specific VF and does not influence the shared part of the port. Setting the PF FLR bit resets the entire function. A FLR requires driver interaction to quiesce the function.

A FLR reset to the PF function in an IOV mode is equivalent to a FLR reset in a non-IOV mode. All VFs in the PCIe function of the PF are affected. The affected VFs are not notified of the reset. PFLR and D3hot to D0 transitions have no effect on the PCIe link.

1. OS sets the FLR bit or write 00b in the *PwrState* field of the PCIE configuration space.
2. Global PFLR (or *DeviceStateChange*) interrupt is issued to SM/BSM
3. Internal PCIE configuration space is reset as well as the PEP internal data path and memories. *NotInReset* bit in PCIE_IP register is cleared.
4. PEP re-asserts the *NotInReset* bit once the memory initialization sequence is complete.
5. BSM either polls the *NotInReset* bit until it is read as 1b, or waits for the *OutOfReset* interrupt from the PEP.
    a. If PEP is still not out of reset after 150 μs, this is a fatal HW error. In such a case, a WARM/COLD reset is required.
6. BSM re-loads PCIe configuration space items and other settings from NVM.
    a. The re-configured settings are only those returned to their default value on PFLR events.
7. After 100 ms, OS re-configures the PCIE configuration space and invokes the SW driver
8. SW driver re-configures the PEP DMA.

6.5.3 Data Path Reset

1. SW driver sets the *DataPathReset* bit in the PCIE_DMA_CTRL register.
2. Global *DataPathReset* interrupt is issued to SM/BSM.
3. Internal PEP data path and memories are reset. *NotInReset* bit in PCIE_IP register is cleared.
    a. Data path-related interrupts which were already logged before the reset are sent to host anyway.
4. PEP re-asserts the *NotInReset* bit once the memory initialization sequence is complete.
5. SW driver polls the *NotInReset* bit until it is read as 1b.
    a. If it is still not asserted after 150 μs, this is a fatal HW error. In such a case, a WARM/COLD reset is required.
6. SW driver re-configures the PEP DMA.

6.5.4 Transition to D3hot

1. Software writes 3 in the *PwrState* field of the PCIE configuration space.
2. PEP enters the *DataPathReset* state.
3. Global *DataPathReset* and *DeviceStateChange* interrupts are issued to SM/BSM.
4. Internal PEP data path and memories are being reset. *NotInReset* bit in PCIE_IP register is cleared.

6.5.5 Virtual Function FLR (VFLR)

A VF operating in an IOV mode can issue a FLR. VFLR resets the resources allocated to the VF. It also clears the PCIe configuration for the VF. There is no impact on other VFs or on the PF. Whenever possible, the software driver must place the function into a quiescent state before initiating the FLR. Placing a function into FLR can result in data corruption due to stale completions. This occurs when the function has been reset while there are still outstanding reads from that function. The following flow details the FLR procedure for quiescing and resetting the function:

1. Software that's performing the FLR synchronizes with other software that might potentially access the VF directly, and ensures such accesses do not occur during this algorithm.
2. Software clears the entire VF_CFG_CMD[VF] register, disabling the VF from issuing any new requests on the PCIe interface.
   a. Clearing of the *BusMasterEn* bit prevents any further upstream requests from the virtual function.
3. Software polls the *TransactionPending* bit in the VF_CFG_PCIE_DEV_CTRL[VF] register until it is cleared or until it is reasonably certain that the completion time out has expired.
4. Software sets the *InitiateFLR* bit in the VF_CFG_PCIE_DEV_CTRL[VF] register and waits for 100 ms before going to Step 9.
5. Hardware sets the *DetectEvent* bit that corresponds to the VF in the PFVFLRE bitmap register, sets the EICR.*VFLR* bit, and issues an immediate interrupt to the PF Driver.
6. Hardware clears the VF configuration space, and the VF mailbox memory is blocked (returns all 0s on read) until the PF clears it.
   a. Pending interrupts to the VF are dropped and cleaned up if issued while the VF is resetting Otherwise they may be issued as spurious interrupts to other VFs or to the re-enabled VF.
7. Consequential to receiving a VFLR interrupt from PEP, PF Driver reads the PFVFLRE bitmap register to detect what is/are the VF(s) that got reset. PF Driver then closes all the queues owned by the VF(s), making use of the non-graceful Queue Disable Operation flows listed below, and remaps them to the PF. If OS has temporarily masked all interrupts to PF, this step may take a few tenths of milliseconds. If PF driver is down, it is assumed it has closed all VF queues prior to going down.
   a. Receive Queue Disable, starting from Step 5.
   b. Transmit Queue Disable, starting from Step 3.
8. Once the Queue Disable flows are completed (i.e. *ENABLE* bits read as 0b), or once PF Driver is up again:
   a. PF Driver clears the VFLR event(s) by writing 1b in the corresponding *ClearEvent* bit(s) in the PFVFLREC bitmap register.

b. PF Driver clears the 64-bytes VF mailbox memory to release its read access. The least significant byte of PCIE_VFMBMEM[0] is written last by PF, since it is the trigger used by hardware for the release.

9. Once the 100 ms time-frame ends, Software can re-initialize the virtual function.
10. PF Driver can re-allocate and re-enable the queue resources.

6.6 PCI Express Error Events and Error Reporting

6.6.1 General Description

PCIe defines three error reporting paradigms:

- **Baseline capability** — Required for all PCIe devices and define the minimum error reporting requirements.
- **Advanced Error Reporting (AER) capability** — Defined for more robust error reporting and is implemented with a specific PCIe capability structure.
- **Proprietary mechanism** — Used for errors which are better handled by the software device driver using CSRs.

6.6.2 Error Events

Table 6-4 lists the error events identified by the FM10000 and the response in terms of logging, reporting, and actions taken. Refer to the PCIe specification for the effect on the PCI Status register.

Table 6-4 Response and Reporting of PCIe Error Events

Error Name	Error Events	Default Severity	Action
Physical Layer Errors			
Receive Error	• 8b/10b Decode Errors • Packet Framing Error	Correctable Send ERR_CORR	TLP to initiate NAK, drop data, DLLP to drop.
Data Link Errors			
Bad TLP	• Bad CRC • Illegal EDB • Wrong Sequence Number	Correctable Send ERR_CORR	TLP to initiate NAK, drop data.
Bad DLLP	• Bad CRC	Correctable Send ERR_CORR	DLLP to drop.
Replay Timer Timeout	• REPLAY_TIMER expiration	Correctable Send ERR_CORR	Follow LL rules.
Replay Num Rollover	• REPLAY NUM Rollover	Correctable Send ERR_CORR	Follow LL rules.
Data Link Layer Protocol Error	• Violations of Flow Control Initialization Protocol	Uncorrectable Send ERR_FATAL[1]	

Table 6-4 Response and Reporting of PCIe Error Events [Continued]

Error Name	Error Events	Default Severity	Action
TLP Errors			
Poisoned TLP Received	• TLP With Error Forwarding	Uncorrectable ERR_NONFATAL[1] Log Header	If Completion TLP: 1. If error is non-fatal (default case): • Send error message if advisory. • Retry the request once and send advisory error message on each failure. • If fails, send uncorrectable error message, 2. If error is defined as fatal: • Send uncorrectable error message. • Report internally the error as UR.
ECRC Check Failed	• Failed ECRC check	Uncorrectable ERR_NONFATAL[1] Log Header	1. If error is non-fatal (default case): • Send error message. 2. If advisory error is defined as fatal: • Send uncorrectable error message. • Return CA completion to non-posted bad ECRC packets.
Unsupported Request (UR)	• Receipt of TLP with unsupported Request Type • Receipt of an Unsupported Vendor Defined Type 0 Message or of other unsupported messages that are not silently dropped • Invalid Message Code • Wrong Function Number • Received TLP Outside BAR Address Range • Receipt of a Request TLP during D3hot, other than Configuration and Message requests • Access to the I/O region	Uncorrectable ERR_NONFATAL[1] Log Header	Send completion with UR.
Completion Timeout	• Completion Timeout Timer Expired	Uncorrectable ERR_NONFATAL[1]	1. If error is non-fatal (default case) • Send error message. 2. If advisory error is defined as fatal. • Send uncorrectable error message.
Completer Abort (CA)	• Received Target Access with illegal data size • VF access to a queue not owned by it	Uncorrectable ERR_NONFATAL[1] Log Header	Send completion with CA.
Unexpected Completion	• Received Completion Without a Request For It (Tag, ID, etc.)	Uncorrectable ERR_NONFATAL Log Header	Discard TLP.
Receiver Overflow	• Received TLP Beyond Allocated Credits	Uncorrectable ERR_FATAL[1]	Receiver behavior is undefined.

Table 6-4 Response and Reporting of PCIe Error Events [Continued]

Error Name	Error Events	Default Severity	Action
Flow Control Protocol Error	• Minimum Initial Flow Control Advertisements • Flow Control Update for Infinite Credit Advertisement	Uncorrectable ERR_FATAL[1]	Receiver behavior is undefined.
Malformed TLP (MP)	• Data Payload Exceed Max_Payload_Size • Received TLP Data Size Does Not Match Length Field • TD field value does not correspond with the observed size • PM Messages That Don't Use TC0. • Usage of Unsupported VC • Target request crosses a 4 KB boundary	Uncorrectable ERR_NONFATAL[1] Log Header	Drop the packet. Do not free FC credits.
Completion with Unsuccessful Completion Status		No Action (already done by originator of completion)	Free FC credits

[1] If the error is defined as 'advisory non fatal', the severity is ERR_CORR.

6.7 Internal Memory Read Errors

Internal memory read errors are classified into two categories whether they are correctable or not.

Correctable ECC errors are reported to the local host and to the switch manager via the SRAM Error interrupt. The switch manager reads the PCIE_SRAM_IP register to know on which block the error occurred. The correctable error bits in the register are cleared by write performed by the switch manager.

Note: If a correctable ECC error was inserted during the write, the error may be reported repetitively anytime the device reads the memory line. In this case, a PEP reset is required to stop the reports.

Uncorrectable ECC/parity errors are handled by the following flow:

1. If an uncorrectable ECC/parity error is detected on a PEP memory, the PEP data path is stopped.
   a. Current packet transmission to the switch is immediately terminated with an ECC error code.
   b. Current packet reception from the switch is internally stopped (the signal is temporarily de-asserted until PEP goes reset).
   c. Current TLP to the host is either dropped (if the error is in PCI block), or terminated with the EDB symbol.
2. The concerned block is reported in the PCIE_SRAM_IP register, and a SRAM Error interrupt is sent to the local host and to the Switch Manager (SM).
   a. MSI-X vector and logic used for reporting uncorrectable error on mailbox or interrupt memories is implemented in flip-flops and not in memories.
3. SM and local host read the PCIE_SRAM_IP register, and SM does the following:
   a. If the error is in RHI/THI/PP/PCA block, SM isolates the PEP, sends a message to host via MBX to initiate a Data Path reset, and clears the corresponding bit in PCIE_SRAM_IP.
   b. If the error is in FUM/PCW block, SM isolates the PEP, sends a message to host via MBX to request a PCIe in-band reset, and clears the corresponding bit in PCIE_SRAM_IP.

c. If the error is in MBX_INT block, SM isolates the PEP, does not handle the error, and goes to step 4. Local host issues a PEP reset (either WARM reset, D0 -> D3 -> D0 sequence, PCIe in-band reset, or PFLR).

d. In case the error is in the PCI block, it is most probable that the host reboots further to a PCIE AER notification from root complex, and thus reset the PEP.

4. Once it receives the OutOfReset interrupt from PEP, SM clears the PCIE_SRAM_IP bit(s) that was read as 1.

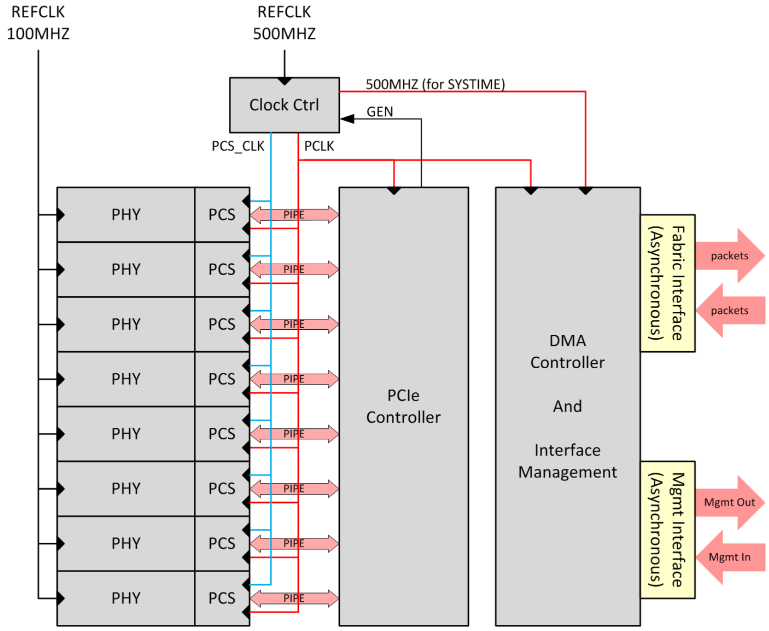
   a. In case no PEP reset occurred within one minute since the uncorrectable SRAM error was reported, SM goes back to step 3.

6.8 Clocking

The PCIe receives 2 clocks:

- A very low-jitter 100 MHz reference clock received by LVPECL I/O pads.
- A 500 MHz reference clock generated by a PLL in the management block.

The 100 MHz is distributed to each SerDes and has no other function in the PCIe unit. The 500 MHz is to be used to derive operating clocks depending of the negotiated speed of the interface; GEN1, GEN2, GEN3. Both the 100 MHz and 500 MHz are stable before COLD_RESET is de-asserted. The distribution is shown in Figure 6-9.



SPEED	PCS_CLK	PCLK	PIPE_WIDTH
2.5GT/s	250MHZ	125MHZ	16 bits
5.0GT/s	500MHZ	250MHZ	16 bits
8.0GT/s	500MHZ	500MHZ	16 bits

Figure 6-9 PCIe Clock Distribution

The clock controller needs to generate:

- A 250 MHz (GEN1) or 500 MHz (GEN2/GEN3) to SerDes digital interface (SerDes CORE_CLK)
- A 125 MHz (GEN1), 250 MHz (GEN2) or 500 MHz (GEN3) for the PIPE interface (SerDes PCLK), the PCIe controller, the DMA controller and function manager.
- A 500 MHz (equal to input) for the function manager for time reference (SYSTIME)

Also, an internal 400 MHz clock generated by LANE 0 is distributed to all lanes. Figure 6-10 details the clocking around the SerDes hard macro (PHY+PCS). The LANE 0 is never powered down for the system to work properly.

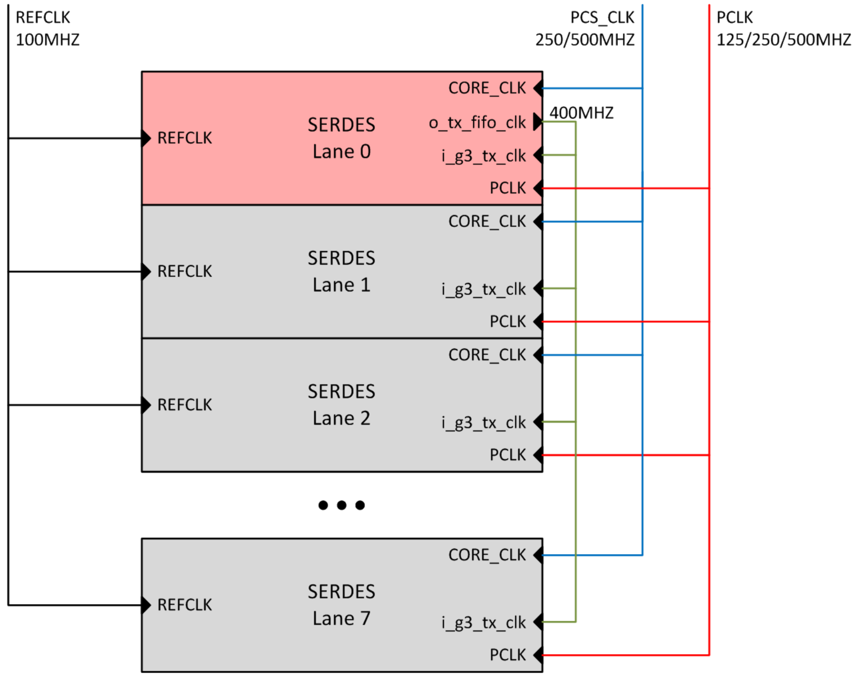


Figure 6-10 PCIe SerDes Clocking

6.9 SerDes Management

The SerDes connections are shown in Figure 6-11.

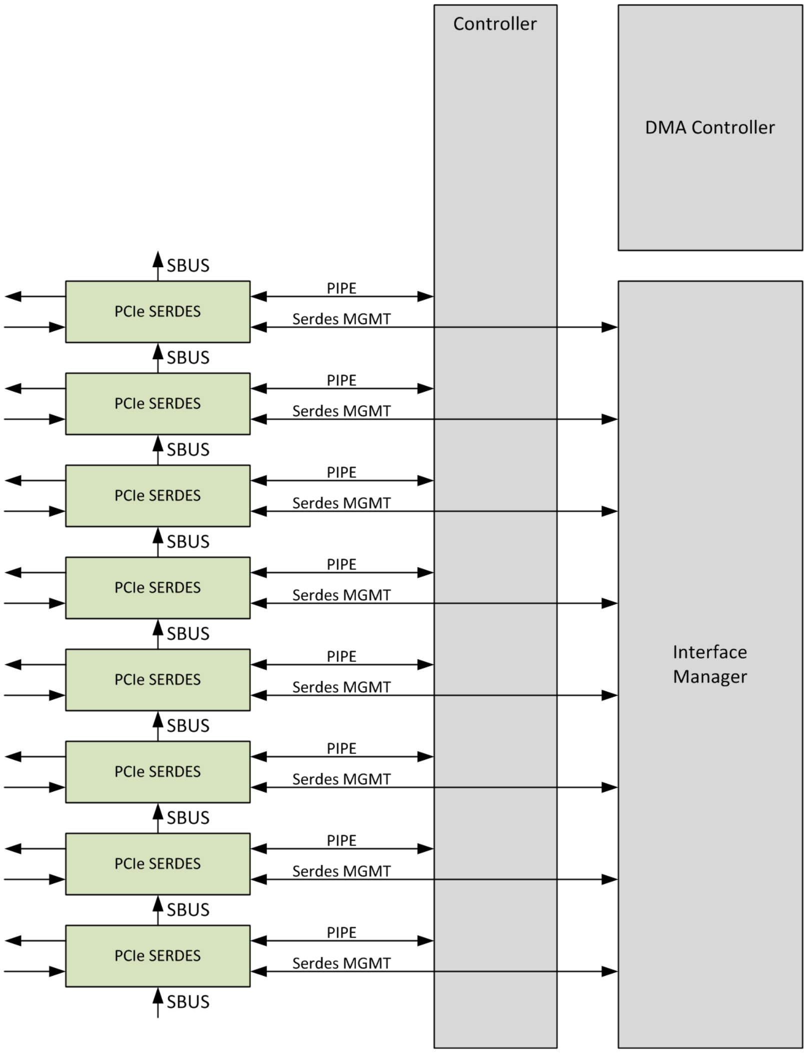


Figure 6-11 PCIe SerDes Interfaces

The SerDes have four interfaces:

- High-speed serial interface (2 GbE, 5 GbE, 8 GbE).
- High-speed parallel interface (PIPE).
- PCIE management interface for common setup & control.
- SBUS interface for initial firmware download and backdoor SerDes management.

The PCIe controller taps to the PIPE interface and the static parallel interface, the PIPE interface is connected to the controller while the management interface is connected to the function manager.

The SBUS interface is available through the management module (refer to Section 2.2).

The PCIE management interface is available through the PCIE_SERDES_CTRL register.

6.10 32-bit/64-bit Addressing

The device reports 64-bit mode in PCIE_BARs and requires a 64-bit address to be loaded in the PCIE_BARs. However, the device accepts 3 DW transactions format and assumes that the upper 32 bits are zero for decode purpose.

As required by the PCIe 3.0 Base specification for addresses below 4 GbE, the device initiates transactions using 32-bit format request addressing (3 DW TLP header format). For addresses at 4 GbE or above, the device initiates transactions using 64-bit format request addressing (4 DW TLP header format).

6.11 Memory Map

The PCIe supports 3 BARS for PF and 2 BARs for VF:

The PCIe physical function supports 3 BARs:

- BAR0/BAR1 (2 MB) — Access to per-PCIe endpoint register map.
- BAR2/BAR3 (8 KB) — Access to MSI-X registers.
- BAR4/BAR5 (64 MB) — Access to all registers of the FM10000 (enable at boot), excepted to the registers of the local PCIe endpoint and to the MSI-X registers of all PCIe endpoints.

Each virtual function supports 2 BARs:

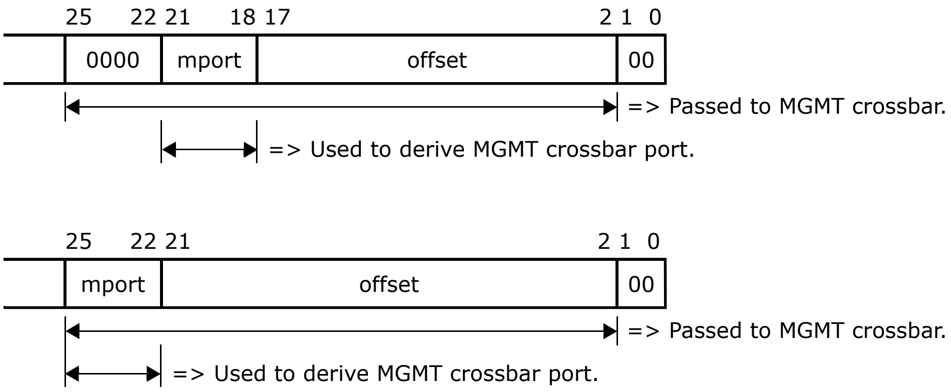
- BAR0/BAR1 (4 MB) — Access to per-VF register map.
- BAR2/BAR3 (8 KB) — Access to MSI-X registers.

All registers in the global map (BAR 4/5) are accessed as 32-bit registers and are sent to management crossbar. If the ultimate target register is greater than 32-bit, it is assumed that software issues multiple 32-bit register writes in sequence without interleaving. For all BARs, if the host attempts to access registers with an access greater or smaller than 32-bit, or with a non-32-bit aligned access, the access is completed with a 'Completer Abort' status.

Any read access to a non-defined address returns all 0s.

The port to use on the management crossbar is derived from the address. The address passed to the management crossbar is a word address and is equal to ADDR[25:2] of the address received from host.

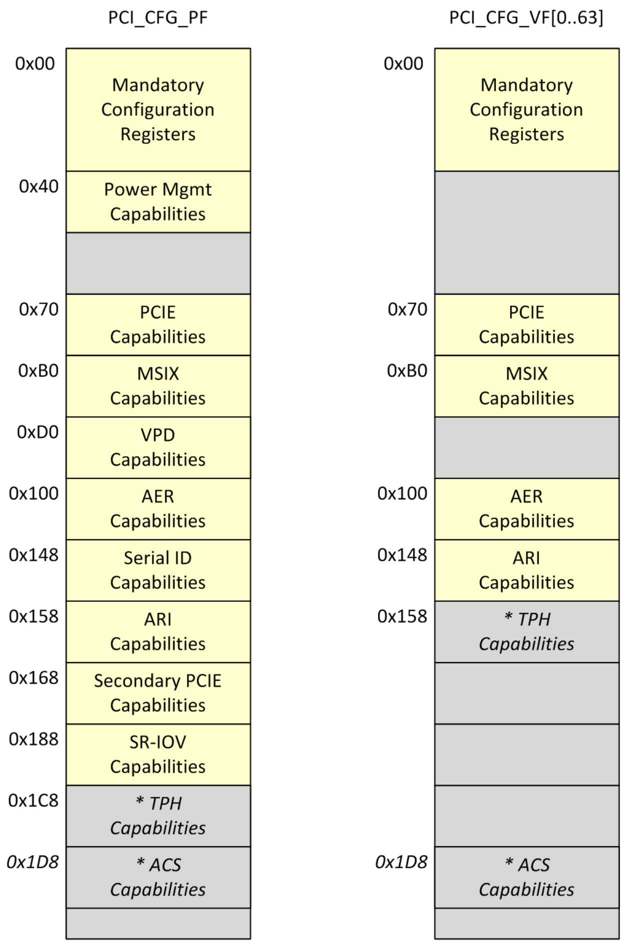
FROM HOST:


25
22 21
18 17
2 1
0
0000
mport
offset
00
=> Passed to MGMT crossbar.
=> Used to derive MGMT crossbar port.
25
22 21
2 1
0
mport
offset
00
=> Passed to MGMT crossbar.
=> Used to derive MGMT crossbar port.

```
if ( ADDR[25:22] == 0 ) crossbarPort = ADDR[21:18]
else if ( ADDR[25:22] < 0x0C ) crossbarPort = ADDR[25:22]
else crossbarPort = 0x0F
```

The access to BAR4 is controlled for the whole unit via the PCIE_CTRL.*BAR4_Allowed* register. This register is set by default after a WARM_RESET, thus allowing BAR4 accesses. It can be cleared to prevent future accesses. VFs have no access to BAR4, which means the Switch Manager cannot sit on a VF. Mapping of the BAR4 offset is identical to the BAR0/1 mapping (see below).

The VF and PF configuration registers are shown in Figure 6-12. They have similar configuration offsets and accessible 2 ways; via config cycle by a host, via back door access through the global register map view.



** Not exposed capability. Can be exposed by NVM setting.*

Figure 6-12 PCI Configuration Memory Map

The PF BAR0/1 address space is a 4 MB memory space divided as follow:

```
      BYTE ADDRESS
000000_xxxxxxxx_xxxxxxxx => General PF register visible from PF (64KB)
000001_QQQQQQQQ_xxxxxxxx => RX queue configuration and state (256Q x 256B)
000010_QQQQQQQQ_xxxxxxxx => TX queue configuration and state (256Q x 256B)
0001xx_xxxxxxxx_xxxxxxxx => General PF register visible from PF (256KB)
001xxx_xxxxxxxx_xxxxxxxx => Backdoor DBI space (config & core debugging) (DBI_CS2=0) (accessible only
                                                                                      from BAR4)
01QQQQ_QQQQdddd_ddddxxxx => TX queue descriptors (256Q x 256D x 16B)
100xxx_xxxxxxxx_xxxxxxxx => Backdoor DBI space (config & core debugging) (DBI-CS2=1) (accessible only
                                                                                      from BAR4)
```

Where:

```
QQQQQQQQ => PHYSICAL QUEUE #
DDDDDDDD => DESCRIPTOR #  (push model only)
```

Notes: The global map allows access to all registers of the device including all PCIe interfaces but excluding the unit that generated the BAR4 request in the first place.

MSI-X registers of all PCIe end-points are not visible through BAR4/5.

The PF and VF configuration registers are only visible through BAR4/5, not from BAR0/1.

DBI space is a backdoor access from the Switch Manager to the PCIE configuration space. Not all registers are writeable through this interface. The detailed access list is out of the scope for this document.

The VF BAR0/1 address space is similar:

```
     BYTE_ADDRESS
00000_xxxxxxxx_xxxxxx00 => General VF register (64KB)
00001_QQQQQQQQ_xxxxxx00 => RX queue configuration and state (256Q x 256B)
00010_QQQQQQQQ_xxxxxxxx => TX queue configuration and state (256Q x 256B)
00011_xxxxxxxx_xxxxxxxx => NOT ADDRESSABLE
001xx_xxxxxxxx_xxxxxxxx => NOT ADDRESSABLE
01xxx_xxxxxxxx_xxxxxxxx => NOT ADDRESSABLE
1QQQQ_QQQQDDDD_DDDDxxxx => TX queue descriptors (256Q x 256D x 16B)
```

The virtual function queue number mapping to physical queue number is detailed in the next sections.

All accesses in the PF BAR0/1 and VF BAR 0/1 are 32-bit accesses except for TX queue descriptors writes, which use 64-bits MMIO copy.

All accesses in the PF BAR2/3 and VF BAR 2/3 are 32-bit accesses except for MSI-X vectors. The controller supports 32, 64 or 128-bit write accesses (byte and 16-bit accesses not supported). All other register writes are 32-bit access only. All register reads are 32-bit access only.

If host attempts to access a non-populated address within the BAR, the cycle is completed with no error logged. Writes do not have effect and reads returns all 0s. Host attempt to access an address out of the BAR is handled as a UR, and an error is logged.

6.11.1 RX Queue Mapping

For RX queues management, the host only needs access to queue control/status registers. It does not need access to the descriptor cache itself.

The address relative to BAR0/1 for both PF and VF is:

00001_QQQQQQQQ_xxxxxxxx

For accesses from Physical Function, the Q extracted from the address is the physical queue number.

```
q = ADDR[15:8]
```

For access from a Virtual Function, the Q extracted from the address is the virtual queue number and must be translated to a physical queue number using a redirection table. The redirection table supports up to 256 queues if number of VFs is smaller than 8, or 32 queues per VF otherwise. The table PCIE_RQMAP is used for this purpose:

```
vf = from TLP
vq = ADDR[15:8]
q = ( NumVFs > 8 ) ? PCIE_RQMAP[vf * 32 + vq & 31]
                               : PCIE_RQMAP[vf * 256 + vq]
```

A specific queue can only belong to either a physical function or to one virtual function at any time. This is defined in PCIE_RXQCTL[q].*OwnedByVF* (set to 0b if queue belongs to PF) and PCIE_RXQCTL[q].*VF* (set to VF # if the queue does not belong to PF). Those attributes are managed (READ/WRITE) by the physical function and READ-ONLY for virtual function.

The decoding could possibly derive a physical queue that does not belong to the entity addressing it. Therefore, the access to that queue must be validated before being granted.

```
if ( AccessFromVF && ((PCIE_RXQCTL[q].VF != VF) || (PCIE_RXQCTL[q].OwnedByVF == 0)) )
 )
   # This virtual function not allowed to use this queue,
   # report the error, log Q and interrupt once
```

If not granted, the error is logged, a fault interrupt is issued, and the access is flush (for write) or return all 0s for read request.

The register PCIE_RQMAP belong to PF and only this entity could access it. The VF has no access to this table.

6.11.2 TX Queue Mapping

The TX queue mapping is similar as in Rx queue mapping except that the mapping table is PCIE_TQMAP when access is from a VF.

```
vf = from TLP
vq = ADDR[15:8]
q = ( NumVFs > 8 ) ? PCIE_TQMAP[vf * 32 + vq & 31]
                   : PCIE_TQMAP[vf * 256 + vq]
```

And, as for RX, the queue access must be verified.

```
if ( AccessFromVF && ((PCIE_TXQCTL[q].VF != VF) || (PCIE_TXQCTL[q].OwnedByVF == 0))
 )
   # This virtual function not allowed to use this queue,
   # report the error, log Q and interrupt once
```

The register PCIE_TQMAP belong to PF and only this entity could access it. The VF has no access to this table.

6.11.3 Virtual Function Queue Configuration

Per-queue registers have definitions when accessed through Virtual Function or Physical Function. However, some registers can only be modified when accessed via the physical functions to prevent a VM to affect another VM or PF. The restrictions are:

PF Register	VF Register	VF Restriction
PCIE_TPH_RXCTRL	PCIE_VFTPH_RXCTRL	Read only
PCIE_RXQCTL	PCIE_VFRXQCTL	Read only, excepted to *ENABLE* field
PCIE_RXDCTL	PCIE_VFRXDCTL	Read only
PCIE_QPRC	PCIE_VFQPRC	Read only
PCIE_QPDRC	PCIE_VFQPDRC	Read only
PCIE_QBRC	PCIE_VFQBRC	Read only
PCIE_TPH_TXCTRL	PCIE_VFTPH_TXCTRL	Read only
PCIE_TXQCTL	PCIE_VFTXQCTL	Read only
PCIE_TQDLOC	PCIE_VFTQDLOC	Read only
PCIE_TQDLOC	PCIE_VFTQDLOC	Read only
PCIE_QPTC	PCIE_VFQPTC	Read only
PCIE_QPDTC	PCIE_VFQPDTC	Read only
PCIE_QBTC	PCIE_VFQBTC	Read only
PCIE_TQDLOC	PCIE_VFTQDLOC	Read only
PCIE_TXSGLORT	PCIE_VFTXSGLORT	Read only
PCIE_PFVTCTL	PCIE_VFVTCTL	Read only

6.11.4 TX Descriptors Mapping

The TX process supports also a push model (detailed in the TX section) where the host has access to the TX memory to write descriptors directly. The TX descriptor memory is 128 KB, and is shared with all VFs and PF and also used to hold TX descriptors for the pull model.

The address map to access this resource is wide enough to provide flexibility in partitioning of this memory.

FROM PF:

```
01QQQQ_QQQQDDDD_DDDDxxxx => TX queue descriptors (256Qx256Dx16B) (PUSH)
                              Q = physical queue number
                              D = descriptor number
```

FROM VF:

```
1QQQQ_QQQQDDDD_DDDDxxxx => TX queue descriptors (256Qx256Dx16B) (PUSH)
                              Q = virtual queue number
                              D = descriptor number
```

As for queue management, the queue number indicated in the VF TX queue descriptor access is to be translated to physical queue number using the same method and validated.

The following registers in the Physical Function defines the area used for the push model.

- Descriptors locations in cache (indexed by physical queue number):
  - — PCIE_TQDLOC[0..255].*Base* — Defines base address in 64-byte chunks.
  - — PCIE_TQDLOC[0..255].*Size* — Defines log2(number of descriptors), min 4 desc.

This is shown in Figure 6-13.

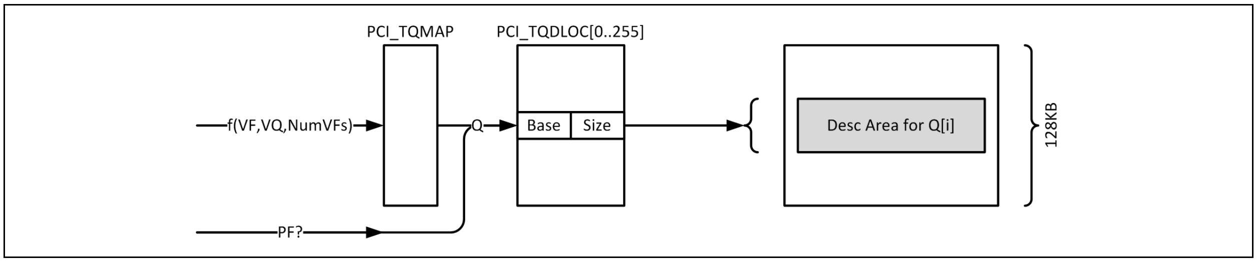


Figure 6-13 PCIe Tx Cache Memory Allocation

Note: The descriptors location must be defined regardless of the push model is used or not. The index to PCIE_TQDLOC is the physical queue number.

For descriptors writes:

1. If VF, translate Q to physical queue using QMAP. If PF, then Q is physical queue.
2. Validate that access to this queue is allowed from PF or VF#.
3. Extract descriptor #, mask according to PCIE_TQDLOC[q].*Size*.
4. TX cache memory address is:

```
offset = (PCIE_TQDLOC[q].Base << 6)
       + ((D & ((1 << PCIE_TQDLOC[q].Size)-1) ) << 4)
       + ADDR[3:0]
```

6.12 Interrupts and Faults

6.12.1 Interrupt Sources

There are two categories of interrupts generated by the end-point:

- Local Interrupts:
  - Signaled to local host through Message Signals Interrupts Extended (MSI-X).
- Global Interrupts:
  - Signals to other components of the device via PCIE_INTERRUPT_OUT.

The local interrupts are:

- TX queue events (packets transmitted, VF or PF).
- RX queue events (packets received, VF or PF).
- Other events:
  - PF (logged in PCI_EICR, masked by PCI_EIMR):
    - 0: Mailbox
    - 1: PCIe fault
    - 2: Switch Ready (up/down) Event
    - 3: Switch Interrupt
    - 4: SRAM Error
    - 5: VFLR
    - 6: Max hold timer ended
  - VF: (check PCIE_VFMBICR)
    - Mailbox (1 per VF)

The global interrupts are:

- 0: Hot reset
- 1: Device state changes
- 2: Global mailbox interrupts
- 3: VPD request
  - Switch manager use the config back door to read VPD capabilities and perform the request and then clears the PCIE_IP.*VPD_Request* register bit.
- 4: SRAM Error
- 5: PFLR
- 6: Data path reset

- 7: Out of Reset
  - Use by the BSM to reload PEP defaults from EEPROM and then clear the PCIE_IP.*OutOfReset* register bit.
- 8: Timeout
  - Reported also in PCIE_HOST_MISC register for the case PCIE_IP reporting logic is stuck. Refer to Section 6.12.4.

6.12.2 Interrupt Mapping

Figure 6-14 shows the interrupt dispatching.

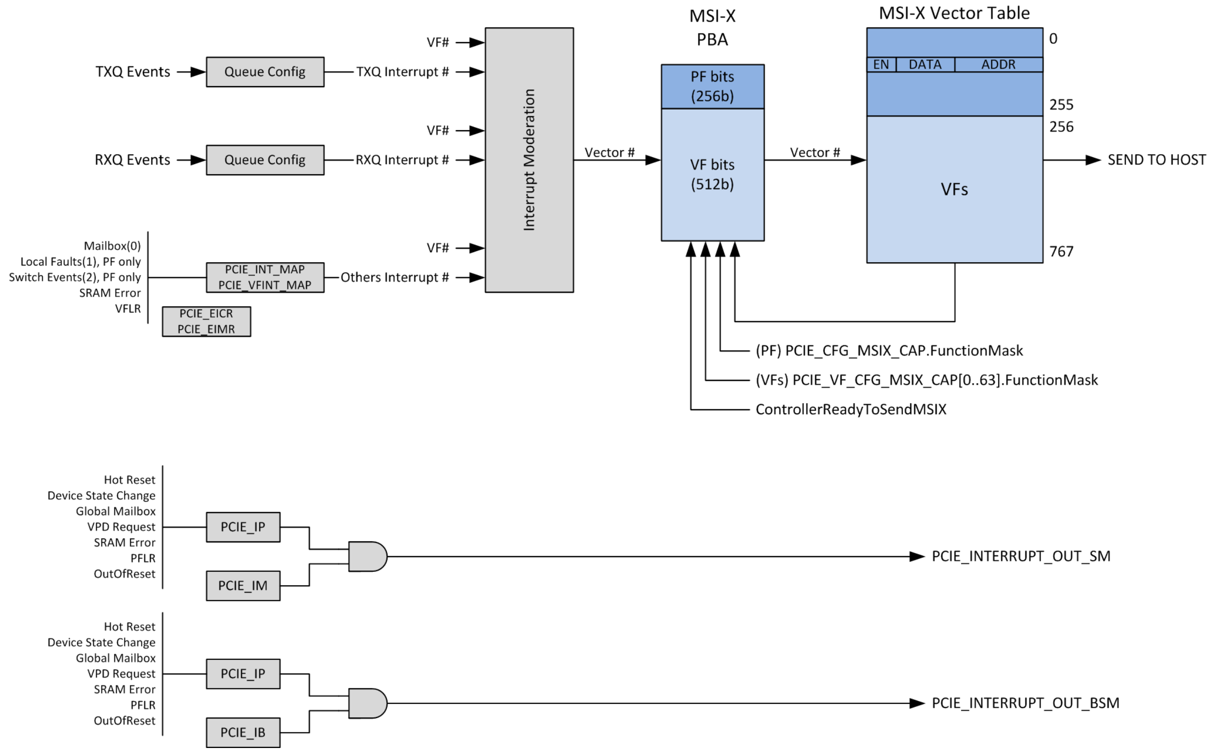


Figure 6-14 PCIe Interrupt

Processing of global interrupts to the SM is simple. Events are latched in PCIE_IP and could be masked using the PCIE_IM register. The PCIE_INTERRUPT_OUT_SM is asserted whenever the expression "PCIE_IP & ~PCIE_IM" is non zero. A similar processing is done for global interrupts to the BSM, making use of PCIE_IB register for the masking to BSM.

Processing of local interrupts is more complex.

The most common type of interrupts are TX queue and RX queue events. The queue configuration defines the VF or PF to which this queue belongs along with interrupt number and interrupt timers to use. Those are defined in PCIE_RXINT and PCIE_TXINT registers and detailed in the Rx and Tx processing sections.

The other local events are either occasional (by example mailboxes) or rare exceptions (faults or device state changes or reset), this category of events might either of interest to a VF or only to a PF. A map register, index by the event number, defines the MSI-X interrupt number to use for each of these events (PCIE_INT_MAP and PCIE_VFINT_MAP), the event is also latched in the PCIE_EICR register and could be masked using the PCIE_EIMR register. No new event is sent to host until it clears the corresponding register cause bit.

All interrupts are filtered through programmable interrupt moderators to prevent overload processor cores with too many interrupts.

The FM10000 supports 768 vectors partitioned statically between the PF (256 vectors) and the VFs (512). The allocation per VF depends on the number of VFs configured, and can be up to 32 vectors per VF. The Vector Index for the MSIx table is as follows:

```
if ( PF )
    vector_index = interrupt# & 255
else
    vector_index =  256 +
                   (NumVFs > 32 ) ? ((interrupt# & 0x007) + (VF << 3)) :
                   (NumVFs > 16 ) ? ((interrupt# & 0x00F) + (VF << 4)) :
                                    ((interrupt# & 0x01F) + (VF << 5)) ;
```

The interrupt posting consist of two independent processes:

- Interrupt moderation — Interrupts generated are then stored in the MSI-X Pending Bit Array table (PCIE_PBACL).
- Generate MSI-X vector to host — The vector is written to the host using the address and data configured for each vector (defined by host in the MSI-X Table) if allowed (interrupt not masked and PCI controller ready to send interrupt). The Pending Bit is cleared when the vector is actually sent to host.

There are 768 MSI-X Pending Bits which are visible through the PCIE_PBACL register and also through BAR2/3. The mapping of those 768 into registers depends on how many VFs are configured. The PCIE_PBACL has 8 x 32 bits for PF vectors and 64 x 32 bits registers for VF vectors (at least one 32-bit per VF even if there are only 4 bits used for a given VF). The following shows how to compute the PCIE_PBACL index and bit number from the vector index.

```
if ( vector_index < 256 )
    pba_index = vector_index >> 5;
    pba_bit   = vector_index & 0x1F;
else if ( NumVFs > 32 )
    pba_index = 8 + VF;
    pba_bit   = vector_index & 0x07;
else if ( NumVFs > 16 )
    pba_index = 8 + VF;
    pba_bit   = vector_index & 0x0F;
    pba_index = 8 + VF;
    pba_bit   = vector_index & 0x1F;
```

There is only one MSI-X vector table and pending bit array table per end point and it is shared between VF and PF. Both tables are accessible fully through the normal PF register map (BAR0/1) via PCIE_MSIX_VECTOR and PCIE_PBACL registers but should not normally be addressed this way, they should rather be addressed through the BAR2/3 of either PF and VF which provides a 4 KB alignment and discretionary memory mapping in the host memory. The mapping of BAR2/3 accesses to vector_index and pba_index is as follows:

```
PF access:
   BAR2/3: ADDR[13:0] = 0_vvvv_vvvv_0000 (4KB page)
       v = vector_index
       x = ignored
   BAR2/3: ADDR[13:0] = 1_xxxx_xxxz_zz00 (4KB page)
       z = pba_index
       x = ignored
VF access:
   BAR2/3: ADDR[13:0] = 0_vvvv_vvvv_0000 (4KB page)
       v = vector_index
       x = ignored
       vector_index = 256 +
                      (NumVFs > 32 ) ? ((v & 0x07) + (VF << 3)) :
                      (NumVFs > 16 ) ? ((v & 0x0F) + (VF << 4)) :
                                       ((v & 0x1F) + (VF << 5)) ;
   BAR2/3: ADDR[13:0] = 1_xxxx_xxxz_zz00 (4KB page)
       z = pba_index
       x = ignored
       pba_index = 8 +
                   (NumVFs > 8 ) ? VF :
                   (NumVFs > 4 ) ? ((z & 1) + VF << 1) :
                   (NumVFs > 2 ) ? ((z & 3) + VF << 2) :
                                   ((z & 7) + VF << 3) ;
```

6.12.3 Faults

The PCIe endpoint has multiple error checking features available throughout the module to enhance reliability and help tracking software errors. These errors are reported by the unit as “faults” interrupts to PF only and logged into the PCIE_EICR register:

- **PCIe Core Faults** — Event latched in PCI_EICR.*PCA_Fault*, fault record in PCI_PCA_FAULT.
- **Function Manager** — Event latched in PCI_EICR.*FUM_Fault*, fault record in PCIE_FUM_FAULT.

The fault records are all identical for all sub-units and are 128 bits wide:

- *Address* (64 bits) — The address at the time the fault was detected.
- *FaultSpecificInfo* (32 bits) — Extra info on this fault (fault dependent).
- *FaultType* (8 bits) — Fault value dependent on sub-unit.
- *VF* (6 bits) — Indicate VF that made the fault.
- *PF* (1 bit) — Set to 1b to indicate fault was made by PF.
- *Valid* (1 bit) — Set to 1b when a fault is logged, no further fault reported for this sub-unit until this bit is cleared.

6.12.4 Timeout Interrupt

Read request to PEP over the Management interface may timeout. Such error condition shall be handled as follows:

1. SM/BSM issues a read request to PEP over Management interface.
2. If the read times out, a global Timeout interrupt is sent to the SM/BSM and the PEP is prevented from issuing any further response.
   a. It means this read and any further read access to PEP by SM/BSM returns all 0s, until PCIE_HOST_MISC.*Timeout* bit is cleared (see step 4).

3. SM receives the interrupt and checks PCIE_IP register (normal global interrupt handling flow). PCIE_IP is read as all 0s in this case.
4. If PCIE_IP.*NotInReset* is read as 0b for more than 100 μs, SM checks PCIE_HOST_MISC.*Timeout* bit (which is not implemented inside the PEP):
   a. If PCIE_HOST_MISC.*Timeout* bit is read as 1b:
      1. SM isolates the malfunctioning PEP and relies on the host to reset the PEP (must be a WARM_RESET) within 1 minute, otherwise SM goes to step 4.b.
      2. WARM_RESET resets the PCIE_HOST_MISC.*Timeout* bit, which releases PEP read accesses from SM/BSM.
      3. Once PEP is out of reset, SM clears any cause bit in PCIE_IP (logged events in PCIE_IP are not reset by WARM_RESET).
   b. If PCIE_HOST_MISC.*Timeout* bit is read as 0b or if PEP is not reset by host within 1 minute: a HW fatal error occurred, a COLD_RESET is required.

6.12.5 Interrupt Moderators

The interrupt moderators consist of Interrupt Throttlers (ITR).

A global 16-bit timer counts up by 1 every 512 clock cycles (1 µs time base for PCIe GEN3). This timer is used as a base for the ITRs.

6.12.5.1 Interrupt Throttling

Interrupt throttling is a mechanism that guarantees a minimum gap between two consecutive interrupts. It is implemented in Interrupt Throttlers Registers (ITRs). The FM10000 support 768 ITRs, one ITR for each MSI-X vector. Each ITR has two programmable timers and one fix zero-time timer. All ITRs in use are linked together using a link list. ITRs are split into two registers:

- PCIE_ITR — Control/status of interrupts and timers, visible from PF and VF.
  — The VF view (PCIE_VFITR) is limited to the vectors used by this VF and follows the same allocation as the MSI-X vectors and thus depends on the number of VFs configured.
- PCIE_ITR2 — Link list of ITRs and status, visible from PF only.

The PCIE_ITR[0..767] includes the following:

- *Interval0* (12 bits) — Minimum interval between interrupts, timer 0.
- *Interval1* (12 bits) — Minimum interval between interrupts, timer 1.
- *Timer0Expired* (1 bit) — Timer 0 has expired.
- *Timer1Expired* (1 bit) — Timer 1 has expired.
- *Pending0* (1 bit) — Pending event waiting for time 0 expiration or rate limiter.
- *Pending1* (1 bit) — Pending event waiting for time 1 expiration or rate limiter.
- *Pending2* (1 bit) — Pending event waiting for rate limiter only.
- *Automask* (1 bit) — Indicates that the interrupt is masked immediately after being posted.

- *Mask* (2 bits) — Indicates/control the current mask state.
  - Read:

    00b = Interrupt is not masked (allowed).
    01b = Interrupt is masked (blocked).
  - Write:

    00b = Do not change the mask state.
    01b = Set the mask.
    10b = Clear the mask.

The PCIE_ITR2[0..767] defines the following:

- *NextVector* (10 bits) — Next vector in the list.
- *LastPosting* (16 bits) — Last time an MSI-X interrupt was posted.

Note: The *LastPosting* is read only fields in the ITRs registers. The *Pending* bits are Write-1-Set and allowed software to artificially force an interrupt. Setting *Pending2* triggers an interrupt as soon as possible, while setting Pending0 or Pending1 triggers an interrupt only once the corresponding timer has expired.

6.12.5.2 Interrupt Posting into PBA

The FM10000 delays posting interrupts into the Interrupt Pending Array until the interrupt rate limiter has credits, respective timers have expired and the interrupt is not masked. Once authorized, the device posts the interrupt as per MSI-X protocol, then logs the current time and clears all pending bits in the corresponding ITR and optionally mask the interrupt if the ITR[n].*AutoMask* is set.

The interrupt causes are mapped to one of the ITRs and one of the ITR timers by the PCIE_TXINT, PCIE_RXINT, PCI_VFINT_MAP and PCI_INT_MAP registers respectively.

- PCIE_TXINT[q].*Interrupt* (8 bits) — Relative to VF/PF reserved block.
- PCIE_TXINT[q].*InterruptTimer* (2 bits) — Must be 0,1,2.
- PCIE_RXINT[q].*Interrupt* (8 bits) — Relative to VF/PF reserved block.
- PCIE_RXINT[q].*InterruptTimer* (2 bits) — Must be 0,1,2.
- PCIE_INT_MAP[i].*Interrupt* (8 bits) — Relative to PF reserved block.
- PCIE_INT_MAP[i].*InterruptTimer* (2 bits) — Must be 0,1,2.
- PCIE_VFINT_MAP[i].*Interrupt* (5 bits) — Relative to VF reserved block.
- PCIE_VFINT_MAP[i].*InterruptTimer* (2 bits) — Must be 0,1,2.

The interrupt timer is either set to 0,1 or 2 where 0,1 are programmable timers, 2 is immediate timer.

The algorithm is the following:

```
for(;;)
{
  // Perform hosts accesses
  if ( HOST ACCESS TO PCIE_INT_CTRL, PCIE_INTR, PCIE_ITR, PCIE_ITR2)
     - perform access

  // Latch interrupts events
  if ( RX FRAME EVENT() )
  {
      // latch temporarily
      RxEvent[q] = 1;
  }
```

```
if ( RX FRAME EVENT() )
{
    // latch temporarily
    TxEvent[q] = 1;
}
if ( OTHER EVENT() )
{
    // latch temporarily
    // number of possible events are 5 for PF and 64x1 for VFs
    OtherEvent[x] = 1;
}

// Process events
if ( PCIE_INT_CTRL.EnableModerator == 1 )
{
    // Recover next event pending in Event[0..515]

    if ( q = getNextRxEvent() )
    {
      // q is queue number posting interrupt
      // f is function id (pf or vf#), recovered from RXQCTL

      t = PCIE_RXINT[e].InterruptTimer;
      v = mapToAbsoluteVector(PCIE_RXINT[e].Interrupt,f);

      // Check if possible to post interrupt
      if ( ( t == 2 ||
             t == 1 && PCIE_ITR[v].Expired1 ||
             t == 0 && PCIE_ITR[v].Expired0 ) &&
           PCIE_ITR[v].Mask == 0 )
           LogInterrupt(v);
        else if ( t == 2 )
           PCIE_ITR[v].pending2 = 1;
        else if ( t == 1 )
           PCIE_ITR[v].pending1 = 1;
        else if ( t == 0 )
           PCIE_ITR[v].pending0 = 1;
    }

    // Recover next event pending in TxEvent[0..255]
    else if ( q = getNextTxEvent() )
       - same as RX except using PCIE_TXINT[q], TxEvent

    // Recover next event pending in OtherEvent[0..3]
    else if ( q = getNextOtherEvent() )
       - same as RX except using PCIE_INT_MAP[event], OtherEvent

    else
    {

       // Process current vector

       v = PCIE_INT_CTRL.NextVector;
       // Software ensures that v always points to a valid vector
       PCIE_INT_CTRL.NextVector = PCIE_ITR2[v].NextVector;

       // Check for pending interrupts
           // Check if a pending interrupt could now be posted.
           if ( !PCIE_ITR[v].Timer0Expired &&
                (time-PCIE_ITR2[v].LastPosting) > PCIE_ITR[v].Interval0 )
                PCIE_ITR[v].Timer0Expired = 1
           if ( !PCIE_ITR[v].Timer1Expired &&
                (time-PCIE_ITR2[v].LastPosting) > PCIE_ITR[v].Interval1 )
                PCIE_ITR[v].Timer1Expired = 1
           if ( (PCIE_ITR[v].Pending2 ||
                 PCIE_ITR[v].Pending1 && PCIE_ITR[v].Timer1Expired ||
                 PCIE_ITR[v].Pending0 && PCIE_ITR[v].Timer0Expired) &&
                PCIE_ITR[v].Mask = 0 )
               LogInterrupt(v);

       }

    }
```

```
LogInterrupts(v)
{
  setPBA(v);
  PCIE_ITR[v].Pending1 = 0;
  PCIE_ITR[v].Pending2 = 0;
  PCIE_ITR[v].Pending0 = 0;
  PCIE_ITR[v].Time0Expired = 0;
  PCIE_ITR[v].Time1Expired = 0;
  PCIE_ITR2[v].LastPosting = time;
  if ( PCIE_ITR[v].AutoMask ) PCIE_ITR[v].Mask = 1;
}
```

Software could maintain the link lists of ITRs on the fly while the moderator is active provided the changes are done using the right sequence.

- To add Vector to a list, then in order:
  a. PCIE_ITR[new].*NextVector* = previous
  b. PCIE_ITR[previous].*NextVector* = new
- To remove Vector from a list, then in order:
  a. PCIE_ITR[previous].*NextVector* = PCIE_ITR[remove].*NextVector*
  b. PCIE_INT_CTRL.*NextTimer* = previous

6.12.5.3 Processing MSI-X Pending Bits

The FM10000 checks if the PCIe controller is ready to send a vector, and, if yes, select which vector to send among the 768 vectors. The device round-robin's among all vectors so that each vector has equal chance to be served, qualifying each vector by the vector enable and the function enable.

The Pending Bit array is cleared right after the vector has been sent to the controller.

6.13 Statistics

There are two types of stats; per queue and global.

Statistics per queue:

- QPRC[0..255] — RX Frame count forwarded to host (32 bits).
- QBRC[0..255] — RX Byte count forwarded to host (48 bits).
- QPRDC[0..255] — RX Frame dropped count (32 bits).
- QPTC[0..255] — TX Frame count forwarded from host (32 bits).
- QBTC[0..255] — TX Byte count forwarded from host (48 bits).

Global statistics:

- PCIE_STATS_TIMEOUT — Count Completion timeouts.
- PCIE_STATS_UR — Count Unsupported Requests completions.
- PCIE_STATS_CA — Count Completer Aborts completions.
- PCIE_STATS_UM — Count Unsupported Messages received from host (silently dropped).
- PCIE_STATS_XEC — Count Checksum Errors.

- PCIE_STATS_VLAN_DROP — Count VLAN Membership Drops.
- PCIE_STATS_LOOPBACK_DROP — Count Loopback Suppress Drops.
- PCIE_STATS_NODESC_DROP — Count No Descriptors Drop.

All counters are initialized to zero at power-up and at their relative reset events. Otherwise, they count up forever and wrap around.

6.14 Time Reference

The FM10000 maintains a synchronized time reference (SYSTIME) in all interfaces (PCIe and Ethernet), this time reference is used to time tag incoming packets to the switch. The Ethernet ports maintains a 32-bit reference time while the PCI-Express maintains a 64-bit reference time. Both times are unsigned numbers that rolls over at maximum values.

The centralized management module sends a set of four signals to each PCIe interface to maintain a local reference time. Figure 6-15 shows the timing diagram:

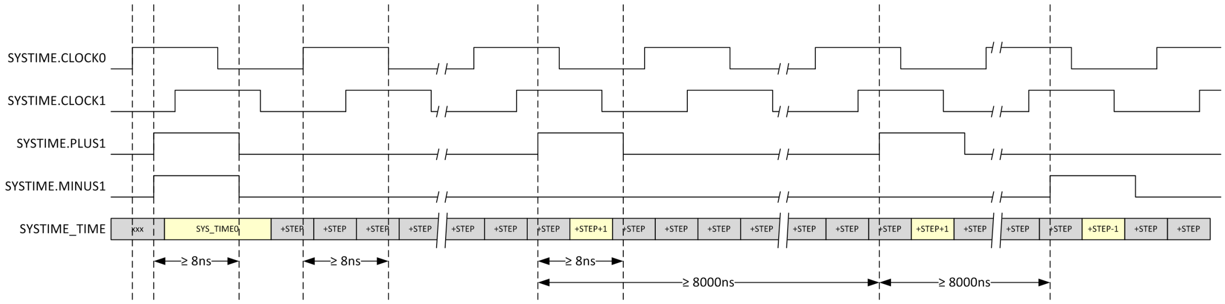


Figure 6-15 PCIe System Time Interface

The signals are:

- **SYSTEM.CLOCK0/1** — An asynchronous clock. Note that CLOCK0/CLOCK1 are guaranteed to be at the same exact rate but the relative phase is not guaranteed. For each edge, the local SYSTEM_TIME is incremented by STEP (corrected by +1 or -1 if necessary).
- **SYSTEM.PLUS1** — An asynchronous pulse requesting a +1 adjustment on the next SYSTEM_TIME increase.
- **SYSTEM.MINUS1** — An asynchronous pulse requesting a -1 adjustment on the next SYSTEM_TIME increase.

The STEP is configured in PCIE_SYSTIME_CFG.*Step* and is at least 2.

The local system time is only reset at COLD_RESET or when SYSTEM.PLUS1 and SYSTEM.MINUS1 are both sampled high. Otherwise, it continues to run un-interrupted while PCIE_RESET or SWITCH_RESET are asserted. It is assumed that the 500 MHz reference clock is used to sample the SYSTEM_TIME asynchronous signals, not the derated core clock (250/125 MHz) derived for GEN2/GEN1 interface. The skews between the four signals might vary as the signals are distributed across the device with no phase guaranteed alignment. Therefore, the implementation is as follows:

Table 6-5 PCIE System Time Action Table

CLOCK0	CLOCK1	PLUS1	MINUS1	Action
x	x	1	1	PCIE_SYSTIME = PCIE_SYSTIME0
↑	↑	—	—	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP
↑	—	—	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP
—	↑	—	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP
↓	↓	—	—	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP
↓	—	—	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP
—	↓	—	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP
↑	↑	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP +1
↑?	—	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP + 1
—	↑	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP + 1
↓	↓	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP + 1
↓	—	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP + 1
—	↓	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + STEP + 1
↑	↑	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP - 1
↑	—	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + STEP - 1
—	↑	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + STEP - 1
↓	↓	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + 2*STEP - 1
↓	—	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + STEP - 1
—	↓	—	↑	PCIE_SYSTIME = PCIE_SYSTIME + STEP - 1
—	—	↑	—	PCIE_SYSTIME = PCIE_SYSTIME + 1
—	—	—	↑	PCIE_SYSTIME = PCIE_SYSTIME - 1

The Ethernet switch time-tags packets at the time they are received, using a 32-bit time reference. The time tag is stored in lieu of the CRC. The PCIe converts this into a 64-bit time tag using the following conversion:

```
PacketTimeTag = "last 32 bits of packet";
if ( PCIE_SYSTIME[31:0] < PacketTimeTag )
   RDESC.TIME_TAG[63:32] = PCIE_SYSTIME[63:32] - 1;
   RDESC.TIME_TAG[31:0] = PacketTimeTag
else
   RDESC.TIME_TAG[63:32] = PCIE_SYSTIME[63:32];
   RDESC.TIME_TAG[31:0] = PacketTimeTag
```

On transmit (toward the switch), the PCIe adds a time tag equal to PCIE_SYSTIME[31:0] on all packets sent to the switch at the time the first byte of the packet (after FTAG) is transfered to the switch. The switch replaces this time tag by a CRC on the way out. This is useful for tracking internal delays in the switch.

6.15 DMA Controller Overview

6.15.1 Overview

The DMA controller transfers packets between the core switch and the host memory following a list of buffer descriptors which are organized in queues. The two directions operates independently of each other, the directions are relative to host point of view:

- Receiving: from fabric to host
- Transmitting: from host to fabric

Figure 6-16 shows the overall organization of the DMA controller.

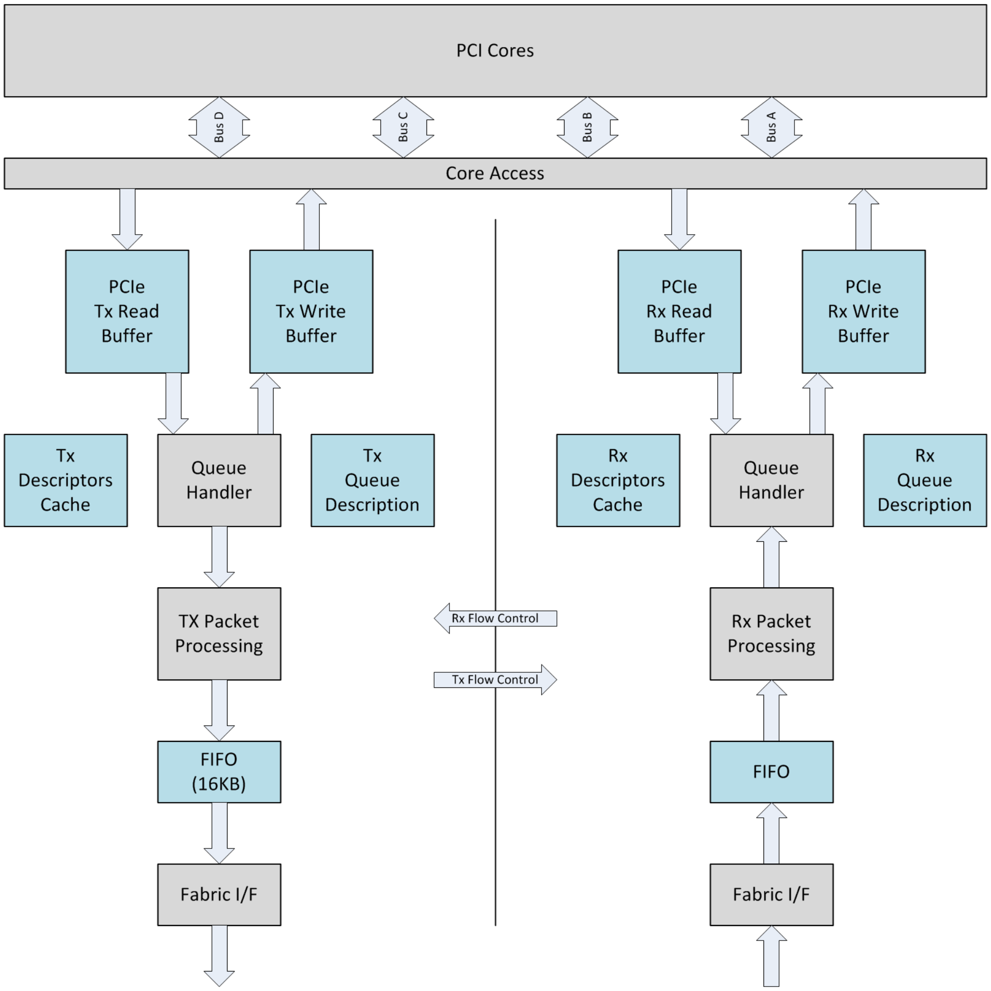


Figure 6-16 PCIe DMA Controller Overview

The RX packet processing performs packet parsing, TCP/UDP checksum computation, receive side scaling and queue mapping. The RX packet parsing might receive PAUSE packets from the fabric, in this case, the PAUSE packet is parsed, the content is passed to the packet transmission engine and the packet flushed, not passed to the queue handler. The RX FIFO is very shallow, no data needs to be stored in this queue for very long.

The RX queue handler forward the packet received to the host memory using the queue descriptors associated with each queue. To ensure maximum throughput, the queue handler prefetches a minimum of four descriptors per queue ahead of time to ensure that incoming data can flow through the host memory without enduring the latency required to fetch the buffers. The queue handler also writes back to the host memory the updated descriptors adding information on the packet received. The queue handler writes updates to descriptors and packet payload into a write buffer sorted by destination host.

In the transmit direction, the host post packets to queues and update the queue pointers in the DMA controller. The TX queue handler then retrieves the descriptors from memory ((if internal free space for at least four descriptors) and proceed in retrieving the packets from memory forwarding the payload to the fabric.

The TX packet processing unit then proceed in doing packet parsing and TCP/UDP checksum computation for queues that require this service. Because the packets are up to 15 KB, the TX packet processing writes the data into the FIFO but hold forwarding to the fabric until the end of the packet and the checksum written back to the proper location at the head of the packet.

TCP segmentation is a cooperative work between the queue handler and the packet processing.

The Core Access module forward read or write requests from TX or RX in a round-robin fashion alternating between READ/WRITE queues for TX and RX. In a dual host environment, the Core Access module is capable of pushing data request to each host independently, i.e. not having one host blocked because the second host cannot keep up momentarily.

6.15.2 Data Flow

The following sections detail the interactions between the Root Port and the Endpoint for reception and transmission of packets. Ordering requirements are derived from these flows as well as the PCIe ordering rules.

6.15.2.1 Pull-Mode Transmit Data Flow

This Tx data flow provides a high-level description of all data/control transformation steps needed for transmission of packets from the Root to the Switch Fabric. Pull mode refers to the mode whereby the descriptors are fetched (pulled) by the FM10000 from the host.

Table 6-6 Tx Pull Data Flow Steps

Step	Description
1	The host maintains a descriptor ring in DRAM. The host configures one of the FM10000 transmit queues with the base address and length of the ring.
2	The host is requested by the software stack to transmit a packet; it gets the packet data within one or more data buffers.
3	The host builds a descriptor(s) that points to the data buffer(s), this includes additional control parameters that describes the needed hardware functionality. An end of packet indication and a descriptor write back indication is included in the final descriptor. The host places this descriptor(s) on the tail of the appropriate TX ring in DRAM.
4	The host updates the appropriate Queue Tail Pointer (TDT) in the FM10000.
5	The FM10000 DMA senses a change of the TDT and as a result sends a PCIe request(s) to fetch the descriptor(s) from host memory. All Descriptor Queues are treated with equal priority by the FM10000.
6	The descriptor(s) content is received in a PCIe read completion and is written to the appropriate TX descriptor queue (TDQ) within the FM10000.
7	Once all descriptors for a given packet are received (descriptor with end of packet indication), the DMA begins reading the descriptors from the TDQ and processing its content. As a result, the DMA sends PCIe requests to fetch the packet data from system memory.
8	As completions return, the data is stored in a TX Packet Buffer (TPB). PCIe requests to memory are only issued while there is available space for all of its completion data in the TPB.
9	The DMA engine continues reading descriptors from the TDQ until it reaches the last descriptor of the packet. As packet data associated with each descriptor is enqueued to the TPB, the Queue Head Pointer (TDH) is incremented (see step 10 for read of last packet descriptor). TDH is incremented in the same sequence as the descriptors are fetched. Only when all packet data is received is the DMA engine free to service another Transmit Descriptor Queue.
10	The host releases descriptor buffers using one of the two mechanisms supported: **DRAM Poll Mode:** On the final descriptor read from the TDQ, if a descriptor write back is required (indicated within the descriptor), the write back is scheduled targeting the current head of the Queue. Once the write back has entered the transmit pipeline, the head pointer is then updated to indicate removal of the final descriptor. The host polling the ring in DRAM can detect the write-backs and thereby determine when and how many descriptors have been removed from the ring. **Interrupt Mode:** Interrupt mode consists of 64 pairs of timer and counters. In SRIOV mode the counters can be assigned per VF, in non-SRIOV mode only one pair is used across all queues. The counter is programmed to indicate a number of transmitted packets. Once the DMA engine has completed processing this number of packets (in total or in SR-IOV mode per VF) it asserts an interrupt for the associated function. Alternatively, if the timer expires, an interrupt is generated. The host uses this interrupt to then read the head pointer from DRAM to determine the number of descriptors removed from the ring. **Head Poll Mode** (Not Supported)**:** In this mode the host simply polls the head pointer from the FM10000 at regular intervals to determine the number of descriptors removed from the ring.
11	Once the entire packet is received it passes through the transmit DMA which performs any programmed manipulations such as checksum offload or TSO offload on the packet.
12	12 The packet is then forwarded to the FM10000's switch interface.

6.15.2.2 Push-Mode Transmit Data Flow

This Tx data flow provides a high-level description of all data/control transformation steps needed for transmission of packets from the Root to the Switch Fabric. Push mode refers to the mode whereby the descriptors are written (pushed) by the host directly into the FM10000 descriptor cache.

Table 6-7 Tx Push Data Flow Steps

Step	Description
1	The host maintains a descriptor ring in DRAM. The host configures one of the FM10000 transmit queues with the base address and length of the ring.
2	The host is requested by the SW stack to transmit a packet. It gets the packet data within one or more data buffers.
3	The host builds a descriptor(s) that points to the data buffer(s), including additional control parameters that describe the needed hardware functionality. An end-of-packet indication and a descriptor write-back indication is included in the final descriptor. The host writes this descriptor(s) on the tail of the appropriate TX ring in DRAM
4	The host typically performs a 64-bits at a time MMIO copy to send the Tx descriptor from host memory to the device internal cache memory in TX_DESC[queue#, desc#, word#]. However the host may write Tx descriptors at any granularity and alignment supported for this mode (refer to Section 6.11). Host is restricted to write the chunks of descriptors into the PEP's internal cache memory - in order.
5	The DMA senses the descriptor(s) write in its internal memory, and begins reading the descriptors from the TX_DESC array and processes its content. Only descriptors with a null DONE bit are processed. As a result, the DMA sends PCIe requests to fetch the packet data from system memory.
6	As completions return, the data is stored in a TX Packet Buffer (TPB). PCIe requests to memory are only issued while there is available space for all of its completion data in the TPB.
7	The DMA engine continues reading descriptors from the TX_DESC until it reaches the last descriptor of the packet. As packet data associated with each descriptor is enqueued to the TPB, the Queue Head Pointer (TDH) is incremented (see bullet 8 for read of last packet descriptor). TDH is incremented in the same sequence as the descriptors are fetched. Only when all packet data is received is the DMA engine free to service another Transmit Descriptor Queue.
8	The host releases descriptor buffers using one of the two mechanisms supported: **DRAM Poll Mode:** On the final descriptor read from the TDQ, if a descriptor write-back is required (indicated within the descriptor), the write-back is scheduled targeting the current head of the Queue. Once the write-back has entered the transmit pipeline, the head pointer is then updated to indicate removal of the final descriptor. The host polling the ring in DRAM can detect the write-backs and thereby determine when and how many descriptors have been removed from the ring. **Interrupt Mode:** Interrupt mode consists of 64 pairs of timers and counters. In SRIOV mode the counters can be assigned per VF, in non-SRIOV mode only one pair is used across all queues. The counter is programmed to indicate a number of transmitted packets. Once the DMA engine has completed processing this number of packets (in total or in SRIOV mode per VF) it asserts an interrupt for the associated function. Alternatively, if the timer expires, an interrupt is generated. The host then uses this interrupt to read the head pointer from DRAM to determine the number of descriptors removed from the ring. **Head Poll Mode** (Not Supported)**:** In this mode the host simply polls the head pointer from the FM10000 at regular intervals to determine the number of descriptors removed from the ring.
9	Once the entire packet is received, it passes through the transmit DMA, which performs any programmed manipulations such as checksum offload.
10	The packet is then forwarded to the FM10000's switch interface.

6.15.2.3 Receive Data Flow

The Rx data flow provides a high-level description of all data/control transformation steps needed for reception of packets from the Switch Fabric to the Root.

Table 6-8 Rx Data Flow Steps

Step	Description
1	The host maintains a descriptor ring in DRAM. The host configures one of the FM10000 receive queues with the base address and length of the ring.
2	The host builds a descriptor(s) that points to an empty data buffer(s). The host places this descriptor(s) on the tail of the appropriate RX ring in DRAM.
3	The host updates the appropriate Queue Tail Pointer (RDT) in the FM10000.
4	The FM10000 detects the update to RDT and starts to pre-fetch descriptors from the Host. Up to 128 descriptors per queue are stored locally in the Rx Descriptor Queue (RDQ), depending on the number of enabled queues. Pre-Fetch for a queue occurs only if there is internal free space for at least 4 descriptors for the queue. All queues are treated with equal priority by the FM10000.
5	A packet enters the the FM10000 Rx pipeline of the PCIe block from the FM10000 Switch Fabric.
6	The packet is parsed by the RX packet processing block. The RX packet processing block determines if the packet needs to be forwarded to the host, and if so, forwards the packet to an RXFIFO.
7	If the packet needs to be forwarded to the host, the RX packet processing block assigns a queue number. This is based on a number of criteria, detailed in Section 6.18.
8	The RX packet processing block then splits the packet into bursts, as required, for transmission to the Host.
9	If a descriptor from the assigned queue is available for the burst, at least one burst is popped from the RXFIFO and scheduled for transmission to the host. As packets are scheduled for transmission to the host, the associated descriptors are popped from the RDQ.
10	A write-back descriptor is built by the FM10000 based on the size and type of the burst(s) sent to the host. The descriptor write-back is issued to the DRAM ring offset indicated by the RDH. The descriptor write back is scheduled to the host based on one of two modes: **Immediate:** A descriptor write back occurs after each pop of a descriptor from the RDQ i.e. packet has already been scheduled for transmission. Once the descriptor write back has been scheduled for transmission to the host the Queue Head Pointer (RDH) is incremented. **Coalesced:** A descriptor write-back occurs each time a cache-line of write-back descriptors has been filled or a time-out value is reached. The descriptor write-back occurs to the DRAM ring offset indicated by RDH. • If no time-out has occurred, and an increment of RDH causes it to cross the current cache line boundary, a descriptor-write back is scheduled to the DRAM ring offset indicated by the RDH. RDH is then incremented, otherwise RDH is incremented as descriptors are taken from the RDQ. • The descriptor coalescing timer starts on detection of a descriptor dequeue from the RDQ, the timer is reset on each descriptor write-back. If the timer has timed out, a write-back is scheduled for transmission to the host to the dram offset location indicated by RDH (the offset in the ring) and the RDH increment then occurs otherwise RDH is incremented as descriptors are taken from the RDQ.
11	The host starts to process the RX Packet and release the RX descriptor buffers using one of the two mechanisms supported: **DRAM Poll Mode:** The host polling the ring in DRAM can detect the final descriptor write-back of a single, or multiple, packets. As the host processes the packets it can release the associated descriptor buffers from the ring. **Interrupt Mode:** The FM10000 can initiate an MSI-X interrupt to the host, for a particular queue, on the final descriptor write-back of a packet. The interrupt triggers the host to start to process packets and thereby release the descriptor buffers from the ring. **Head Poll Mode** (Not Supported)**:** The host simply polls the head pointer from the FM10000 at regular intervals to determine the number of packets that have been posted to DRAM.

6.15.3 Queue Disable Operation

When any specific Queue is disabled the hardware resources associated with that Queue must be freed up. Software quiesces traffic targeting the Queue, disables the Queue and flushes all pending traffic. Once the pipeline has been flushed, all remaining hardware resources associated with the Queue are cleaned up. The following procedures are used to disable Receive or Transmit Queues.

6.15.3.1 Receive Queue Disable

1. Software sets the Queue in drop mode (RXDCTL.*DropOnEmpty* to 1b) to accelerate the switch flush process in next steps.
2. Software requests the Switch Manager to stop mapping traffic to the Queue and to initiate the Flushing Frame Memory procedure for the concerned PC.
   a. Communication with the SM is performed using the mailbox mechanism.
   b. If traffic mapped to the Queue by RSS, Software reprograms the RETA indirect table.
3. Switch Manager notifies Software via mailbox mechanism that the Flushing Frame Memory procedure has completed.
   a. It ensures that no more packet destined to the Queue is be received from the switch.
4. Software waits for a period of three times the DMA_CTRL.*MaxHoldTime*, which ensures the receive buffer in the PEP has been flushed.
   a. There can be at most 4 packets in the receive buffer.
5. Software disables the Queue by writing zero to the RXQCTL[Q].*ENABLE* bit.
6. Software waits until the *ENABLE* bit is read as 0b or for a 100 µs timeout (the first to occurrence).
7. Hardware performs the following within a brief delay:
   a. The Queue is operated internally in *DropOnEmpty* mode (this is useful for a non-graceful Queue Disable flow where Steps 1-4 are skipped).
   b. If an Rx buffer is currently being written, the write is stopped at a TLP boundary. Subsequent data buffers needed for that packet are not written.
   c. Current Rx descriptor write back is completed, including associated ITR, which is logged.
   d. Next packets destined to the Queue are dropped.
   e. Descriptors write back is stopped for the Queue, including logging new interrupts for it. Already logged interrupt for the Queue is be issued to host.
   f. Descriptor fetch is stopped for the Queue.
   g. Track all pending completions for the Queue and drop them on their return.
8. Hardware drops the *ENABLE* bit once all pending completions for the Queue are returned and internal cleanup of Step 7 is completed
9. Software can release the memory resources allocated to the Queue if at least 2 µs elapsed since the *ENABLE* bit was read as 0b. The Queue is ready for re-initialization.
   a. When the *ENABLE* bit is reset to 1b by software, the hardware resets any internal state related to the Queue (including descriptor cache) prior to handling packets for that Queue.

6.15.3.2 Transmit Queue Disable

1. Software stops posting or pushing further descriptors to the Queue.
2. Software waits until the Transmit Descriptor Head (TDH[Q]) equals the Tail (TDT[Q]) which indicates the Queue is empty.
    a. This step is skipped for a non-graceful Tx Queue disable flow.
3. Software disables the Queue by writing zero to the TXDCTL[Q].*ENABLE* bit.
4. Software waits until the *ENABLE* bit is read as 0b or for a 100 µs timeout (the first occurrence).
5. Hardware performs the following:
    a. Current transmission to the switch of a packet from that Queue is completed normally.
    b. Current Tx descriptor write back is completed, including associated ITR, which is logged.
    c. Next packets scheduled for the Queue are dropped.
    d. Descriptor fetch and descriptor write back are stopped for the Queue, including logging new interrupts for it. Already logged interrupt for the Queue is be issued to host.
    e. Track all pending completions for the Queue and drop them on their return.
    f. Reset any internal state related to the Queue (including descriptor cache and data relative to the Queue in the re-order buffer).
6. Hardware drops the *ENABLE* bit once all pending completions for the Queue are returned and internal cleanup of Step 5 is completed.
7. Software can release the memory resources allocated to the Queue.
    a. When the *ENABLE* bit is read as 0b, the Queue is ready for re-initialization.

6.15.4 Performance

Performance numbers reported in this section were observed with the below setup and at a +/-10% accuracy.

6.15.4.1 System Setup

This section describes the system setup used to define performance targets. Performance targets are based on the Grantley and Purley platforms with Broadwell-EP and Skylake-EP Xeon CPUs, under the following assumptions:

- 1 to 2 populated CPU sockets.
- A Gen3 x8 PCIe link.
- Read/Write payload size on the PCIe:
    - Max-Payload-Size: 256 bytes
    - Max-Read-Request-Size: 4 KB
- Loaded latency — CPU supports enough requests to sustain 50 GbE traffic even under load conditions.

- Limitations on placing TLPs on PCIe:
    - — Per PCIe spec for upstream traffic.
    - — Downstream TLPs start only on lane 0 and on even clock symbols.
- Service rate (assume the smallest number of the two):
    - — Broadwell-EP — 125 MB cache lines per second for each of read and write.
    - — Skylake-EP — 250 MB cache lines per second for each of read and write.
- Average system latency (in a non-virtualized environment) is as follows:
    - — Minimal configuration (1 socket system, lightly loaded) — 220 ns.
    - — Maximal configuration (2 socket system, loaded) — 800 ns.
    - — System latency is defined as the latency from first byte of the request on PCIe to first byte of completion (or new credit) on PCIe.
- CPU power/performance Cx states are disabled.
- PCIe credits provided to the end-point (validate both cases) for a Gen3 x8 PCIe link by:
    - — Broadwell-EP root complex:
        - PCIe Posted header/data credits — 88/352
        - PCIe Non-posted header/data credits — 84/12
    - — Skylake-EP root complex:
        - PCIe Posted header/data credits — 120/362
        - PCIe Non-posted header/data credits — 94/16
        - PCIe Completion header/data credits — Infinite
    - — Half amount of credits are provided for a Gen3 x4 PCIe link.

6.15.4.2 Small Packet Throughput

The following setup is assumed:

- Maximal system configuration (determine average latency).
- Up to 64 queues enabled.
- Tail bump rate:
    - — Tx — At least per 8 packets.
    - — Rx — At least per 8 packets.
- Descriptors:
    - — Tx — RS bit is set infrequently (no more than every 2 KB of transmitted data).
    - — Rx — no header split (single buffer per packet), descriptor ring aligned with 4 KB boundaries.

Packet Size	# of Qs	Rx-Only Rx Bit Rate (Gb/s)	Rx-Only Rx Packet Rate (Mp/s)	Tx-Only Tx Bit Rate (Gb/s)	Tx-Only Tx Packet Rate (Mp/s)	Rx/Tx Rx Bit Rate (Gb/s)	Rx/Tx Rx Packet Rate (Mp/s)	Rx/Tx Tx Bit Rate (Gb/s)	Rx/Tx Tx Packet Rate (Mp/s)
64 B	Single	37	54	25	37	34	51	24	36
	Dual	36	53	26	39	30	45	16	24
	3 to 64	37	55	25	37	32	47	19	28
69 B	Single	40	56	27	38	35	49	24	33
	Dual	35	49	23	33	31	43	15	21
	3 to 64	38	53	24	33	31	44	19	27
128 B	Single	43	36	39	33	40	34	37	32
	Dual	43	36	39	33	38	32	26	22
	3 to 64	44	38	39	33	37	31	33	28
133 B	Single	42	34	38	31	38	31	38	31
	Dual	40	33	38	31	36	29	29	23
	3 to 64	43	35	38	31	36	30	31	25
256 B	Single	52	24	48	22	47	21	48	22
	Dual	52	24	48	22	45	20	45	20
	3 to 64	52	24	48	22	45	20	37	17

6.15.4.3 Nominal and Large Packet Throughput

The following setup is assumed:

- Maximal system configuration (determine average latency).
- Up to 256 queues enabled.
- Tail bump rate:
  - Tx — Per 1 packet.
  - Rx — At least per 8 packets.
- Descriptors:
  - Tx — Minimum of 2 descriptors per single packet (one for header, one for data). Assume 4 KB data buffers but packets not aligned to buffer boundaries. For 9 KB jumbos, it means 3 x 4 KB data buffers and one header buffer.
  - Rx — Header split supported. Assume 256-byte header buffers and 2 KB data buffers.

Packet Size	# of Qs	Rx-Only Rx Bit Rate (Gb/s)	Rx-Only Rx Packet Rate (Mp/s)	Tx-Only Tx Bit Rate (Gb/s)	Tx-Only Tx Packet Rate (Mp/s)	Rx/Tx Rx Bit Rate (Gb/s)	Rx/Tx Rx Packet Rate (Mp/s)	Rx/Tx Tx Bit Rate (Gb/s)	Rx/Tx Tx Packet Rate (Mp/s)
512 B	Single	54	13	48	11	46	11	44	10
	Dual	54	13	52	12	45	11	42	10
	3 to 256	54	13	53	12	46	11	38	8.9
1518 B	Single	56	4.5	55	4.4	54	4.4	53	4.3
	Dual	55	4.5	54	4.4	52	4.2	53	4.3
	3 to 256	55	4.5	54	4.4	51	4.2	53	4.3
9 KB	Single	56	0.8	55	0.7	53	0.7	52	0.7
	Dual	56	0.8	55	0.7	54	0.7	52	0.7
	3 to 256	55	0.7	55	0.7	54	0.7	52	0.7

6.15.4.4 Latency (Pull Mode)

The following setup is assumed:

- Minimal system configuration (determine average latency).
- Single connection executing a “ping-pong” test.
- 64-byte packets with CRC/checksum offload enabled.
- Descriptors:
  — Tx — Single descriptor per packet.
  — Rx — No header split (single buffer per packet).
- Immediate interrupt mode.

PEP latency goals:

- Tx — Measured from Tx tail bump to packet delivered to the FM10000 switch fabric interface: 1 μs.
- Rx — Measured from packet arriving on the FM10000 switch fabric interface till packet posted and interrupt issued: 250 ns.

6.15.4.5 Tx Latency in Push Mode

The same setup as for the latency pull mode is assumed, with the single Tx queue operated in push mode.

PEP latency goals:

Tx latency for the push mode queue - measured from first byte of Tx descriptor pushed until packet delivered to the FM10000 switch fabric interface: 580 ns.

Note: When more than a single queue is enabled, best average Tx latency is obtained in push mode with these arbitration settings in DMA_CTRL2: *RxDescPrio* set to 1, *TxDescPrio* to 4, and *TxDataPrio* to 0.

6.15.5 Packet Format

Packets from or to fabric include an 8-byte tag header (called FTAG) before the packet payload. The general packet format is shown in Figure 6-17. All packet header fields are stored in host memory in big endian ordering for both bytes and bits.

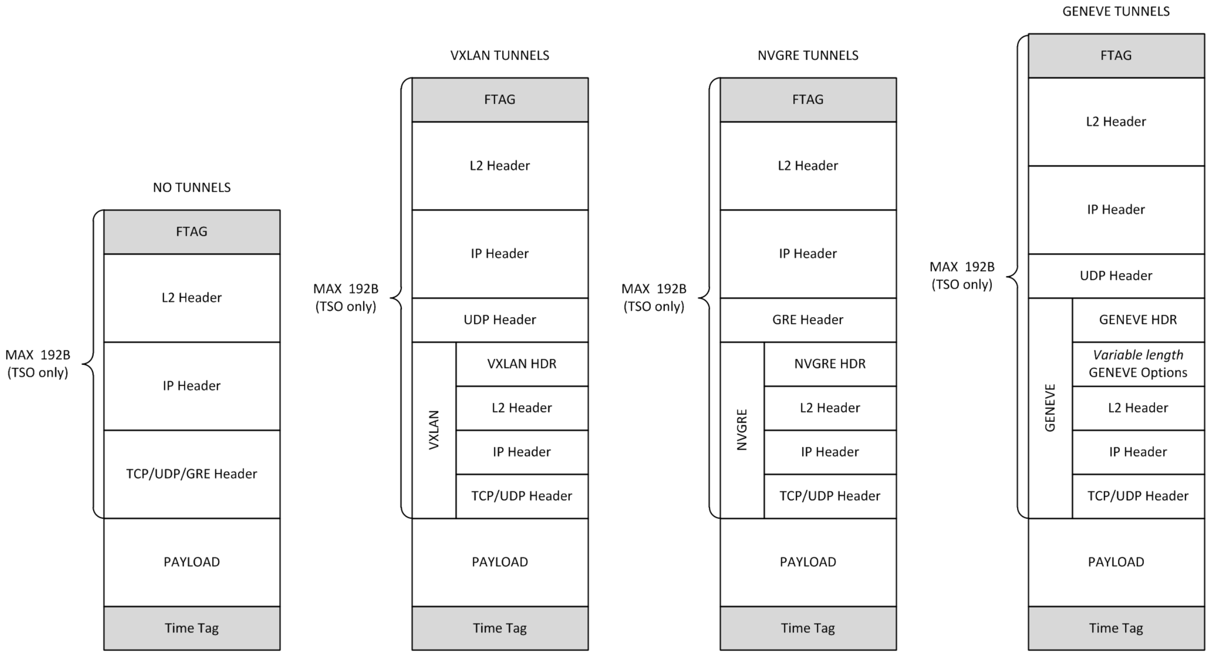


Figure 6-17 PCIe Packet Format

The size of packets transmitted from this unit to the Ethernet switch is at least 72 bytes (including FTAG), transmitter can do padding for frames falling short of this size. In the receive direction, this unit could get packets from switch as small as 48 bytes (including FTAG) and still processed them normally. However, the PCIe unit is prepared to received even smaller packets and, in this case, simply discards them, regardless if the packet was good or bad.

The FTAG header is consumed on Rx and removed from the packet before the packet is stored in memory. On transmission, the FTAG is normally added by the endpoint and not in the packet created by the host, however there is an option to add the FTAG allowing the host to generate specially directed packet.

PAUSE packets have FTAG header and time tags as any other packets. The PAUSE packets are recognized by their DMAC address (01:80:c2:00:00:01). The FTAG header content is ignored on RX for those packets and set to all 0s on TX. The SMAC is also ignored on RX and set to 00:00:00:00:00:01 on TX.

The DMA controller also parse and process the packets in each direction to offload the processor for simple tasks. The DMA controller parser only recognizes the following packets:

- 8-byte FTAG header
  - Consumed on RX, generated on TX.

- L2 header
  - DMAC, SMAC, two tags and then Ethernet Type. The parser supports presence of two tags, the first one is fixed (0x8100, 2 bytes tag, 2 bytes of data) and one configurable (2-byte tag value, data length of 2,4 or 6 bytes, defined in PCIE_EXVET). The relative ordering of the two tags is also configurable. The custom tag could be disabled.
- IPv4 or IPv6 header, with or without options and extension headers - except for GENEVE tunnels, which are always without.
- TCP, UDP for checksum computations.
- Possible three tunnels:
  - **IETF Geneve (GENEVE**) — GENEVE are detected by a match on the L4DST to PCIE_TUNNEL_ID.*NGE_PORT* (matching disabled if *NGE_PORT* is set to 0b). GENEVE packets with version other than 0b or with Next Protocol other than L2 MAC (as specified in PCIE_TUNNEL_ID.*Protocol_Type* field) are handled by the PEP like L3 packets.
  - **NVGRE** — NVGRE are detected by comparing the GRE Protocol Type field to PCIE_TUNNEL_ID.*Protocol_Type* (matching is disabled if it is set to 0b). Presence of KEY is optional, while CHECKSUM, SEQUENCE and ROUTING options are not supported and must not be present.
  - **VXLAN** — VXLAN are detected by a match on the L4DST to PCIE_TUNNEL_ID.*VXLAN_PORT* (matching disabled if *VXLAN_PORT* is set to 0b).

For IPv6, the parser is also capable of detecting the presence of options in the packet by comparing the protocol field to the known options and skipping over the options if they are present. The parser stops after the current header if the protocol in the next header does not match one of the following or on the 6th header (which ever comes first):

- 0x00 = IPv6 Hop-by-Hop Option (length 8N+8 bytes).
- 0x2b = Routing Header for IPv6 (length 8N+8 bytes).
- 0x2c = Fragment Header for IPv6 (length 8 bytes; N is always 0).
- 0x3c = Destination Options for IPv6 (length 8N+8 bytes).
- 0x33 = Authentication Header (length 4N+8 bytes – not the same as the other options).
- 0x87 = Mobility Header (length 8N+8 bytes).

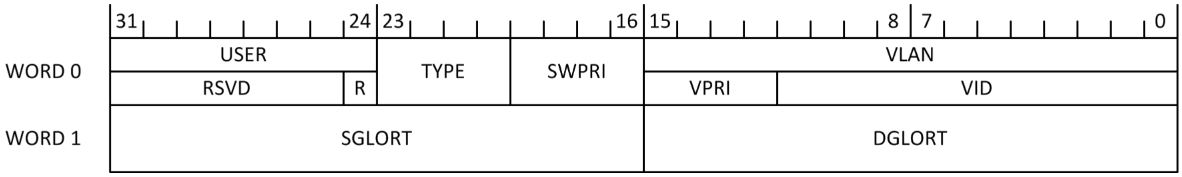
No option headers order is assumed.

The parser also checks if the IPV6 packet is a fragment of a larger packet, and if it is the first fragment. If it is the first fragment, the layer 4 is processed. Otherwise, the parsing stops. If an IPv6 packet is fragmented, it carries a Fragment Header option. If it has a Fragment Header, and the fragment offset (bits 16-28 of the Fragment Header option) is non-zero, this is not the first fragment. Any other packet is passed transparently without any advance processing.

If TSO is used, there is a limit to the size of the header to 192 bytes as the header has to be saved on die for later fragments.

Packets received on Ethernet ports have their CRC replaced by a time stamp. The time stamp corresponds to the moment when the first byte of the preamble was received. If the packet is forwarded to a PEP, the time stamp is removed from the packet and saved in the buffer descriptors. In the transmit direction from PEP to switch, a time stamp command must be specified in the buffer descriptor, which is transferred to USER[1] bit of the FTAG. When the packet egresses the switch, the egress time can be optionally captured and an interrupt posted.

The FTAG format is shown in Figure 6-18:



<table>
<tr><th></th><th colspan="2">31 24</th><th colspan="2">23 16</th><th colspan="2">15 8</th><th>7 0</th></tr>
<tr><td rowspan="2">WORD 0</td><td colspan="2">USER</td><td rowspan="2">TYPE</td><td rowspan="2">SWPRI</td><td colspan="3">VLAN</td></tr>
<tr><td>RSVD</td><td>R</td><td>VPRI</td><td colspan="2">VID</td></tr>
<tr><td>WORD 1</td><td colspan="4">SGLORT</td><td colspan="3">DGLORT</td></tr>
</table>

Figure 6-18 FTAG Format

The VLAN tag is encoded as follows:

- VLAN [11:0] = VLAN ID
- VLAN [15:12] = VLAN PRI

The processing of the FTAG header in each direction is described in the receiver or transmitter sections. Some bits of USER fields have special meaning:

- On receive (from switch to PEP):
  - USER[0] — Indicates if the packet was L3 multicasted and this multicast copy's egress VID is identical to the ingress VID. The PEP may use this information to determine if loopback suppress is to be applied or not. The other USER field bits are ignored.
- On transmit (from PEP to switch):
  - USER[1] — Indicates if time stamping required. Other fields are set to 0b by transmitter.

6.15.6 Switch Fabric Congestion

The switch fabric is a shared memory fabric where individual ports (PCIe interface or Ethernet) are all jointly using the shared memory. The switch contains different parameters to define how this shared memory is partitioned among the different traffic classes and when to issue flow control to link partners.

There are two types of PAUSE frames that could be emitted by the fabric on Ethernet ports; port-based PAUSE or class-based PAUSE frames. In the first case, the traffic is expected to stop for the entire port irrelevant of traffic class, in the second case, only the traffic classes that are defined to stop will stop. The type of PAUSE frames to generate is programmable per Ethernet port in the switch.

The PCIe Interface responds to class-based PAUSE frames, but not to 802.3x PAUSE frames. The PAUSE frame is received by the receiver component of the DMA controller and recognized by comparing the DMAC to the IEEE reserved PAUSE address (01:80:c2:00:00:01). The PAUSE packets are not sent to the host, they are parsed and decoded and filtered out. If parsing fails, the packet is discarded and content ignored. If parsing is completed successfully, the parser forwards the pause information to the transmitter, one value per class (up to 8 class).

The transmitter includes a map from queue to flow control class (256 x 3-bit map) which is used to pause the corresponding queue or not. The transmitter stops recovering packets from host memory if a queue is paused and resume recovering packets from host when a queue is not paused, any requests to host are considered committed and are transmitted to the switch. The worst case reaction time of the PEP to a pause frame received from the switch is bounded to two Jumbo packets (30 KB) plus the pause frame parsing time.

The transmitter also includes a hard-coded 1 ms timer (2 ms for GEN2 and 4 ms for GEN1) per traffic class which is started whenever a corresponding Priority Flow Control (PFC) PAUSE packet is received from the switch. If the timer expires, the corresponding pause is cleared.

6.15.7 Host Congestion

Host congestion could occur when the CPU cannot keep up with packet reception from fabric. This could be very temporary at any time (up to few microseconds) or could be longer if queues become empty. The temporary hold could be due to the time to fetch a descriptor (if pre-fetched descriptors ran out) or extra cycles on parsing or any other temporary extra processing. The longer hold is when the queue is empty and waiting for CPU to provide new descriptors.

If the queue becomes completely empty and a packet arrives for which there are no descriptor available, the DMA controller is programmed to either drop (and count) the packet or hold onto the packet until descriptors are available or a time out. Each queue is programmable to ether hold or drop when there are no descriptors available.

A hardware timer is also implemented to automatically change the queue setting to the drop mode and to issue an interrupt to the PF or VF if the packet is held for too long. This is not required as long as the driver is operational, but kicks in as a last resort if the host is non-functional, to free up memory in the fabric.

6.15.8 TLP Processing Hints

The FM10000 supports the TPH capability defined in the PCIe specification in the No Steering mode only. It does not support Extended TPH requests. TPH is disabled by the Intel default Serial Boot NVM image, as it provides no benefits for most root complexes, and contains a non-compliance issue where a read-only field is actually read-write in the device.

On the PCIe link, existence of a TLP Process Hint (TPH) is indicated by setting the TH bit in the TLP header.

6.15.9 Host Response Time Measurements

The performance of the PCIe DMA controller depends on the host response time to DMA read requests. The read completion response time measurement unit (RRTIME) is designed to help tracking host response time.

The unit keeps a local 16-bit counter used as a time reference, this counter is incremented on every clock cycle (every 2 ns for GEN3, 4 ns for GEN2, 8 ns for GEN1). Each read request issued by the controller is time tagged with this time reference at the moment the read request is pushed into the PCIe controller, the time tag is saved in the DMA controller. At the read response, the DMA controller collects the time at the moment the last word of the last read completion for this transaction is received, it then subtracts the number of 16-byte chunks received thus correcting for the transaction length, subtracts the original time the transaction was issued, and then send the time different along with the PF or VF, the queue number and the type of transaction to the time collector. The type of transaction is either:

- RX descriptor read.
- TX descriptor read.
- TX data read.

The time collector unit applies a user-programmable filter to keep only events of interest, there are two orthogonal filters:

- Filter for PF, specific VF, specific queue or any.
- Filter for any transactions

The register PCIE_RRTIME_CFG defines the filters.

The time collector compares the delta time to configurable bins and counts events on each bin. The bins are configured in the PCIE_RRTIME_LIMIT[0..2] registers, while the PCIE_RRTIME_COUNT[0..3] registers are the bins.

```
rrt = read response time delta in clocks
if ( event match PCIE_RRTIME_CFG)
  if ( rrt <= PCIE_RRTIME_LIMIT[0] )
     PCIE_RRTIME_COUNT[0]++
  else if ( rrt <= PCIE_RRTIME_LIMIT[1] )
     PCIE_RRTIME_COUNT[1]++
  else if ( rrt <= PCIE_RRTIME_LIMIT[2] )
     PCIE_RRTIME_COUNT[2]++
  else
     PCIE_RRTIME_COUNT[3]++
```

Software can track performance problems by watching specific queues and specific transactions.

6.15.10 Packet Loopback

The register PCIE_CTRL_EXT.*SwitchLoopback* controls a switch loopback. When set, packets sent to the switch are looped back to PEP, and packets sent by the switch are looped back to the switch. Set to 0b for normal operation.

6.16 Initialization After Reset

The following sequence can be used to bring up the PEP for normal operation after reset is de-asserted. The initial steps of link bring-up can be handled by the Boot State Machine (BSM), or by the Switch Manager (SM) if it is available. Once the link is up, subsequent steps can be handled by the host driver.

Variables used are:

- FS = 24 (Full Swing value for Gen3 PCIe transmit equalization)
- LF = 8 (Low Frequency value for Gen3 PCIe transmit equalization)
- NL = 8/4/2/1 (number of lanes used for this PEP; must be <= 4 if DEVICE_CFG.*PCIeMode* = MODE_2x4 for this PEP, and must be 1 for PCIE_MGMT)

Access to the port logic registers of the PCIe core is based at the PCIE_PORTLOGIC address. For purposes of the code below, PCIE_PORTLOGIC is regarded as a 600 x 32-bit RW register array.

1. BSM or SM sets the link width, in multiple places:
   - PCIE_PORTLOGIC[i,0x10C/4][12:8] = NL; (Link Width and Speed Change reg)
   - PCIE_PORTLOGIC[i,0x10C/4][16] = 1; (Enable Auto-Flip of the lanes)
   - PCIE_PORTLOGIC[i,0x010/4][21:16] = (2*NL-1); (Port Link Control reg)

- Set PCIE_CFG_PCIE_LINK_CAP[i].*MaxLinkWidth* = NL;
- Set PCIE_HOST_LANE_CTRL[i].*MasterLane*, based on NL and link reversal

2. BSM or SM sets the PCIe Gen3 equalization control register:
   - PCIE_PORTLOGIC[i,0x1A8/4][5] = 1; (Phase2_3 2ms Timeout Disable field)
   - PCIE_PORTLOGIC[i,0x1A8/4][23:8] = 1<<4; (Preset Request Vector field: request only preset P4)
   - PCIE_PORTLOGIC[i,0x1A8/4][24] = 0; (Include Initial FOM field)
   - PCIE_PORTLOGIC[I,0x190/4][10] = 1; (Disable requesting reset of EIEOS count during equalization)
3. BSM or SM programs the PCIe Gen3 equalization coefficients:

```
int[] PREC = {0, 0, 0, 0, 0, 10, 8, 10, 8, 6, 0};
int[] POST = {4, 6, 5, 8, 0, 0, 0, 5, 8, 0, 2};

int precursor(int preset) {
    return (PREC[preset] == 0) ? 0 : round(FS/PREC[preset]);
}
int postcursor(int preset) {
    if (preset < 10) {
        return (POST[preset] == 0) ? 0 : round(FS/POST[preset]);
    } else {
        return round((FS-LF)/2);
    }
}
int cursor(int preset) {
    return FS - precursor(preset) - postcursor(preset);
}

for preset = 0..10 {
    PCIE_PORTLOGIC[i,0x19C/4] = preset;
    PCIE_PORTLOGIC[i,0x198/4] = postcursor(preset)<<12 | cursor(preset)<<6 | precursor(preset);
}
PCIE_PORTLOGIC[i,0x194/4] = FS<<6 | LF;
```

4. BSM or SM sets PCIE_HOST_LANE_CTRL[i].IntDisableAutoAssert = 1b
5. BSM or SM allows link initialization to begin. Once the link is up, most of the following steps can be handled by the PF host driver.
   - sbus_pcie_command(0xFF, 0x03, Write, 0x0026_5202); // run iCal on second equalization request.
   - Set PCIE_CTRL[i].LTSSM_ENABLE = 1b
6. PF driver enables interrupts to SM and BSM.
   - Set PCIE_IM, PCIE_IB, PCIE_SRAM_IM to direct interrupts appropriately.
7. PF driver disables PEP's Tx queues in network stack.
8. PF driver disables PEP's Rx filters by requesting the SM via mailbox to stop mapping traffic to the PEP.
9. PF driver disables interrupts to local host.
   - Write PCIE_EIMR with all 1s.
10. PF driver resets the PEP.
    - Issue a *DataPathReset* reset and wait until the bit is read as 0b before going to step 8.
    - In certain cases, issue a PCIe in-band reset.

11. On PCIe reset or WARM reset events, or on transition from D3hot to D0u, BSM loads the following PEP settings from EEPROM:
    - INTERRUPT_MASK_BSM.*PCIE_BSM*[PEP#] bit to be cleared when booting from EEPROM/BSM, and perhaps other mask bits for other boot methods.
    - PCIE_CFG_SPD_NUMBER
    - PCIE_CFG_SRIOV_NUM.*NumVFs*
    - PCIE_CFG_SRIOV_CFG.*InitialVFs*
    - PCIE_CFG_2.*MultiFunction* bit could be required to be set to 1b if SR-IOV is enabled (which is the default) in spite it is not in line with PCIE compliance testing.
    - PCIE_CFG_SRIOV_HDR.*Nextpointer* must be set to 0x1C8 to expose TPH capability or to 0x1D8 to expose ACS capability (if TPH capability not exposed), or to 0x000 otherwise.
    - PCIE_CFG_TPH_HDR.*Nextpointer* must be set to 0x1D8 to expose ACS capability (if TPH capability is exposed)
    - PCIE_CFG_SRIOV_DEVID
    - PCIE_CFG_SUBID, PCIE_VF_CFG_SUBID[0..63]
    - PCIE_CFG_PCIE_DEV_CAP, PCIE_VF_CFG_PCIE_DEV_CAP[0..63]
    - PCIE_CFG_PCIE_LINK_CAP, PCIE_VF_CFG_PCIE_LINK_CAP[0..63]
    - PCIE_CFG_PCIE_LINK_CTRL, PCIE_CFG_PCIE_LINK_CTRL_STAT[0..63]
    - PCIE_CFG_CMD, PCIE_VF_CFG_CMD[0..63]
    - PCIE_SM_AREA
    - PCIE_GCR.*ComplTimeoutResendEn*
    - PCIE_VF_CFG_ARI_CAP.*NextCapPtr* must be set to 0x158 to expose TPH capability or to 0x1D8 to expose ACS capability (if TPH capability not exposed), or to 0x000 otherwise.
    - PCIE_VF_CFG_TPH_HDR.*NextCapPtr* must be set to 0x1D8 to expose ACS capability (if TPH capability is exposed)
    - PCIE_VF_CFG_MSIX_CAP[0..63].*Tablesize*
    - PCIE_CTRL.*BAR4_Allowed*
12. On exit from PCIe reset or WARM reset or on transition from D3hot to D0u, BIOS/Kernel waits until the device enters D0u state before it eventually changes some PCIe configuration space settings, and then invokes the PF driver.
    - PCIE_CFG_SRIOV_PAGE_CF
    - PCIE_CFG_TPH_CTRL
    - PCIE_CFG_PCIE_DEV_CTRL, PCIE_VF_CFG_PCIE_DEV_CTRL[0..63]
    - PCIE_CFG_MSIX_CAP, PCIE_VF_CFG_MSIX_CAP[0..63]
13. PF driver does general PEP settings.
    - PCIE_CTRL_EX
    - PCIE_EXVET
    - PCIE_TUNNEL_CFG
    - PCIE_DMA_CTRL2

- PCIE_DTXTCPFLGL
- PCIE_DTXTCPFLGH
- PCIE_TPH_CTRL
- PCIE_VLAN_TABLE[0..4095]

14. PF driver initializes virtualization support.
    - PCIE_TQMAP[0..2047]
    - PCIE_RQMAP[0..2047]
    - PCIE_TXQCTL[0..255] to map transmit queues to VF/PF
    - PCIE_RXQCTL[0..255] to map receive queues to VF/PF
15. PF driver configures TCs.
    - PCIE_SWPRI_MAP[0..15]
    - PCIE_TC_CREDIT[0..63]
    - PCIE_TC_MAXCREDIT[0..63]
    - PCIE_TC_RATE[0..63]
16. PF driver maps interrupts to queues and enables interrupts.
    a. PCIE_INT_MAP[0..15], PCIE_VFINT_MAP[0..15]
        - SRAM interrupt is mapped to vector 0.

    b. PCIE_MSIX_VECTOR[0..767]
        - It may that MSI-X table is configured by the OS via BAR2 and prior to the driver setting PCIE_INT_MAP registers.

    c. PCIE_RXINT[0..255], PCIE_VFRXQCTL[0..255], PCIE_TXINT[0..255], PCIE_VFTXINT[0..255]

    d. PCIE_ITR[0..767], PCIE_VFITR[0..255], PCIE_ITR2[0..767]

    e. PCIE_PBACL[0..71], PCIE_VFPBACL[0..7]

    f. PCIE_PFVFLREC with all 1s, to clear (old) VFLR events for all VFs.

    g. PCIE_MBMEM[0..1023] (starting from 1023 down to 0) with all 0s, to clear the VF mailbox memory for all VFs.

    h. PCIE_INT_CTRL
17. PF driver starts Rx/Tx DMA engines.
    - PCIE_DMA_CTRL – Write TxEnable and RxEnable with 1 to start Tx/Rx data pipes.
18. PF/VF driver initializes transmit.
    a. PCIE_TQDLOC[0..255]

    b. PCIE_TX_SGLORT[0..255]

    c. PCIE_PFVTCTL[0..255]

    d. PCIE_TPH_TXCTRL[0..255]

    e. PCIE_TDBAL[0..255], PCIE_TDBAH[0..255], PCIE_VFTDBAL[0..255], PCIE_VFTDBAH[0..255]

    f. PCIE_TDLEN[0..255], PCIE_VFTDLEN[0..255]

  g. PCIE_TDH[0..255], PCIE_TDT[0..255] to reset the heads and the tails to 0.
  h. PCIE_TXQCTL[0..255], PCIE_VFTXQCTL[0..255]
  i. PCIE_TXDCTL[0..255], PCIE_VFTXDCTL[0..255] to enable transmit queues.

19. PF/VF drivers initializes receive.
    a. PCIE_RXDCTL[0..255]
    b. PCIE_TPH_RXCTRL[0..255]
    c. PCIE_DGLORTMAP[0..7], PCIE_DGORTDEC[0..7]
    d. PCIE_MRQC[0..64]
    e. PCIE_RX_SGLORT[0..255]
    f. PCIE_RSSRK[0..64][0..9], PCIE_VFRSSRK[0..9], PCIE_RETA-0..64][0..9], PCIE_VFRETA[0..31]
    g. PCIE_RDBAL[0..255], PCIE_RDBAH[0..255], PCIE_VFRDBAL[0..255], PCIE_VFRDBAL[0..255]
    h. PCIE_RDLEN[0..255], PCIE_VFRDLEN[0..255]
    i. PCIE_SRRCTL[0..255], PCIE_VFSRRCTL[0..255]
    j. PCIE_RXQCTL[0..255], PCIE_VFRXQCTL[0..255] to enable receive queue.
    k. PCIE_RDT[0..255], PCIE_VFRDT[0..255] for tails bump once receive queues have been filed with valid descriptors.
        - This step is done just after enabling each queue.

20. PF driver enables PEP's Rx filters by requesting the SM via mailbox to (re-)map traffic to the PEP.

21. PF/VF driver enables Tx queues in network stack and checks that PCIE_TXDCTL.*ENABLE* bit is read as 1b.
    - PCIE_TDT[0..255], PCIE_VFTDT[0..255] for tails bump once transmit queues have been filed with valid descriptors.
    - In push mode, descriptors are pushed directly into PCIE_TX_DESC array.

22. SM notifies the PF driver via mailbox that PEP's Rx filters are (re-)enabled.

Note: All registers that get their default setting from EEPROM (i.e. registers listed in steps 11 and 12) do not reset on *DataPathReset* events.

Note: Table entries and registers mapped to memories may not reset on WARM reset or *DataPathReset* events, and therefore are reconfigured by the driver according to the flow above.

6.17 VPD Handling

VPD access should be handled by the interrupt handler in the Boot State Machine. Here is a suggested flow for VPD access handling:

1. On boot, the BSM loads the VPD images for each of the 9 PEPs into the on-chip scratch memory BSM_SCRATCH.
2. The PCIE_IB[i].*VPD_Request* and INTERRUPT_MASK_BSM.*PCIE_BSM*[i] interrupt mask bits are cleared, to allow a *VPD_Request* interrupt to wake up the BSM.
3. When a VPD request is made over PCIe, the BSM is woken up by the PCIE_IP.*VPD_Request* interrupt. The BSM starts executing from serial ROM, starting at the address configured in BSM_ARGS. The remaining steps of the flow are under the control of code loaded from the serial ROM.
4. The BSM reads the GLOBAL_INTERRUPT_DETECT register and sees that the PCIE_BSM[i] bit is set.
5. The BSM reads PCIE_IP from PEP #i and sees the *VPD_Request* bit set.
6. The BSM reads the PCIE_CFG_VPD_CAP (VPD address) register from PEP #i, to get the address and the read/write flag.
7. On a read, the BSM copies a word from the VPD image in BSM_SCRATCH to the PCIE_CFG_VPD_DATA register. On a write, if the address is a writable VPD area, it copies from PCIE_CFG_VPD_DATA into the VPD image.
8. The BSM clears the PCIE_IP.*VPD_Request* interrupt bit.
9. The BSM writes back to the PCIE_CFG_VPD_CAP register to invert the flag.
10. If writable VPD is supported, the BSM cannot commit changes back to the EEPROM or flash directly. Instead the switch manager must be signaled (e.g. using SW_IP.*SoftIP* to interrupt the SM) to indicate that a VPD write needs to be committed. To commit the write to flash, SM should disable BSM, then use the SPI controller or some other mechanism to write back the new VPD image into the flash.

6.18 Packet Reception

6.18.1 Overview

Packet reception is from Fabric to Host. Figure 6-19 shows the overall packet reception process.

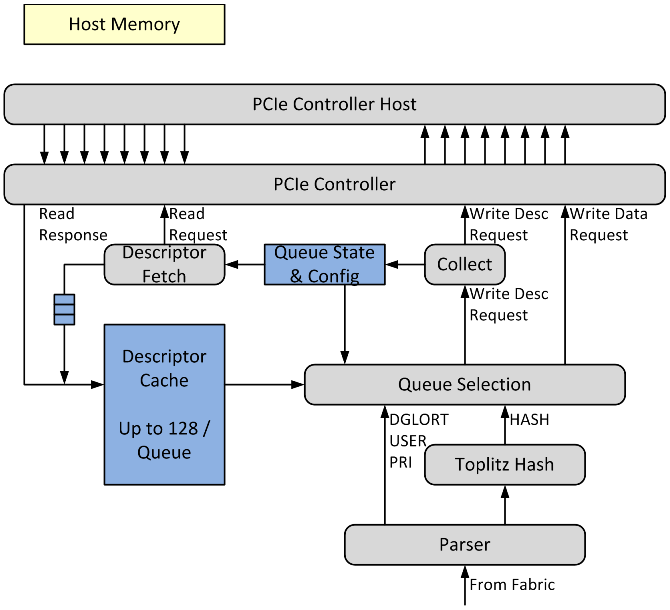


Figure 6-19 PCIe Receiver Block Diagram

The packets reception includes the following elements:

Table 6-9 Packet Reception Elements

Element	Description
Parser	Parse the packet content to extract the different fields.
Hash	Compute a hash key for the packet using fields extracted from the packet.
Queue Selection	Determine which queue to select based on parsing and hashing results.
Queue Status and Config	Stores all queues status and configuration.
RX Descriptor Cache	Stores descriptors retrieved ahead of time.
Prefetch Descriptor	Retrieves descriptors ahead of time.
Collect	Collects multiple descriptors together before posting interrupts.

6.18.2 Parsing

The parser performs the following functions:

- Determine the VF to which the packet belongs.
- Extract critical fields for Receive Side Scaling.
- Compute checksum.
- Determine packet classification.
- Pass PAUSE packets information to TX side.

The packet classification is normally done by the fabric using resources in the frame processing pipeline. The fabric passes the result of the classification to the PCIe core via the FTAG.

The FTAG contains the fields listed in Table 6-10.

Table 6-10 FTAG Fields

Field	Description
FTAG.{VLAN,VPRI}	Simply passed to the receiver packet descriptor and not used for queue assignment.
FTAG.SWPRI	Identifies the priority of this packet. The DMA controller includes a SWPRI-to-PC map, which defines how packets get mapped to internal queues.
FTAG.SGLORT	Identifies the original logical port that received the packet. This could be another PCIe port or an Ethernet port. The SGLORT is not used for queue assignment but is used in loopback suppress function.
FTAG.DGLORT	Identifies the logical destination port and represents the main mechanism by which the packet gets assigned to a queue.
FTAG.USER	Contains various indications from the switch: USER[0] = (SWITCHED) Set if this packet was L3 multicasted and the egress VID equals the ingress VID for this multicast copy (1b) as opposed to routed to another VLAN (0b).

The FTAG.USER The FTAG is stripped from the packet and not written into the data buffer to host memory, however the SGLORT, DGLORT and VLAN/VPRI are copied into the RX descriptor.

If a packet is smaller than 48 bytes, the packet is simply flushed.

Rx parser stops parsing the packet header after 1 KB, which means RSS, header split, and other receive functionalities are either performed according to the partial header captured, or not performed at all (e.g. checksum).

6.18.3 Queue Assignment

The queue assignment uses DGLORT as a primary driver, followed by extracted switch priority and RSS hash computation to determine which queue is used for the packet received.

The PCIe contains eight DGLORT decoders (PCIE_DGLORTMAP[0..7] and PCIE_DGLORTDEC[0..7]) that define how the DGLORT is decoded. The DGLORT decoder includes a pattern match to detect which decode is active ({value,mask} set). At least one of the eight decoders will match. Otherwise, the packet is assigned to queue 0 if there is no match. If there is more than one match, the highest DGLORT decoder is retained. Each decoder then defines how the queue is selected.

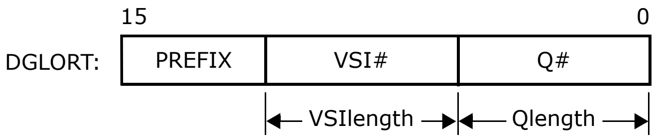
The fields in each DGLORT mapper are:

- PCIE_DGLORTMAP[0..7].*Value* (16 bits)
- PCIE_DGLORTMAP[0..7].*Mask* (16 bits)

The decoding is thus:

```
dglortIdx = -1;
foreach i (0..7)
    value = PCIE_DGLORTMAP\[i\].Value;
    mask = PCIE_DGLORTMAP\[i\].Mask;
    if ( (dglort & mask) == value )
    {
        dglortIdx = i;
    }
```

The PCIE_DGLORTDEC is then used to determine the queue selection method. The DGLORT is normally interpreted as follow:



Where VSI# is the VSI number relative to a base defined in the PCIE_DGLORTDEC and Q# is relative to a based queue defined also in the PCIE_DGLORTDEC. The Q# could be a specific queue or a queue pool. Furthermore, the PCIE_DGLORTDEC defines also the Priority Class length, and the RSS hash length.

The final queue number is computed this way:

```
// Extract fields from GLORT decoder register configuration
qLength = PCIE_DGLORTDEC[dglortIdx].Qlength;
vsiLength = PCIE_DGLORTDEC[dglortIdx].VSIlength;
vsiBase = PCIE_DGLORTDEC[dglortIdx].VSIbase;
qBase = PCIE_DGLORTDEC[dglortIdx].Qbase;
pcLength = PCIE_DGLORTDEC[dglortIdx].PClength;
rssLength = PCIE_DGLORTDEC[dglortIdx].RSSlength;

// Derive priority class from switch priority
pc = PCIE_SWPRI_MAP[FTAG.SWPRI].pc;

// Extra Q# and VSI# from DGLORT received
qIn = DGLORT & ((1<<qLength)-1);
vsiIn = (DGLORT >> qLength) & ((1<<vsiLength)-1);

// RSS computation
t = vsiIn+vsiBase;                 // Select table
key[0..9] = PCIE_RSSRK[t][0..9];   // Select secret key
hashCfg = PCIE_MRQC[t];            // Select compute options
h = hash(packet,key,hashCfg);      // Compute hash

// Use redirection table to recover queue redirection
qRedirect = RETA[t][h & 127] & ((1<<rssLength)-1);

// Compute final Q
Q = (vsiIn<<(pcLength+rssLength+qLength)) +
    (qIn<<(pcLength+rssLength)) +
    (qRedirect<<pcLength) +
    (pc & ((1<<pcLength)-1)) + qBase

// Recover final VF & PF
VF = PCIE_RXQCTL[Q].VF;
PF = ~PCIE_RXQCTL[Q].OwnedByVF;
```

The packet is sent to queue Q, which is defined to belong to a specific VF or a PF.

The function RSS redirection is detailed in the next section. The secret key, RSS field selection and RETA used depend on the VF/PF derived from the DGLORT (and not from the final Q which is derived possibly from VF).

6.18.4 Receive Side Scaling

RSS is a mechanism defined by Microsoft to distribute received packets into several descriptor queues. Software then assigns each queue to a different processor, therefore sharing the load of packet processing among several processors.

The FM10000 supports RSS by computing a Toeplitz hash on received packet, and uses the lower 7-bit result to index a redirection table which returns an RSS index (one redirection table per “vfIn”). The new RSS index is then used for queue assignment (refer to Section 6.18.3).

RSS can be enabled or disabled on the fly, provided that it is certain the queue is not being accessed during the change.

6.18.4.1 RSS Hash Function

A single hash function is defined with several variations for the following cases:

- **TcpIPv4** — The FM10000 parses the packet to identify an IPv4 packet containing a TCP segment per the following criteria. If the packet is not an IPv4 packet containing a TCP segment, RSS is not done for the packet.
- **IPv4** — The FM10000 parses the packet to identify an IPv4 packet. If the packet is not an IPv4 packet, RSS is not done for the packet.
- **TcpIPv6** — The FM10000 parses the packet to identify an IPv6 packet containing a TCP segment per the following criteria. If the packet is not an IPv6 packet containing a TCP segment, RSS is not done for the packet.
- **IPv6** — The FM10000 parses the packet to identify an IPv6 packet. If the packet is not an IPv6 packet, RSS is not done for the packet.
- **UdpIPV4** — The FM10000 parses the packet to identify a packet with UDP over IPv4.
- **UdpIPV6** — The FM10000 parses the packet to identify a packet with UDP over IPv6.

RSS hash computation could use either the outer header only or use the inner header if found present. This is controlled by the DGLORT_DEC[i].*InnerRSS* bit.

IPv6 extensions should be parsed for a Home-Address-Option or for a Routing-Header (with Segments Left > 0). If found, RSS is not done for the packet.

A packet is identified as containing a TCP segment if all of the following conditions are met:

- The transport layer protocol is TCP (not UDP, ICMP, IGMP, etc.).
- The TCP segment can be parsed (such as IP options can be parsed, packet not encrypted, etc.).
- The packet is not fragmented (even if the fragment contains a complete L4 header).

The bitmap in the Multiple Receive Queues Command register (PCIE_MRQC) enables each of the above hash function variations (several might be set at a given time). If several functions are enabled at the same time, priority is defined as follows (skipping functions that are not enabled):

IPv4 packet:

- Try using the TcpIPv4 function.
- Try using the UdpIPv4 function.
- Try using the IPv4 function.

IPv6 packet:

- Try using the TcpIPv6 function.
- Try using the UdpIPv6 function.
- Try using the IPv6 function.

The following combinations are currently supported:

- Any combination of IPv4, TcpIPv4, and UdpIPv4.

and/or:

- Any combination of either IPv6, TcpIPv6, and UdpIPv6.

The 32-bit result of the hash computation is written into the packet descriptor and also provides an index into the redirection table. When a packet cannot be parsed by the previous rules, it is assigned an RSS output index of zero and a hash result of 0.

The following notation is used to describe the following hash functions:

- Ordering is little endian in both bytes and bits. For example, the IP address 161.142.100.80 translates into 0xa18e6450 in the signature.
- A "^" denotes bit-wise XOR operation of same-width vectors.
- @x-y denotes bytes x through y (including both of them) of the incoming packet, where byte 0 is the first byte of the IP header. In other words, we consider all byte-offsets as offsets into a packet where the framing layer header has been stripped out. Therefore, the source IPv4 address is referred to as @12-15, while the destination v4 address is referred to as @16-19.
- @x-y, @v-w denotes concatenation of bytes x-y, followed by bytes v-w, preserving the order in which they occurred in the packet.

All hash function variations (IPv4 and IPv6) follow the same general structure. Specific details for each variation are described in the following section. The hash uses a random secret key of length 320 bits (40 bytes); the key is stored in the RSS Random Key Register (RSSRK).

The algorithm works by examining each bit of the hash input from left to right. Our nomenclature defines left and right for a byte-array as follows: Given an array B with b bytes, our nomenclature assumes that the array is laid out as follows:

- B[0],B[1],B[2],…,B[b-1]

B[0] is the left-most byte, and the MSB of B[0] is the left-most bit. B[b-1] is the right-most byte, and the LSB of B[b-1] is the right-most bit.

```
ComputeHash(input[], N)
    K = secret key of 320 bits;
    Result = 0;
    For each bit b in input[]
    {
        if (b == 1) then Result ^= (left-most 32 bits of K);
        shift K left 1 bit position;
    }
    return Result;
```

6.18.4.1.1 Pseudo-Code Examples

Example: Hash Calculation for IPv4 with the TCP Header

Concatenate the SourceAddress, DestinationAddress, SourcePort, and DestinationPort packet fields into a byte array, preserving the order in which they occurred in the packet:

```
Input[12] = @12-15, @16-19, @20-21, @22-23
Result = ComputeHash(Input, 12)
```

Example: Hash Calculation for IPv4 Only

Concatenate the SourceAddress and DestinationAddress packet fields into a byte array:

```
Input[8] = @12-15, @16-19
Result = ComputeHash(Input, 8)
```

Example: Hash Calculation for IPv6 with the TCP Header

Concatenate the SourceAddress, DestinationAddress, SourcePort, and DestinationPort packet fields into a byte array, preserving the order in which they occurred in the packet:

```
Input[36] = @8-23, @24-39, @40-41, @42-43
Result = ComputeHash(Input, 36)
```

Example: Hash Calculation for IPv6 Only

Concatenate the SourceAddress and DestinationAddress packet fields into a byte array:

```
Input[32] = @8-23, @24-39
Result = ComputeHash(Input, 32)
```

6.18.4.1.2 RSS Verification Suite

The following example provides a verification test suite. Assume that the random key byte-stream is:

```
0x6d, 0x5a, 0x56, 0xda, 0x25, 0x5b, 0x0e, 0xc2,
0x41, 0x67, 0x25, 0x3d, 0x43, 0xa3, 0x8f, 0xb0,
0xd0, 0xca, 0x2b, 0xcb, 0xae, 0x7b, 0x30, 0xb4,
0x77, 0xcb, 0x2d, 0xa3, 0x80, 0x30, 0xf2, 0x0c,
0x6a, 0x42, 0xb7, 0x3b, 0xbe, 0xac, 0x01, 0xfa
```

The results are:

Table 6-11 IPv4

Destination Address/Port	Source Address/Port	IPv4 only	IPv4 with TCP
161.142.100.80:1766	66.9.149.187:2794	0x323e8fc2	0x51ccc178
65.69.140.83:4739	199.92.111.2:14230	0xd718262a	0xc626b0ea
12.22.207.184:38024	24.19.198.95:12898	0xd2d0a5de	0x5c2b394a
209.142.163.6:2217	38.27.205.30:48228	0x82989176	0xafc7327f
202.188.127.2:1303	153.39.163.191:44251	0x5d1809c5	0x10e828a2

Table 6-12 IPv6

Destination Address/Port	Source Address/Port	IPv6 only	IPv6 with TCP
3ffe:2501:200:1fff::7(1766)	3ffe:2501:200:3::1(2794)	0x2cc18cd5	0x40207d3d
ff02::1(4739)	3ffe:501:8::260:97ff:fe40:efab(14230)	0x0f0c461c	0xdde51bbf
fe80::200:f8ff:fe21:67cf(38024)	3ffe:1900:4545:3:200:f8ff:fe21:67cf(44251)	0x4b61e985	0x02d1feef

6.18.5 VLAN Membership Checking

All packets received must be checked for VLAN membership. The VLAN ID recovered from the packet's FTAG.VLAN and the VSI number computed during queue assignment are used to index the PCIE_VLAN_TABLE.

```
VID = FTAG.VLAN[11:0]

VSI = vsiIn+vsiBase
 member = PCIE_VLAN_TABLE[VSI][(VID div 32)].VID_Membership[(VID mod 32)]
if (member == 0)
{
   DropPacket()
   PCIE_STATS_VLAN_DROPS++
}
```

6.18.6 Loopback Suppress

All packets received must be checked for loopback suppression to prevent a packet from a given VM to return to the same VM. This is done by verifying if the SGLORT received with a packet matches the SGLORT assigned to the Q that the packet is going to, and only done for non-routed packets.

```
if ((FTAG.USER[0] ==1) && (PCIE_SRRCTL.loopbackSuppress == 1))
{
    if ( FTAG.SGLORT == PCIE_RX_SGLORT\[q\] )
        DropPacket()
        PCIE_STATS_LOOPBACK_DROPS++
}
```

6.18.7 Receive Descriptors

The size of the receive descriptors is either 32 bytes. The format is different on read and write back. The read descriptors format is shown in Figure 6-20.

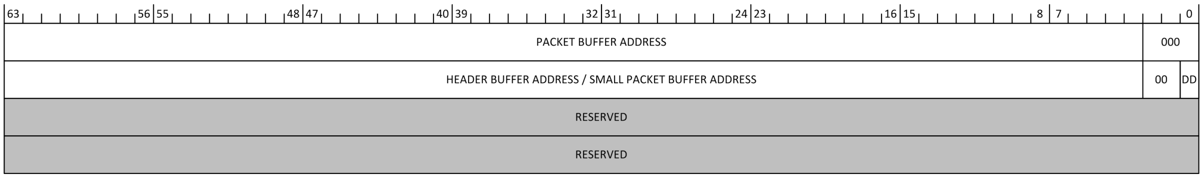


Figure 6-20 Receive Descriptor Read (32 Bytes)

The buffer addresses are either 512-byte aligned, or 8-byte aligned but without crossing host memory pages (4KB alignment boundaries). The lowest bit of the header/small-packet address is used to indicate if the buffer is available (Data *Done* bit). It is cleared by software and set by hardware.

The receive descriptor write back formats is shown in Figure 6-21.

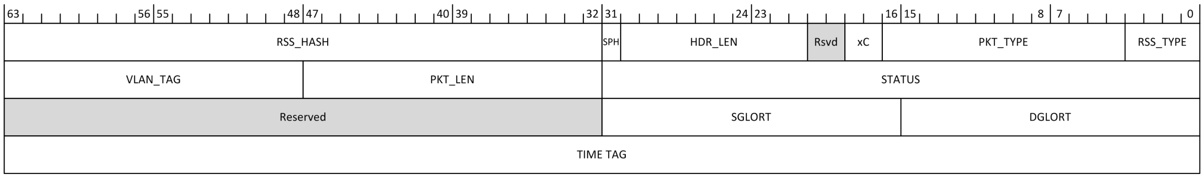


Figure 6-21 Receive Write Back Descriptor Read (32 Bytes)

The following fields are valid in all descriptors of a packet:

- DD flag (Descriptor Done)
- EOP flag (End of Packet)
- PKT_LEN field (Packet Length in the current buffer).

The following fields are valid only in the first descriptor of a packet:

- HDR_LEN (Header Length in the header buffer)
- SPH (Small Packet Header is identified for header split functionality)
- HBO (Header Buffer Overflow).

All other fields are valid only in the last descriptor of a packet.

6.18.7.1 RDESC.RSS_TYPE

RSS type is:

- 0x0 = No hash computation done for this packet
- 0x1 = HASH_TCP_IPv4
- 0x2 = HASH_IPv4
- 0x3 = HASH_TCP_IPv6
- 0x4 = Reserved
- 0x5 = HASH_IPv6
- 0x6 = Reserved
- 0x7 = HASH_UDP_IPv4
- 0x8 = HASH_UDP_IPv6
- 0x9 – 0xF = Reserved

6.18.7.2 RDESC.PKT_TYPE

Packet type is 13-bit mask:

- Bits 2:0 — L3 type

  000b = not IP
  001b = IPv4
  010b = IPv4 with extensions
  011b = IPv6
  100b = IPv6 with extensions
  101b = Reserved
  110b = Reserved
  111b = Reserved

- Bits 5:3 — L4 type

  000b = Unknown
  001b = TCP
  010b = UDP
  011b = GRE
  100b = VXLAN
  101b = NVGRE
  110b = GENEVE
  111b = Reserved

- Bit 8:6 — Inner L3 type (see L3 type in Section 6.18.7.2).
- Bit 11:9 — Inner L4 type (see L4 type in Section 6.18.7.2).
  - Except VXLAN/NVGRE/GENEVE is never set.
- Bit 12 — Reserved

6.18.7.3 RDESC.xC

The xC field encodes the xCast type of the packet as follows:

00b = Unicast
01b = Reserved
10b = Multicast
11b = Broadcast

6.18.7.4 RDESC.HDR_LEN

HDR_LEN is the number of bytes written in the header buffer. It is only valid in the first descriptor of the packet and is ignored in follow-on descriptors.

6.18.7.5 RDESC.SPH

When set to 1b, indicates that the hardware has found the length of the header or packet is a small packet. The header buffer is used only if SPH is 1 and HBO is 0. If the received header size is greater or equal to 1024 bytes, the SPH bit is not set and header split functionality is not supported. The SPH bit is meaningful only on the first descriptor of a packet. Refer to Section 6.18.9 for further details.

6.18.7.6 RDESC.DGLORT and RDESC.SGLORT

These fields holds the DGLORT and SGLORT received from the switch.

6.18.7.7 RDESC.RSS_HASH

This field holds the RSS hash value computed. Set to 0b if not computed.

6.18.7.8 RDESC.STATUS

This field defines the status of this packet. Two format are used depending if this is the last buffer of the packet or not. They are shown in Figure 6-22.

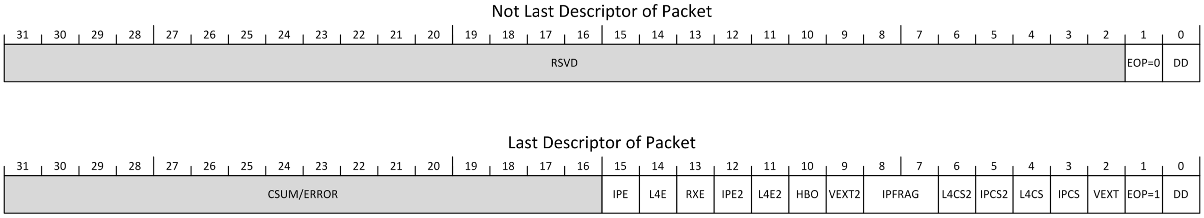


Figure 6-22 Receiver Status

EOP(1) and DD(0) — End of Packet and Done bits are listed in Table 6-13.

Table 6-13 EOP and DD Bits

DD	EOP	Description
0	0	Software setting of the descriptor when it hands it to hardware.
0	1	Reserved (invalid option)
1	0	A completion status indication for a non last descriptor of a packet that spans across multiple descriptors.
1	1	A completion status indication of the entire packet and software might take ownership of its descriptors. All fields in the descriptor are valid (reported by the hardware).

VEXT — A VLAN tag was found in the packet. Normally, the VLAN tag is not present in the packet and the VID/VPRI is transported in the FTAG (and copied to the RX descriptor).

VEXT2 — A custom tag was found in the packet.

IPCS — Indicates an IPv4 checksum was computed.

L4CS — Indicates an L4 checksum was computed.

IPCS2 — Indicates an IPv4 checksum was computed for an inner header.

L4CS2 — Indicates an L4 checksum was computed for an inner header.

IPFRAG — Indicates if this is a fragment or not (outer header only). The values are:

00b = Not a fragment.
01b = First fragment but the checksum is 0 (for UDP or GRE).
10b = First fragment and checksum not 0, computed checksum in CSUM field.
11b = Not first fragment, computed checksum in CSUM field.

HBO — The Header Buffer Overflow bit is set if the packet header (calculated by hardware) is bigger than the header buffer (defined by PCIE_SRRCTL.*BSIZEHEADER*). In this case, the header was not copied in the header buffer but kept with the frame in the data buffer.

L4E2 — This bit is set if L4 processing in the inner payload fails (TCP checksum or UDP checksum error).

IPE2 — This bit is set if L3 processing in the inner payload fails (IP checksum error or some other parsing error).

RXE — The RXE error bit is an indication that the packet has encountered an error. The error is logged in the CSUM/ERROR field.

L4E — This bit is set if L4 processing fails (TCP checksum or UDP checksum error).

IPE — This bit is set if L3 processing fails (IP checksum error or some other parsing error).

CSUM/ERROR — This field holds the L4 checksum (computed starting from first L4 byte, with no pseudo header) for IP fragments if RXE is cleared, or an error code if RXE is set. The L4 checksum inserted for IP fragments is the one computed by the hardware for outer L4 header only. For fragments, software could add all CSUM or each fragments of a given message to verify the final checksum of the packet and compare to the value expected. This field is valid only if either IPFRAG ≥ 2 or if RXE is set to 1; its contents is not inspected otherwise. The ERROR field is a bitmap where each bit reports the following error:

- SWITCH_ERROR(0) — The switch is reporting this packet as bad.
- NO_DESCRIPTOR(1) — Packet was terminated due to no descriptor available.
- PP_ERROR(2) — Packet was terminated due to a RAM error detected during packet processing.
- SWITCH_READY(3) — The SWITCH_READY signal was de-asserted in the middle of a packet.
- TOO_BIG(4) — PCIE_SRRCTL.*BufferChainingEn* bit is set to 0b and the packet would spread over more than a single data buffer. The rest of the packet is flushed.

6.18.7.9 RDESC.PKT_LEN

PKT_LEN holds the number of bytes posted to the packet buffer. The length covers the data written to a receive buffer excluding CRC bytes. Software must read multiple descriptors to determine the complete length for packets that span multiple receive buffers. If SRRCTL.*DESCTYPE* = 1 (header splitting enabled) and the buffer is not split because the header is bigger than the allocated header buffer, this field reflects the size of the data written to the data buffer (header + data), otherwise, the PKT_LEN does not include the header length. Similarly, if SRRCTL.*DESCTYPE* = 1 (small/large splitting enabled) and the packet is small and written to the header buffer, PKT_LEN is zero.

6.18.7.10 RDESC.VLAN_TAG

The Ethernet switch associates a VLAN ID and VLAN PRI for each packet. This information is written in the FTAG. The PCIe unit copies this information from the FTAG into the VLAN field on the RX descriptor. The packet might contain an extra VLAN tag and this is reported in the VEXT bit in the extended status field but otherwise not processed. The encoding is:

- VLAN_TAG[0..11] = VLAN ID
- VLAN_TAG[12..13] = VLAN PRI

6.18.7.11 RDESC.TIME_TAG

Time stamp is set for all packets and is the time at the moment the last byte of the packet was received on Ethernet ports or the moment the last byte of a packet was sent to switch from PCIe.

6.18.8 Receive Descriptor Queue

Figure 6-23 shows the structure of each of the receive descriptor rings, there is one receive descriptor ring per queue. Note that each ring uses a contiguous memory space, location and size are defined by the PCIE_RDBA[q] and PCIE_RDLEN[q] registers. The maximum ring size is 512 KB (16384 descriptors). Size must be a multiple of 128 bytes. The minimum ring size supported is 4 KB.

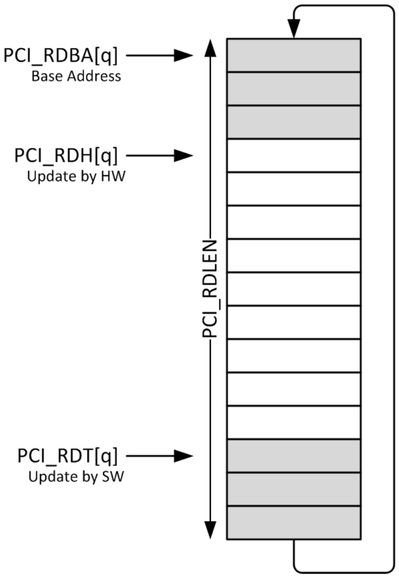


Figure 6-23 Receive Descriptor Queue

Software inserts receive descriptors by first write the update descriptors in the ring and then advancing the tail pointer(s) in the device (PCIE_RDT[q]) to refer to the address of the entry just beyond the last valid descriptor. The FM10000 adjusts its internal tail pointer(s) accordingly. As packets arrive, they are stored in memory and the internal head pointer(s) is incremented by the FM10000.

The FM10000 writes back used descriptors just prior to advancing the head pointer(s). Head and tail pointers wrap back to base when they reach the end of the ring.

The receive descriptor head and tail pointers reference to 32-byte blocks of memory. Shaded boxes in previous represent descriptors that have stored incoming packets but have not yet been recognized by software. Software can determine if a receive buffer is valid by reading descriptors in memory. Any descriptor with a DD bit set has been used by the hardware, and is ready to be processed by software.

Note: The head pointer points to the next descriptor that is to be written back. At the completion of the descriptor write-back operation, this pointer is incremented by the number of descriptors written back. Hardware owns all descriptors between [head... tail] which can be prefetched. Software does not change the content of these descriptors unless the queue is disabled. Any descriptor not in this range is owned by software.

The receive descriptor rings are described by the following registers:

- **Receive Descriptor Base Address registers** (PCIE_RDBA) — This register indicates the start of the descriptor ring buffer; this 64-bit address is aligned on a 64-byte boundary and is stored in two consecutive 32-bit registers. Hardware ignores the lower 4 bits.
- **Receive Descriptor Length registers** (PCIE_RDLEN) — This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128 bytes and be greater or equal to 4 KB (128 descriptors).
- **Receive Descriptor Head registers** (PCIE_RDH) — This register holds a value that is an offset from the base, and indicates the in-progress descriptor. There can be up to 64 K minus 8 descriptors in the circular buffer. Hardware maintains a shadow copy that includes those descriptors completed but not yet stored in memory. Software can determine if a packet has been received by either of two methods: reading the DD bit in the receive descriptor field or by performing a PCIe read of the Receive Descriptor Head register. Reading the RDH is however an unreliable method as reading this register is potentially faster than the time for the DMA controller write queue to complete.
- **Receive Descriptor Tail registers** (PCIE_RDT) — This register holds a value that is an offset from the base, and identifies the location beyond the last descriptor hardware can process. This is the location where software writes the first new descriptor. To avoid confusion between a full and an empty ring, PCIE_RDT is not advanced up to PCIE_RDH. This would be interpreted as an empty ring. It requires the ring be populated to maximum its full capacity minus one descriptor.

If software statically allocates buffers, and uses a memory read to check for completed descriptors, it simply has to zero the low status byte in the descriptor to make it ready for re-use by hardware. This is not a hardware requirement, but is necessary for performing an in-memory scan.

All the registers controlling the descriptor rings behavior should be set before the queue is enabled, apart from the tail registers which are used during the regular flow of data.

6.18.8.1 Receive Descriptors Prefetch

The FM10000 prefetches the descriptors in multiples of 2. aligned on a 64-byte boundary. The unit prefetches as many descriptors at a time as possible without:

- Crossing a 4 K boundary.
- Passing the tail pointer.
- Passing the end of the ring.
- Overflowing the prefetch FIFO.

Therefore the number requested is always between 2 and 128. A request is not issued if there are less than 2 descriptors remaining in the host's ring. As a consequence, an out-of-descriptors condition might arise if all descriptors are used and less than 2 remain unread. A request is not issued if the free space left in the internal cache of the queue is less than for 4 descriptors.

6.18.8.2 Receive Descriptors Optimized Write Back

The unit improves host memory writing to 64 bytes whenever possible by holding back writing descriptors until an aligned 64-byte cache line boundary is reached or until a time out occurs. An option (PCIE_RXDCTL.*WriteBackImm*) enables hardware to write back descriptors immediately when a packet is arrived regardless of current alignment.

6.18.8.3 Receiver Frame Event Posting

A receiver is considered having a frame receive event once a descriptor (or set of descriptors) is written back to host with an end of frame indication. The event is sent to the interrupt controller for processing.

6.18.8.4 Out of Descriptors Behavior

Each queue can be configured for two possible different handling when a queue becomes empty (no descriptors), either waiting for descriptors to arrive or drop the packet. This behavior is defined in PCIE_RXDCTL.*DropOnEmpty*.

If this option is set to 1b (DROP) and no descriptors are available at the time a packet is received, the packet affected is dropped. If the packet was already started to be pushed to the host and the last descriptor is used without reaching end of packet, the last descriptor has the RXE and EOP to 1 and drop the rest of the packet. In both cases, the event is counted.

If the option is set to 0b (HOLD) and the last descriptor of the queue is to be written and not end of packet, this last descriptor is withheld and the unit starts a timer (PCIE_DMA_CTRL.*MaxHoldTime*), waiting for descriptors to become available again. If the timer expires, the last descriptor get written with RXE and EOP set to 1, the queue is set to DROP to avoid future time out on this queue, and an interrupt is posted to the PF or VF to which the queue belongs.

6.18.9 Receiver Header or Small Packet Processing

This feature consists of saving packet headers or small packets into distinct buffers. The FM10000's support for this feature is controlled by the *DESCTYPE* field of the Split Receive Control registers (PCIE_SRRCTL[q]).

The following modes exist in both split and non-split modes:

- 0 = Always use data buffer.
- 1 = Header and payload are split to different buffers.
- 2 = Small packets are stored in small packet buffers.

The receive descriptors define two buffer physical addresses; header/small-packet buffer address and data buffer address.

The sizes of these buffers are statically defined in the PCIE_SRRCTL[q] registers:

- The *BSIZEPACKET* field defines the size of the buffer for the received packet.
- The *BSIZEHEADER* field defines the size of the buffer for the received header or small packets. If header split or small packet split is enabled, this field must be configured to a non-zero value.

When header split is selected, the packet is split only on selected types of packets. A bit exists for each option in PCIE_SRRCTL[q].*PSRTYPE*[n] registers. Therefore, several options can be used in conjunction. If one or more bits are set, the splitting is performed for the corresponding packet type that has the longest header. The options are configurable per queue. If a valid header was detected and the header is smaller or equal to *BSIZEHEADER*, the FM10000 writes the header portion of the packet into the header buffer and set SPH to 1 and HBO to 0. If the header does not fit in the header buffer, only HBO is set, and the entire header and the packet data payload get written into the normal packet buffer. If a valid header was not detected, entire packet is written to packet buffer and both SPH and HBO are cleared.

If small packets triage is used, the receiver checks if packets are smaller or equal to *BSIZEHEADER* and, if yes, stores the entire packet in the small-packet buffer, and the SPH is set to 1. If the packet is bigger than *BSIZEHEADER*, the entire packet goes into the normal data packet buffer and SPH is set to 0. The HBO is always set to 0 in this mode.

6.18.10 Receiver TCP/UDP Checksum Offload

The FM10000 supports the offloading of four receive checksum calculations:

- Outer IPv4 header checksum
  - Bit RDESC.IPCS set if computed, result reported in RDESC.IPE.
- Outer TCP/UDP checksum (or fragment checksum)
  - Bit RDESC.L4CS set if computed, result reported in RDESC.L4E.
- Inner IPv4 header checksum
  - Bit RDESC.IPCS2 set if computed, result reported in RDESC.IPE2.
- Inner TCP/UDP checksum
  - Bit RDESC.L4CS2 set if computed, result reported in RDESC.L4E2.

The inner checksums are only supported for NVGRE, VXLAN, and GENEVE tunnels.

The FM10000 supports only Ethernet II frames for checksum offload.

The FM10000 checksum verification engines are always enabled. However, they do are not always report a result, the IPCS/L4CS/IPCS2/L4CS2 are set only if a meaningful result (pass/fail) is to be interpreted. The IPCS/L4CS/IPCS2/L4CS2 are cleared if the corresponding checksum was not computed.

Table 6-14 indicates when checksum are computed, the table applies to outer header, or on inner header of VXLAN/NVGRE/GENEVE tunnels.

Table 6-14 Packet Type Receiver Checksum Validation

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IP header's protocol field contains a protocol # other than TCP/UDP.	Yes	No
IPv4 + TCP/UDP packets.	Yes	Yes
IPv6 + TCP/UDP packets.	No (N/A)	Yes
IPv4 packet has IP options (IP header is longer than 20 bytes).	Yes	Yes
IPv6 packet with next header options: • Hop-by-hop options. • Destinations options (without home address). • Destinations options (with home address). • Routing (with segment left 0). • Routing (with segment left > 0). • Fragment.	 No (N/A) No (N/A) No (N/A) No (N/A) No (N/A) No (N/A)	 Yes Yes No Yes No No
Packet has TCP or UDP options.	Yes	Yes
IPv4 tunnels other than NVGRE/VXLAN/GENEVE: • IPv4 packet in an IPv4 tunnel. • IPv6 packet in an IPv4 tunnel.	 Yes (outer IPv4) Yes (IPv4)	 No No

Table 6-14 Packet Type Receiver Checksum Validation [Continued]

Packet Type	Hardware IP Checksum Calculation	Hardware TCP/UDP Checksum Calculation
IPv6 tunnels other than NVGRE/VXLAN/GENEVE: • IPv4 packet in an IPv6 tunnel. • IPv6 packet in an IPv6 tunnel.	 No No	 No No
Packet is an IPv4 fragment (outer).	Yes	Checksum assist
Packet is an IPv4 fragment (inner of an outer IP fragment).	No	No
Packet is greater than 1522 bytes.	Yes	Yes
Packet has 802.3ac tag.	Yes	Yes

Note: The FM10000 ignores the content of the IPv6 routing header when computing a checksum and always uses the destination IP v6 header in the IPv6 header when constructing the pseudo header for checksum computation. If the packet is at destination, the checksum is correctly computed. If the packet is not at destination (attached host is expected to do forwarding by example), the checksum computed by the device is most likely erroneous, and software has to redo the checksum in those cases only.

L4 checksums are computed on IPv4 fragments and reported for the outer header only, the RDESC.IPFRAG reports which fragment it is and the checksum is saved in the descriptor.

6.18.11 Receiver Performance

The receiver's parser process data from the switch as it arrives and accumulate the data into a data chunk size equal to a TLP size as determined in Header Processing section.

6.19 Packet Transmission

6.19.1 Overview

The transmitter consist of few blocks shown in Figure 6-24.

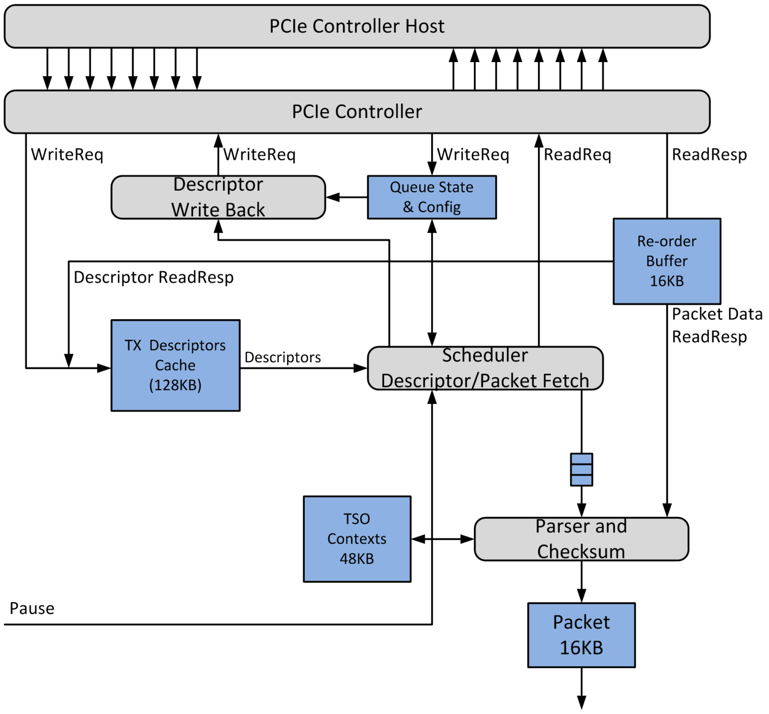


Figure 6-24 PCIe Transmitter Block Diagram

The transmitter includes the following components:

- Queue Status & Configuration
  - — Holds current queue configuration and status for each of 256 queues.
- Transmit Packet Descriptors Cache
  - — Holds descriptors and packets waiting for transmission.
- Descriptor Write Back
  - — Collects completed descriptors write back.
- Scheduler Descriptors/Packet Fetch
  - — Unit responsible to schedule transmission and recover packet payload or descriptors.
- Parser and Checksum
  - — Parser packets from host and compute checksums.

- Transmit Segmentation Offload Context
  - — Holds current data for one queue TSO while serving other queues.
- Re-order Buffer
  - — Re-order transactions from host.
- Packet
  - — Stores one packet for checksum offload.

The transmitter supports two models:

- Pull model — Descriptors and packet payload are recovered from memory by controller.
- Push model — Descriptors are written by host directly into controller.

The pull/push model is configurable per queue.

6.19.2 Pull Model

In the pull model, transmit packets are made up of data buffers in host memory that are indicated to hardware by pointer and length pairs. These pointer and length pairs are named as transmit descriptors that are stored in host memory as well.

Software prepares memory structures for transmission by assembling a list of descriptors. It then indicates this list to hardware by updating the on-chip transmit tail pointer. The hardware fetches descriptors for any non-empty queue by issuing a read request to the host memory, the response is stored in on-die transmit data cache. The hardware stops fetching when the on-die cache is almost full for that queue (i.e. on-die space left for less than 4 descriptors).

A packet (or multiple packets in transmit segmentation) can be composed of one or multiple buffers, each buffer being described by one descriptor. Descriptors of a single packet are consecutive, while the first one points to the first buffer and the last one points to the last buffer. The following rules must be kept:

- Address alignment of the data buffers can be on any byte boundary.
- Total length of all data buffers for any given packet must be at least 17 bytes. Anything smaller is not supported, and causes the packet to be dropped.
- A packet (or multiple packets in transmit segmentation) can span any number of buffers (and their descriptors) up to the capacity of the descriptor icache set by PCIE_TQDLOC[Q].*Size*. For best performance it is recommended to minimize the number of buffers as much as possible.

The transmit scheduler then determine which transmit queue to serve and retrieves packets from memory by issuing a read request using the content of the descriptor stored in the cache. The read response contains the packet data which is forwarded to a parser for computing checksum. If checksum or TSO is required, a packet needs to be stored fully in the FIFO so that the checksum could be updated. If checksum or TSO is not required, the packet transmission might start earlier. The fabric store-and-forward function must be properly setup to ensure no under-run occurs (speed of Ethernet port slower than speed of PCIe and packets are guaranteed to not be split across multiple descriptors).

To keep the data rate as high as possible, the descriptor prefetch and scheduler are decoupled from the response and the controller could issue up to 32 read requests for either descriptor or data before waiting for responses.

The hardware uses a re-order buffer to accumulate the read completions for packet data fetch to make sure all parts of a packet has been received before a packet is transmitted to the switch. The size of this buffer is limited to 16 KB. Therefore, hardware checks the length of a packet and flushes a packet that exceeds either 15 KB in total length, or uses more descriptors than the capacity of the descriptor icache set by PCIE_TQDLOC[Q].*Size*. If this occurs, the hardware disables the queue, returns it to PF, and logs a fault in the PCIE_LVMMC register.

6.19.3 Push Model

In the push model, the descriptors are assumed to be written by host directly into the device's transmit descriptor cache. The descriptor prefetch is inactive for the queues configured in this mode.

For the transmission to work properly, the packet data must be written in the host memory prior to the descriptor writes and the most-significant word of the last descriptor of the packet must be written last, the data and descriptors might be written in multiple TLPs, each of Nx32. In the push model, the device watches the host writes into the descriptor cache and advance its tail pointer as soon as a last word of a descriptor as DONE=0, LAST=1 set. In the pull mode, the only method to inform that new descriptors are available is by writing the PCIE_TDT pointer.

The push descriptor model is enabled in PCIE_TXDCTL[q].*PushDesc* setting.

Transmit segmentation offload (TSO) and checksum offload are supported in PUSH mode.

6.19.4 Transmit Scheduler

Each queue belongs to either a traffic shaping class (TC) or to the unlimited bandwidth class, and to a Priority Flow Control (PFC) class:

- PCIE_TXQCTL[q].*TC* — Defines a traffic shaping class (6 bits, max 64 groups).
- PCIE_TXQCTL[q].*UnlimitedBW* — Allocates unlimited bandwidth to the queue (when set, TC field is meaningless).
- PCIE_TXQCTL[q].*PC* — Defines PFC class (3 bit, max 8 groups).

A traffic-shaped queue is blocked from retrieving packets from host memory if either of its two classes (TC or PFC) is blocked. An unlimited bandwidth queue is blocked only if its PFC is blocked.

A PFC class is blocked if it received a PAUSE with non-zero timer for that PFC class and unblocked if it receives a PAUSE with a zero timer for that group, or if no refresh was received for a while for any pause group.

A traffic shaping class is blocked if the PCIE_TC_CREDIT[group] is negative and unblocked if the PCIE_TC_CREDIT[group] is positive. Traffic shaping groups are credited periodically (512 ns period to refresh all groups) and debited by packet size when all posted read for a given packet have been requested from a host.

Packets that have pending read requests are considered committed, and are forwarded to fabric once the read completion are received.

The scheduler selects packets through a round-robin priority scheme among all queues that have packets and are not blocked.

6.19.5 Packet Padding and Transmitter Throttling

The transmitter expects packets recovered from host memory to be at least 15 bytes and at most 15 KB. Any packet with less than 15 bytes or packet greater than 15 KB causes the packet to be dropped. Note that a chain buffer for a message can be greater than 15 KB if TSO is requested (e.g. an MSS is provided), only individual packets that are greater than 15 KB are actually causing an error.

The transmitter will pad small packets so that the packet length with FTAG and time tag in place is at least 72 bytes. The padding is done by adding 0s before the 4 bytes time tag is added. As an example, if a packet recovered from host is 48-bytes long and the FTAG header is not included in the packet, the transmitter adds 12 bytes to the packet before adding the 4 byte time tag. If the packet recovered from host is 48-bytes long and included the FTAG, the transmitter adds 20 bytes to the packet.

Also, the transmitter supports a minimum packet to packet spacing (start of packet to next start of packet). The transmitter starts a timer at the start of a packet and delays starting the next packet until this timer expires. The timer is defined in PCIE_DMA_CTRL.*TxFrameSpacing* register and is in clock cycles (one clock cycle is 2 ns for GEN3, 4 ns for GEN2 and 8 ns for GEN1). Because the minimum packet size is 72 B and the data path width is 128 bits, setting to any value below 5 is equivalent to have no throttle at all.

6.19.6 Transmit Descriptors

The FM10000 supports one type of transmit descriptors which can be used for both TSO requests and non-TSO requests.

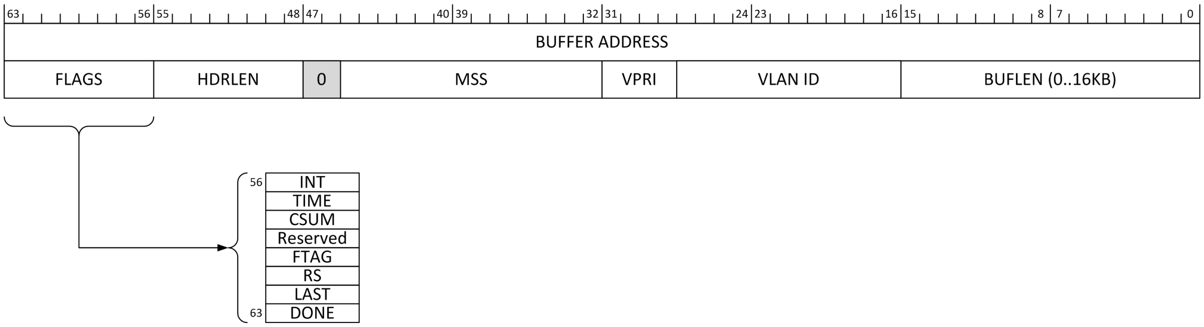


Figure 6-25 Transmit Descriptors Format

The BUFLEN is the number of valid bytes in this buffer. A packet can be broken into an arbitrary number of TX descriptors and the number of bytes per descriptor is also arbitrary. The total number of descriptors for a given packet must be smaller than the number of descriptors that can be hold in the on-die TX descriptor queue size.

The BUFFER ADDRESS is the address of the buffer in host memory.

The VPRI/VLAN_ID is the VLAN priority and VLAN ID that is attached to the frame if an FTAG is generated. If a value of 0 is supplied as a VLAN ID, a default per queue VLAN ID gets assigned (see PCIE_TXQCTL.*VID* in Section 11.27.2.71). If the default VLAN ID for the queue is also 0, the switch sets a default value. The VPRI cannot be defaulted, software must select a default VPRI though it is not used by the transmitter to prioritize this frame versus another, prioritization is controlled by queue only. The VPRI is not necessarily used by the switch for prioritization and policers, as an example, the switch might be configured to select IP DSCP to control priority of a given packet through the switch.

The HDRLEN and MSS fields determine the packet size for TCP Segmentation Offload. These fields allow the packet fetch block to compute the exact amount of data to be read from the processor during segmentation offload. If either field is set to 0b, TCP segmentation offload is disabled. These fields are interpreted and latched on the first descriptor of the TSO request and ignored on successive descriptors for the same TSO request. The packet fetch block retrieves up to (HDRLEN + MSS) for the first packet of the TSO request and retrieves up to MSS for follow-on packets, all data fetches are limited ultimately by the size of the buffer. If a tunnel is present, MSS is the inner TCP payload length an the HDRLEN includes all inner and outer headers.

The FLAGS are a set of bits for this packet:

- DONE — Indicates hardware has completed transmission data pointed by this descriptor.
- LAST — Indicates this is the last descriptor of a packet.
- TIME — Indicates egress time stamp request (encoded in USER[1]), the time tagging is done in the Ethernet port and reported through a switch interrupt.
- CSUM — Indicates request for IPV4 and/or TCP/UDP checksums, wherever it is applicable.
- FTAG — Indicates that this packet has an FTAG header present.
- RS — Indicates a request to write back this descriptor when done transmitting.

The bits DONE, and RS are valid on every descriptor of the packet or TSO request. The bits CSUM and FTAG are only valid on first descriptor of a packet or a TSO request and ignored on successive descriptors for the same packet. The bit TIME is only valid on first descriptor of a packet, it is ignored on TSO requests.

6.19.7 Generating FTAG

All packets transmitted to the switch have an FTAG at the beginning of the packet. If the descriptor already indicates that the FTAG is present, the transmitter does not need to add one. If the FTAG is not present, the transmitter creates one using the following rules:

- DGLORT set to 0b.
  - — SGLORT set to PCIE_TX_SGLORT[q].
- TYPE set to 0b.
- USER set to 0b.
  - — Set USER bit 1 to 1b is TIME is set in the TX Descriptor.
- VPRI/VID set to VPRI/VID from buffer descriptor.
- SWPRI set to VPRI.

For PAUSE packets generated by the end point toward the switch, the setting is:

- DGLORT set to 0b.
- SGLORT set to PCIE_TX_SGLORT[0].
- TYPE set to 0b.
- USER set to 0b.
- VPRI/VID set to 0b.
- SWPRI set to 0b.

Note: The PCIE_PFVTCTL[vf].*FTagDescEnable* bit indicates if a particular VM is allowed to generate packets with FTAG or not. If a VM has generated a packet with FTAG while it is not allowed to do so, this is considered as a fault; the queue is shut down and the event is reported to the PF.

6.19.8 Transmitter Descriptor Queue Structure

The transmit descriptor ring structure (shown in Figure 6-26) uses a contiguous memory space, in host memory or in the transmit cache. A set of four registers (described later in this section) maintain the transmit descriptor ring in the host memory. Hardware maintains internal circular queues of descriptors per queue to hold the descriptors that were fetched from the software ring.

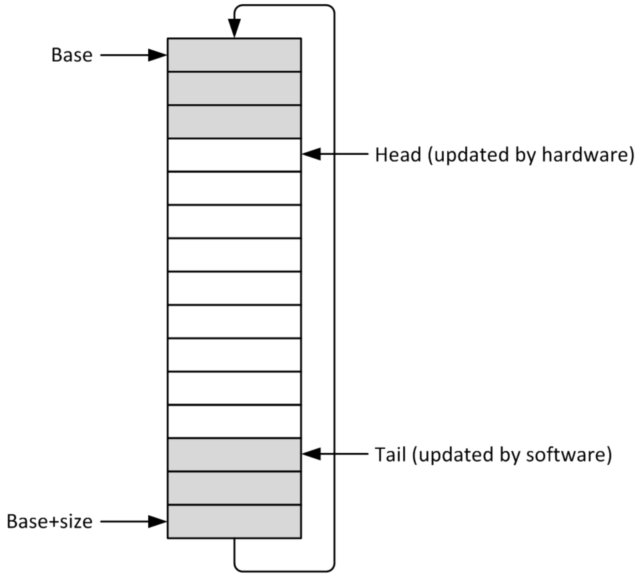


Figure 6-26 Transmit Queue Structure

Descriptors handled to hardware should not be manipulated by software until hardware completes its processing. It is indicated by advancing the head pointer beyond these descriptors.

The transmit descriptor ring is defined by the following registers:

- **Transmit Descriptor Base Address register** (PCIE_TDBA) — This register indicates the start address of the descriptor ring buffer in the host memory; this 64-bit address is aligned on a 64-byte boundary and is stored in two consecutive 32-bit registers. This must be set for both the push model and the pull model. In the push model, this space is used for descriptor writes back only.
- **Transmit Descriptor Length register** (PCIE_TDLEN) — This register determines the number of bytes allocated to the circular buffer. This value must be a multiple of 128 bytes. The maximum ring size is 512 KB (32768 descriptors of 16 bytes). The minimum supported number of descriptors in a ring is 32.
- **Transmit Descriptor Head register** (PCIE_TDH) — This register holds a value that is an offset from the base and indicates the in-progress descriptor. There can be up to 64 K minus 8 descriptors in the circular buffer. The transmit queue consists of the descriptors between the head and tail pointers. Transmission starts with the descriptor pointer by the head registers. When the DMA engine processes a descriptor, it might optionally write back the completed descriptor and then

advance the head pointer. It then processes the next descriptor up to the point that the head pointer reaches the tail. Head equals tail means that the transmit queue in host memory is empty. Reading this register indicates the hardware progress to software. All descriptors behind the head pointer and in front of tail register are owned by software. The other descriptors are owned by the hardware and should not be modified by software.

- **Transmit Descriptor Tail register** (PCIE_TDT) — This register holds a value, which is an offset from the base, and indicates the location beyond the last descriptor hardware can process. Software adds new descriptors to the ring by writing descriptors in the circular buffer pointed by the tail pointer. The new descriptor(s) are indicated to hardware by updating the tail pointer one descriptor above the last added descriptor. Note that a single packet or TSO might be composed of multiple descriptors. The transmit tail pointer should never point to the middle of a packet or TSO, which might cause undesired software/hardware races. To avoid confusion between a full and an empty ring, PCIE_TDT is not advanced up to PCIE_TDH. This would be interpreted as an empty ring. It requires the ring be populated to maximum its full capacity minus one descriptor. Any single Tx packet or any packet in a TSO does not span over more than 32 descriptors.

Software might detect which packets have already been processed by hardware using the following:

- Read the TDH head register to determine which packets (those logically before the head) have been transferred to the on-chip FIFO or transmitted. This method is not recommended as races between the internal update of the head register and the actual write back of descriptors can occur.
- Software can track the DD bits in the descriptor ring. Descriptor write back is controlled by the RS bit and interrupt assertion.
- Issue an interrupt. An interrupt condition is generated each time a packet was transmitted or received and a descriptor was write back or transmit queue goes empty. This interrupt can either be enabled or masked.

All of the registers controlling the descriptor rings behavior should be set before transmit is enabled.

6.19.8.1 Transmit Descriptor Fetching

In pull model, the FM10000 fetches new descriptors from host memory as required for packet transmission depending on its on-die descriptor buffer cache state. It always tries to grab as much descriptors as possible, up to the limit of the local cache memory, without regards to host cache line boundary.

6.19.8.2 Transmit Write Back

The FM10000 periodically updates software on its progress in processing transmit buffers by writing back into the Tx descriptor queue. The write back mechanism is the same for both the push or pull models.

Processed descriptors, i.e. those for which read completion were received for the data they contains, have their DONE bit set and are pushed to a small 4-descriptor FIFO per queue. If the last descriptor has the RS bit set, this descriptor and previous ones up to a 64-byte cache line boundary are written to host memory. As an example, if descriptors 0,1,2,3,4,5,6 were processed and only descriptor 6 has an RS bit set, the FIFO contains 3,4,5,6 at the time RS is detected but only descriptors 4,5,6 are written to memory and not descriptor 3 which is part of previous cache line.

Also, a timer (time defined in PCIE_TXDCTL[q].*MaxTime*) is started every time a write-back descriptor is written to the small 4-descriptor FIFO. If the timer expires, the last descriptor in the FIFO and previous ones up to a 64-byte cache line boundary are written to host memory and an interrupt is posted after the write to indicate that the transmitter is empty.

For performance reasons, the RS is set by driver to create full 64-byte cache lines write backs, i.e. setting the bit for the last descriptor of a cache line and reducing write backs by only setting RS for every N cache line. Driver is only required to check DONE bit for descriptors with RS bit set or for the last one posted and, if these descriptors have their DONE bit set, all previous descriptors are considered done as well.

6.19.9 Transmit TCP Segmentation & Checksum Offload

6.19.9.1 Overview

6.19.9.1.1 TCP Segmentation

Hardware TCP segmentation offload is a feature allowing TCP/IP stacks to pass to the network device driver a TCP message that is bigger than the Maximum Segment Size (MSS) allowed on the medium. It is then the responsibility of the hardware to divide the TCP message into MSS frames that have appropriate layer 2/3/4 headers. These headers must include sequence number, checksum fields, options and flag values as required. Note that some of these values (such as the TCP sequence number) are unique for each packet of the TCP message while other fields such as the source IP address is constant for all packets associated with the TCP message.

The FM10000 supports both TCP segmentation for the following applications:

- Large TCP over VXLAN/GENEVE tunnels
- Large TCP over NVGRE tunnels
- Large TCP over IP

Segmentation offload is automatically enabled as soon as a packet size exceed the MSS provided in the descriptor (setting MSS to 0b disables segmentation). The FM10000 does not allow MSS smaller than 64 bytes, however a global minimum value could be configured to enforce a more reasonable minimum value (PCIE_DMA_CTRL.*MinMSS*). Also, the combined length of all headers (FTAG, outer L2/L3/L4 and inner L2/L3/L4) should not exceed 192 bytes.

The largest TCP segmented message supported is (256 KB - 1).

Note: The FM10000 does not support IP fragmentation on transmit.

6.19.9.1.2 Checksum Offload

Hardware checksum offload is a feature allowing TCP/IP stacks to offload TCP/IP checksums to hardware. The FM10000 supports checksums computation for the IPv4 header, and the TCP and UDP headers of non-tunneled packets. For the supported tunneled applications (NVGRE, VXLAN and GENEVE tunnels only), the FM10000 supports checksums computation for outer IPv4 headers, and for inner IPv4, TCP and UDP headers. Other (unsupported) tunnels are identified as non-tunneled packets, which means checksums computation is done for the outer IPv4 header and outer TCP/UDP header is present.

Note: The FM10000 does not support forcing a bad CRC during TCP segmentation offload.

6.19.9.1.3 Packet Parsing

The transmitter parses the packet to determine the type of packet being send and locate all required headers and checksum locations.

6.19.9.2 Transmission Process

The transmission process involves the following:

- The protocol stack receives from an application a block of data that is to be transmitted.
- The protocol stack calculates the number of packets required to transmit this block based on the MSS allowed on the media and required packet headers.
- The stack interfaces with the device driver and passes the block down with the appropriate header information: Ethernet, IP and TCP headers.
- The stack interfaces with the device driver and commands the driver to send the individual packet. The device driver sets up the interface to the hardware (via descriptors) for the TCP segmentation.
- The hardware transfers (DMA's) the packet data and transmit data saving the header including:
  - Packet encapsulation
  - Header generation and field updates including IPv4/IPv6 and TCP/UDP checksum generation as well as IPv4 Identification field increment by one every packet (both in inner/outer headers for tunneled packets).
- The driver returns ownership of the block of data to the operating system when the hardware has completed the DMA transfer of the entire data block.

6.19.9.3 TCP Segmentation Data Fetch Control

To perform TCP segmentation in the FM10000, the Scheduler and Packet Fetch uses the HDRLEN and MSS size to calculate how much data to fetch for each packet and sends an indication to the Parser and Checksum Unit about the packet size to be transmitted.

The FM10000 enables interleaving between different TSO requests at an Ethernet packet level. In other words, the FM10000 might fetch part of a TSO from a queue, equivalent to one Ethernet packets, then transition to another queue and fetch the equivalent of one or more packets (TSO or not), then move to another queue (or the first queue), etc. The FM10000 decides on the order of data fetched based on its QoS requirements (such as bandwidth allocation and priority).

The FM10000 saves the packet header in on-chip memory while interleaving other queues and indicates disposals of descriptors after they are transmitted, there is no need for the host to wait for the whole TSO to complete to free up the buffers holding the headers.

6.19.9.4 TCP Segmentation Performance

Performance improvements for a hardware implementation of TCP segmentation offload include:

- The stack does not need to partition the block to fit the MSS, saving CPU cycles.
- The stack only computes one Ethernet, IP, and TCP header per segment, saving CPU cycles.
- The stack interfaces with the device driver only once per block transfer, instead of once per frame.
- Larger PCI bursts are used, which improves bus efficiency (such as lowering transaction overhead).
- Interrupts are easily reduced to one per TCP message instead of one per packet.
- Fewer I/O accesses are required to command the hardware.

6.19.9.5 TCP Segmentation

Software indicates a TCP segmentation transmission context simply by setting an MSS and HDRLEN in the descriptor. The hardware parses the packet to define where the checksums are, and also saves the headers for later fragments. The parser locates checksum in outer the header for IPv4 only, and in the inner header for NVGRE, VXLAN, and GENEVE tunnels.

The header can be up to 192 bytes in length (including FTAG).

The driver and protocol stack construct the L3/IP/L4 headers as if it was transmitted as a single packet, i.e. all headers must be constructed correctly except for the following:

- IPv4/TCP/UDP checksums are updated by hardware for each segment
- IP packet length is updated by hardware for each segment

For TCP header, the sequence number should be set as appropriate for FIRST packet sent (first segment) and the PSH, and FIN flags should be set as appropriate for LAST packet sent (last segment).

6.19.9.5.1 Headers for the First Segment

The hardware makes the following changes to the headers of the first packet of a large TCP transfer.

- Outer IPv4 Header:
  - Set IP Total Length.
  - Calculates the IP Checksum.
- Outer IPv6 Header:
  - Set IPv6 Payload Length.
- Outer NVGRE/VXLAN/GENEVE header (if present):
  - Set UDP Payload Length for VXLAN/GENEVE.
- Inner IPv4 Header (if present):
  - Set IP Total Length.
  - Calculates the IP Checksum.
- Inner IPv6 Header (if present):
  - Set IPv6 Payload Length.
- TCP Header:
  - Sequence Number — The value is the sequence number of the first TCP byte in this frame.
  - The flag values of the first frame are set by logic AND function between the flag word in the pseudo header and the PCIE_DTXTCPFLGL.*TCP_flg_first_seg*. The default values of the PCIE_DTXTCPFLGL.*TCP_flg_first_seg* are set. The flags in a TSO that ends up as a single segment are taken from the in the pseudo header in the Tx data buffers as is.
  - Calculates the TCP checksum.

6.19.9.5.2 TCP/IP Header for the Subsequent Frames

The hardware makes the following changes to the headers of the intermediate segments of a large TCP transfer.

- Outer IPv4 Header:
  - — Set IP Total Length.
  - — Calculates the IP Checksum.
- Outer IPv6 Header:
  - — Set IPv6 Payload Length.
- Outer NVGRE/VXLAN/GENEVE header (if present):
  - — Set UDP payload length for VXLAN/GENEVE.
- Inner IPv4 Header (if present):
  - — Set IP Total Length.
  - — Calculates the IP Checksum.
- Inner IPv6 Header (if present):
  - — Set IPv6 Payload Length.
- TCP Header:
  - — Sequence Number update — Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
  - — The flag values of the subsequent frames are set by logic AND function between the flag word in the pseudo header with the PCIE_DTXTCPFLGL.*TCP_Flg_mid_seg*. ** The default values of the PCIE_DTXTCPFLGL.*TCP_Flg_mid_seg* are set.
  - — Calculate the TCP checksum.

6.19.9.5.3 TCP/IP Header for the Last Frame

The hardware makes the following changes to the headers of the last segment of a large TCP transfer.

- Outer IPv4 Header:
  - — Set IP Total Length.
  - — Calculates the IP Checksum.
- Outer IPv6 Header:
  - — Set IPv6 Payload Length.
- Outer NVGRE/VXLAN/GENEVE header (if present):
  - — Set UDP length.
- Inner IPv4 Header (if present):
  - — Set IP Total Length.
  - — Calculates the IP Checksum.
- Inner IPv6 Header (if present):
  - — Set IPv6 Payload Length.

- TCP Header:
  - Sequence number update — Add previous TCP payload size to the previous sequence number value. This is equivalent to adding the MSS to the previous sequence number.
  - The flag values of the last frames are set by logic AND function between the flag word in the pseudo header and the PCIE_DTXTCPFLGH.*TCP_Flg_lst_seg*. The default values of the PCIE_DTXTCPFLGH.*TCP_Flg_lst_seg* are set. The flags in a TSO that ends up as a single segment are taken from the in the pseudo header in the Tx data buffers as is.
  - Calculate the TCP checksum.

6.19.10 Transmit Checksum Offloading

For each packet transmitted, the hardware can perform up to three checksum computations:

- Outer IPv4
- Inner IPv4
- Inner L4 (including pseudo-checksum)

The checksums computation are enabled using the CSUM bit in the descriptor.

- If CSUM is set:
  - Update outer IPv4 checksum.
  - Set checksum to 0b for VXLAN/GENEVE.
  - Update inner IPv4 checksum.
  - Update inner L4 checksum for UDP, TCP.

All checksum calculations use a 16-bit wide one's complement checksum. The checksum word is calculated on the outgoing data.

Note: The FM10000 does not support computing checksums that required substitution of the IPv6 addresses in the pseudo header with an address in an IPv6 option header. Thus, if the IPv6 packet header contains Routing Header Extensions (which require a substitution, refer to RFC 2460 Section 8.1) or if the IPv6 packet header contains a destination header with a home address (refer to RFC 3775 Section 6.1.1), the checksum calculation is incorrect. For those packets, software clears the CSUM bit and computes the checksum itself.

6.20 Virtualization

6.20.1 Overview

I/O virtualization is a mechanism that can be used to share I/O resources among several consumers. For example, in a virtual system, multiple operating systems are loaded and each operates as though the entire system's resources were at its disposal. However, for the limited number of I/O devices, this presents a problem because each operating system might be in a separate memory domain and all the data movement and device management has to be done by a Virtual Machine Monitor (VMM). VMM access adds latency and delay to I/O accesses and degrades I/O performance. Virtualized devices are designed to reduce the burden of the VMM by making certain functions of an I/O device shared among multiple guest operating systems or a Virtual Machine (VM), thereby allowing each VM direct access to the I/O device.

The FM10000 supports two modes of operations of virtualized environments:

- Direct assignment of part of the port resources to different guest operating systems using the PCI SIG SR IOV standard. Also known as native mode or pass through mode. This mode is referenced as IOV mode throughout this section.
- Central management of the networking resources by an IOVM or by the VMM. Also known as software switch acceleration mode. This mode is referred to as Next Generation VMDq mode in this section.

The virtualization offloads capabilities provided by the FM10000 apart from the replication of functions defined in the PCI SIG IOV specification are part of Next Generation VMDq.

A hybrid model, where part of the VMs are assigned a dedicated share of the port and the rest are serviced by an IOVM is also supported. However, in this case the offloads provided to the software switch might be more limited. This model can be used when parts of the VMs run operating systems for which VF drivers are available and thus can benefit from an IOV and others that run older operating systems for which VF drivers are not available and are serviced by an IOVM. In this case, the IOVM is assigned one VF and receives all the packets with Ethernet MAC addresses of the VMs behind it.

The number of VFs configured in the system must be set before any VFs become active, and cannot be changed unless all VFs are disabled.

The following section describes the support the FM10000 provides for these modes.

This section assumes a single-root implementation of IOV and no support for multi-root.

6.20.2 Direct Assignment Model

The direct assignment support in the FM10000 is built according to the following model of the software environment.

It is assumed that one of the software drivers sharing the port hardware behaves as a master driver (Physical Function or PF driver). This driver is responsible for the initialization and the handling of the common resources of the port. All the other drivers (Virtual Function drivers or VF drivers) might read part of the status of the common parts but cannot change them. The PF driver might run either in the VMM or in some service operating system. It might be part of an IOVM or part of a dedicated service operating system.

In addition, part of the non time-critical tasks are also handled by the PF driver. For example, access to CSR through the memory-mapped I/O space or access to the configuration space are available only through the master interface. Time-critical CSR space like control of the Tx and Rx queue or interrupt handling is replicated per VF, and directly accessible by the VF driver.

Note: In some systems with a thick hypervisor, the service operating system might be an integral part of the VMM. For these systems, each reference to the service operating system in the sections that follow refer to the VMM.

6.20.2.1 Rationale

The direct assignment model enables each of the VMs to receive and transmit packets with minimum of overhead. Non time-critical operations such as initialization and error handling can be done via the PF driver. In addition, it is important that the VMs can operate independently with minimal disturbance. It is also preferable that the VM interface to hardware be as close as possible to the native interface in non-virtualized systems to minimize the software development effort.

The main time critical operations that require direct handling by the VM are:

- Maintenance of the data buffers and descriptor rings in host memory. To support this, the DMA accesses of the queues associated to a VM should be identified as such on the PCIe using a different requester ID.
- Handling of the hardware ring (tail bump and head updates).
- Interrupts handling.

The capabilities needed to provide independence between VMs are:

- Per VM reset and enable capabilities.
- Tx rate control.
- Allocating separate CSR space per VM. This CSR space is organized as close as possible to the regular CSR space to enable sharing of the base driver code.
- The rate control and VF enable capabilities are controlled by the PF.

6.20.3 System Overview

The following drawings show the various elements involved in the I/O process in a virtualized system. Figure 6-27 shows the flow in software Next Generation VMDq mode, and Figure 6-28 shows the flow in IOV mode.

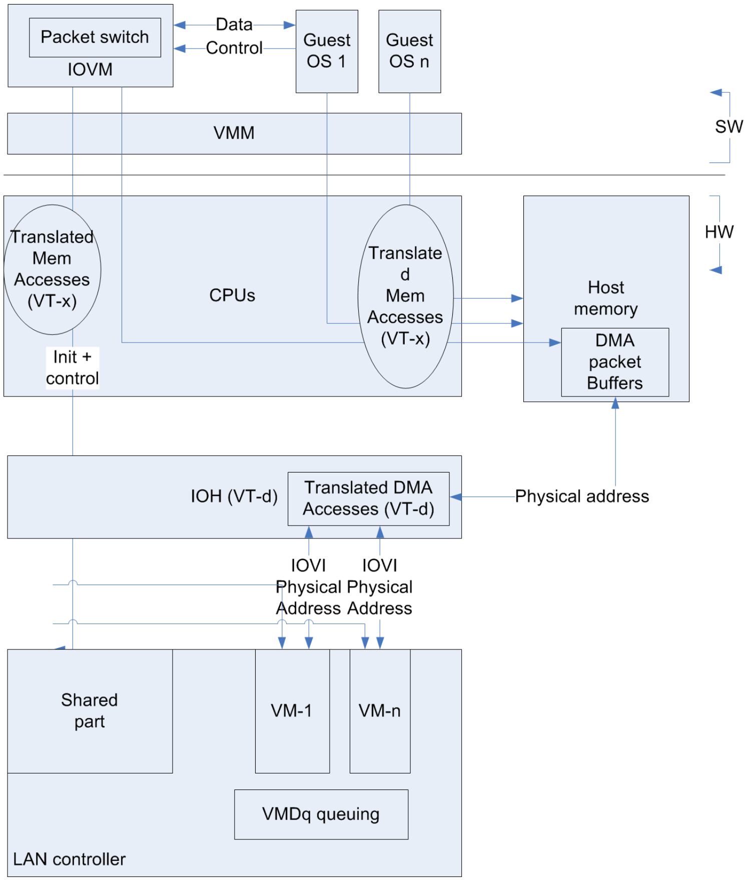


Figure 6-27 Virtualization - VMDq Model

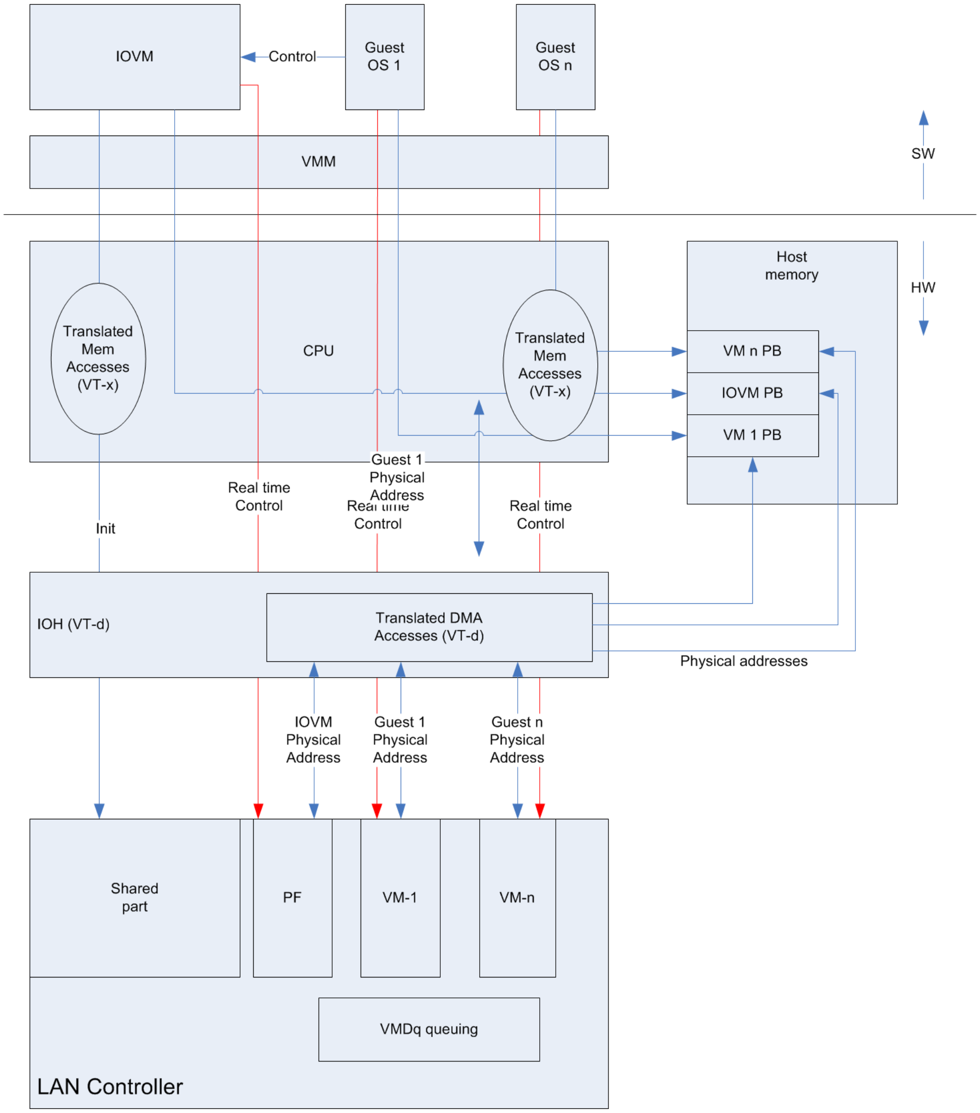


Figure 6-28 Virtualization - SR-IOV Model

This section assumes that in IOV mode, the driver on the guest operating system is aware that it operates in a virtual system (para-virtualized) and there is a channel between each of the VM drivers and the PF driver allowing message passing such as configuration request or interrupt messages. This channel can use the mailbox system implemented in the FM10000 or any other means provided by the VMM vendor.

6.20.4 PCI-SIG SR-IOV Support

6.20.4.1 SR-IOV Concepts

SR-IOV (Single Root I/O Virtualization) defines the following entities in relation to I/O virtualization:

- **Virtual Image (VI)** — Part of the I/O resources are assigned to a A VM.
- **I/O Virtual Intermediary (IOVI) or I/O Virtual Machine (IOVM)** — A special VM that owns the physical device and is responsible for the configuration of the physical device.
- **Physical function (PF)** — A function representing a physical instance — One port for the FM10000. The PF driver is responsible for the configuration and management of the shared resources in the function.
- **Virtual Function (VF)** — A part of a PF assigned to a VI.

6.20.4.2 Configuration Space Replication

The SR-IOV specification defines a reduced configuration space for the virtual functions. Most of the PCIe configuration of the VFs comes from the PF.

6.20.4.3 Memory BARs Assignment

The SR-IOV specification defines a fixed stride for all the VF BARs so that each VF can be allocated part of the memory BARs at a fixed stride from the a basic set of BARs. In this method, only two decoders per replicated BAR per PF are required and the BARs reflected to the VF are emulated by the VMM.

Since the only BARs that are useful for the VFs are BAR0 and BAR3, only those are replicated. Table 6-15 lists the existing BARs and the stride used for the VFs:

Table 6-15 BARs in FM10000

BAR	Type	Usage	Requested Size per VF (=Stride)
0, 1	Mem	CSR Space	Maximum (16 KB, page size). For page size, refer to Section 9.4.4.8 for more details.
2, 3	Mem	MSI-X	Maximum (16 KB, page size).
4, 5	Mem	Global Register Space	

BAR0 of the VFs are a sparse version of the original PF BAR and include only the register relevant to the VF. For more details refer to Section 6.11.

6.20.4.4 PCIe Capability Structure

The PCIe capability structure is shared between the PF and the VFs. The only relevant bits that are replicated are:

- Transaction pending
- Function Level Reset (FLR)

6.20.4.5 MSI-X Interrupts

Virtual functions interrupts are programmable via MSI-X using BAR2/3. Refer to Section 6.12 for more details.

6.20.4.6 VPD Capability

VPD is implemented only once and is accessible only from the PF.

6.20.4.7 Power Management Capability

The FM10000 does not support power management per VF. The power management registers exist for each VF, but only the D0 power state is supported.

6.20.4.8 Serial ID

The serial ID capability is not supported in VFs.

6.20.4.9 Error Reporting Capabilities (Advanced and Legacy)

All the bits in this capability structure are implemented only for the PF. Note that the VMs see an emulated version of this capability structure.

6.20.4.10 FLR Capability

The FLR bit is required per VF. Setting of this bit resets only a part of the logic dedicated to the specific VF and does not influence the shared part of the port. This reset should disable the queues, disable interrupts and the stop receive and transmit process per VF. Refer to Section 6.5.5 for details.

6.20.4.11 Error Reporting

Error reporting includes legacy error reporting and Advanced Error Reporting (AER) or role-based capability.

The legacy error management includes the following functions:

1. Error capabilities enablement. These are set by the PF for all the VFs. Narrower error reporting for a given VM can be achieved by filtering of the errors by the VMM. This includes:
    a. SERR# Enable
    b. Parity Error Response
    c. Correctable Reporting Enable
    d. Non-Fatal Reporting Enable
    e. Fatal Reporting Enable
    f. UR Reporting Enable

2. Error status in the configuration space. These should be set separately for each VF. This includes:
    a. Master Data Parity Error
    b. Signaled Target Abort
    c. Received Target Abort
    d. Master Abort
    e. SERR# Asserted
    f. Detected Parity Error
    g. Correctable Error Detected
    h. Non-Fatal Error Detected
    i. Unsupported Request Detected

AER capability includes the following functions:

1. Error capabilities enablement. The Error Mask, and Severity bits are set by the PF for all the VFs. Narrower error reporting for a given VM can be achieved by filtering of the errors by the VMM. These includes:
    a. Uncorrectable Error Mask register.
    b. Uncorrectable Error Severity register.
    c. Correctable Error Mask register.
    d. ECRC Generation Enable.
    e. ECRC Check Enable.
2. Non-Function Specific Errors Status in the configuration space.
    a. Non-Function Specific Errors are logged in the PF.
    b. Error logged in one register only.
    c. VI avoids touching all VFs to clear device level errors.
    d. The following errors are not function specific:
        - All Physical Layer errors.
        - All Link Layer errors.
        - ECRC fail.
        - UR, when caused by no function claiming a TLP.
        - Receiver overflow.
        - Flow Control Protocol error.
        - Malformed TLP.
        - Unexpected completion.
3. Function Specific Errors Status in the configuration space.
    a. Allows Per VF error detection and logging.
    b. Help with fault isolation.
    c. The following errors are function specific:

- Poisoned TLP received.
- Completion timeout.
- Completer abort.
- UR, when caused by a function that claims a TLP.
- ACS violation.

4. Error logging. Each VF has it’s own header log.
5. Error messages. To ease the detection of the source of the error, the error messages should be emitted using the requester ID of the VF in which the error occurred.

6.20.4.12 Alternative Routing ID (ARI) and IOV Capability Structures

To allow more than eight functions per end point without requesting an internal switch, as usually needed in virtualization scenarios, the PCI-SIG defines the ARI capability structure.

The FM10000 supports ARI capability and is enabled in the SR-IOV capabilities (PCIE_CFG_SRIOV_CTRL register) for proper addressing of the virtual functions when in SR-IOV mode. The sole decoding supported is as follows:

Device/Function	VF #
0	PF
1	0
2	1
.	.
.	.
.	.
64	63

6.20.4.13 Mailboxes

Mailboxes are available for PF⇔VF and PF⇔SM for communications between a PF and VF drivers, and between PF drivers and external Switch Managers. Each PCIe end point has its own set of mailboxes. This channel can be used for the PF driver to send command/status updates to the VFs (such as link change, memory parity error, etc.) or for the VF to send requests to the PF (add to VLAN, etc.), same for PF to SM vice versa.

Note: An external Switch Manager is a Switch Manager running on another host than the current directly attached host. The external Switch Manager receives interrupts through the PCI_INTERRUPT_OUT_SM from this PCIe interface, which gets routed to either the INT_N pin or routed to another PCI-Express via LSM. If routed to another PCI-Express, the signal is received via PCI_INTERRUPT_IN signal. This later one results in an MSI-X interrupt posted to the PF or VF.

The mailboxes are stored in PCIE_MBMEM[0..2047] (32-bit words) and used this way:

- 64 x 64-byte mailboxes, for communication with VFs (one 64-byte mailbox per VF)
- 1 x 4 KB mailbox for communication between PF and SM

The set of registers for this unit are:

- For PF
  - PCIE_MBMEM[0..2047] = Mailboxes
  - PCIE_MBX[0..63] = VF Mailboxes Control
  - PCIE_GMBX = Global Mailbox Control & Interrupt Status
  - PCIE_MBICR[0..3] = VF Mailboxes Interrupt Cause to MSI-X (PF)
  - PCIE_MBIMR[0..1] = VF Mailboxes Interrupt Mask to MSI-X (PF)
- For VF
  - PCIE_VFMBMEM[0..15] = VF view of its dedicated mailbox
  - PCIE_VFMBX = VF view of its dedicated mailbox control

Figure 6-29 shows the arrangement:

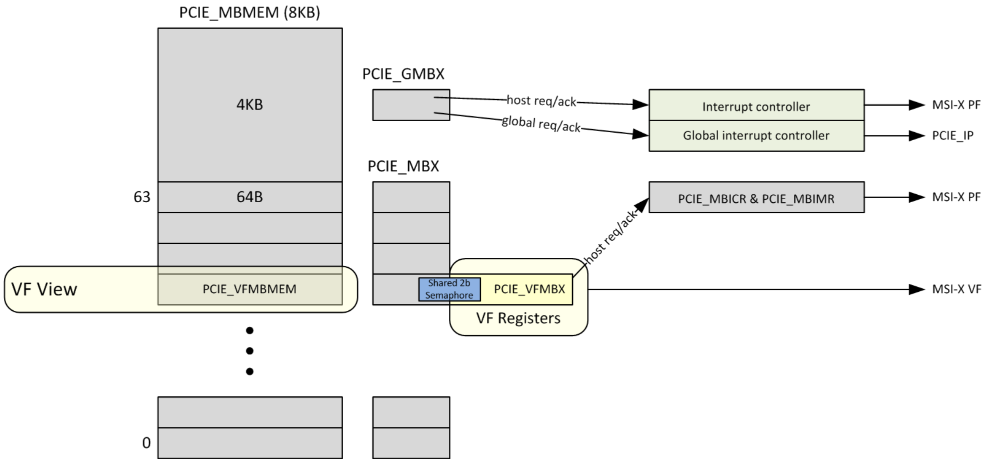


Figure 6-29 PCIe Mailbox Unit

The PF writes messages to VF into the PCIE_MBMEM[64*vf..64*vf+15] and writes messages to SM into the PCIE_MBMEM[1024..2047]. It announces message availability using the PCIE_MBX[vf] and PCIE_GMBX respectively.

The VF writes messages to PF into PCIE_VFMBMEM, which points to its mailbox. It announces message availability using the PCIE_VFMBX register which contains fields to route announcement to PF.

The SM writes messages to PF into the PCIE_GMBMEM. It announces message availability using the PCIE_GMBX. When it exits from reset, the SM releases the PF/SM mailboxes of all PEPs, just in case it was the owner before resetting.

An hardware semaphore is available for each mailbox to resolve collision. The hardware semaphore is 1-bit long and contained in PCIE_MBX and PCIE_GMBX control registers.

The process from software is the following:

Requester:

1. Requester reads the semaphore (MBX.Owner).
2. Requester won access if the bit is read as 1b.
3. If it did not won, the requester has to try later.
4. If it won, the requester writes the message and send a "request" interrupt to target and wait for an "ack" interrupt from target. Requests are announced by setting the MBX.*xxxReq* bit which result in a mailbox interrupt on the receiver and the request latched in the MBX registers (mirror on MBxICR registers for VF interrupts). Acknowledges are detected by reading the MBX registers (mirror on MBxICR registers for VF interrupts).

Receiver:

1. Wait for request interrupt.
2. Read message.
3. Clears the PCIE_MBX.*Owner* semaphore bit back to zero by writing 1b.
4. Write back ack. Acknowledges are announced by setting the MBX.*xxxAck* bit which result in a mailbox interrupt on the sender and the acknowledge latched in the MBX registers.

The hardware proceeds this way:

- If MBX.*Owner* is read and the current value is one, the value returned on that read is one, and the value is changed to zero. All subsequent reads return zero.
- If MBX.*Owner* is reset to one by host writing 1b. The value is set internally to one regardless of the current value (mailbox can be forced free by any source). Host writing zero has no effect.

Table 6-16 shows a step-by-step example for PF to VF.

Table 6-16 PF-to-VF Messaging Flow

Step	Event
1	PF reads PCIE_MBX[n].*Owner* and check that *Owner* is 1b. Otherwise waits at this step.
2	Hardware set Owner to 0b if PCIE_MBX[n].*Owner* was read as 1b.
3	PF writes message to relevant location in PCIE_MBMEM[n].
4	PF sets PCIE_MBX[n].*Req* bit.
5	Hardware indicates an interrupt to VF #n, setting PCIE_VFMBMEM.*ReqInterrupt* bit.
6	VF gets interrupt.
7	VF reads the message from VFMBMEM.
8	VF clears PCIE_VFMBX[n].*Owner* by writing 1b, *ReqInterrupt* and sets *PFAck*.
9	Hardware Indicates an interrupt to PF, setting the relevant PCIE_MBX[n].*PFAckInterrupt* bit.
10	PF clears PCIE_MBX[n].*PFAckInterrupt*.

Table 6-17 shows a step-by-step example for VF to PF.

Table 6-17 VF-to-PF Messaging Flow

Step	Event
1	VF reads PCIE_VFMBX.*Owner* and check that *Owner* is 1b. Otherwise wait at this step.
2	Hardware set *Owner* to 0b if PCIE_VFMBX.*Owner* was read as 1b.
3	VF writes message to PCIE_VFMBMEM.
4	VF sets PCIE_VFMBX.*Req* bit, setting PCIE_MBX[n].*PFReqInterrupt* bit.
5	Hardware indicates an interrupt to PF, setting the relevant bit in PCIE_MBICR.
6	PF gets interrupt and reads PCIE_MBICR to identify the VF(s).
7	PF reads the message from relevant location in PCIE_MBMEM[n].
8	PF clears PCIE_MBX[n].*Owner* by write 1b, *Req* and sets *Ack*.
9	Hardware Indicates an interrupt to VF, setting the PCIE_VFMBX.*AckInterrupt* bit.
10	VF clears PCIE_MBX[n].*PFAckInterrupt*.
11	PF polls PCIE_MBICR until the relevant bit is cleared, and going back to step 7 for handling mailbox messages from other VFs.

In any case, the content of the message is hardware independent and is determined by software.

6.20.5 Packet Switching

All packet switching functions are handled by the fabric.

6.20.6 Security Features

The FM10000 performs some security checks needed to provide isolation between VMs.

6.20.6.1 Inbound Security

Every packet posted to the host has passed the VLAN membership check before. Refer to Section 6.18.5 for more details.

6.20.6.2 Outbound Security

The FM10000's MAC table verifies that the combination of SMAC, VLAN, and SGLORT in Tx packets are valid, if not, it raises a MAC security issue.

The switch CAM table is configured to prevent a VF from issuing LLDP and other link layer packets on the wires.

6.20.6.3 Malicious Driver Protection

In SR-IOV scenarios the driver might be malicious or malfunctioning. The FM10000 PCIe end-point limits the impact of a mis-behaved driver to the concerned VM. The security checks and actions defined in this section are done toward the PF and the VF drivers indifferently. PCIE_LVMMC and PCIE_DEBUG_RHI contain information on the type of error detected.

6.20.6.3.1 Out of Range Memory Access

Host physical memory address space can be set from 48 up to 64 bits long. It is configured at init time into PCIE_CTRL.*PhyAddrSpace* field. When it is set to less than 64-bits, an out of range read access may cause system hang, and therefore such a request is not sent to host and is terminated internally as an Unsupported Request (UR). Out of range write access are just blocked internally, with no error reported to DMA.

6.20.6.3.2 Unsupported Request (UR) or Completer Abort (CA)

If a DMA request ended with one of these errors, the hardware drops the packet, disables the Queue, and moves the Queue ownership to PF. The PF driver can re-enable the Queue later on.

Note: Completion timeout is internally reported to DMA as UR (dummy).

6.20.6.3.3 Queue Context Validation

Any host write attempt to a queue context register field which does not respect the following rules is either silently discarded or not made possible:

- RDBAL.*RDBAL* must point to a 128 byte-aligned block of data
- RDLEN.*LEN* must be a multiple of 128 bytes and the minimum number of descriptors in a ring 128.
- RDT.*RDT* must not point to outside the descriptor ring boundaries (as per RDLEN.*LEN* field). In such a case, it saturates internally at the ring boundary.
- SRRCTL.*BSIZEPACKET* should not be set to 0x0 or above 16 KB.
- SRRCTL.*BSIZEHEADER* greater than 1024 bytes.
- SRRCTL.*DESCTYPE* set to 1 or 2 and *BSIZEHEADER* field set to zero.
- TDBAL.*TDBAL* must point to a 128 byte-aligned block of data.
- TDLEN.*LEN* must be a multiple of 128 bytes and the minimum number of descriptors in a ring 32.
- TDT.*TDT* must not point to outside the descriptor ring boundaries (as per TDLEN.*LEN* field). In such a case, it saturates internally at the ring boundary.

6.20.6.3.4 Queue Context Change on the Fly

The device does not perform a Queue context change while the Queue *ENABLE* bit is set. It concerns the following Queue parameters:

- RDBAL, RDBAH, RDLEN, SRRCTL
- RXDCTL.*MaxTime*, *WriteBackImm*
- TDBAL, TDBAH, TDLEN
- TXDCTL.*PushDesc*

6.20.6.3.5 Tx Parser

The Tx parser tolerates any packet format it encounters, even though it does not comply with the relevant standard. It may drop the malformed packet.

6.20.6.3.6 Tx Descriptor Validation

Tx DMA tolerates non-valid or non-consistent Tx descriptor contents, and drops the concerned packet(s), disables the queue and moves queue ownership to PF.

- Packet size < 17 bytes (and 25 bytes if FTAG is included) — This case is handled only by packet drop.
- Packet size > 15 KB (excluding FTAG).
  - Only entire 16-byte lines which are aligned to 16-byte address boundaries are stored in the 16 KB deep TCB. Therefore, it may be that TCB is not large enough to contain a misaligned Jumbo packet that spans over more than 32 descriptors. In such a case, the packet is dropped, the queue disabled, and its ownership moved to PF.
- HDRLEN = 0 and MSS > 0
- HDRLEN > 0 and MSS = 0
- HDRLEN > packet size or MSS or 192 bytes in TSO
- HDRLEN < 54
- MSS < MinMSS
- TSO > (256 KB - 1)
- BUFLEN = 0
- Number of descriptors for a packet or TSO > descriptor cache size.
- FTAG bit set and PFVTCTL[Q].*FTagDescEnable* bit cleared.

6.20.6.3.7 Rx Buffers Alignment

Rx data/header buffers are either 512-bytes aligned, or never cross host memory page boundaries. TLPs that would cross page boundaries are dropped internally before reaching the PCIe interface, causing data/header garbage to the malfunctioning/malicious VF.

7.0 Ethernet Port Logic (EPL)

7.1 Overview

Each EPL includes four 25 Gb/s SerDes with associated MAC and PCS layers to use the four SerDes lanes either independently (for port rates up to 25 GbE) or together (to support 40 GbE and 100 GbE). Each single-lane MAC is connected to the fabric through a 1.56 GHz x 16-bit channel. These four channels are used together for the multi-lane MAC to support 4 SerDes lanes on a single port. This is shown in Figure 7-1.

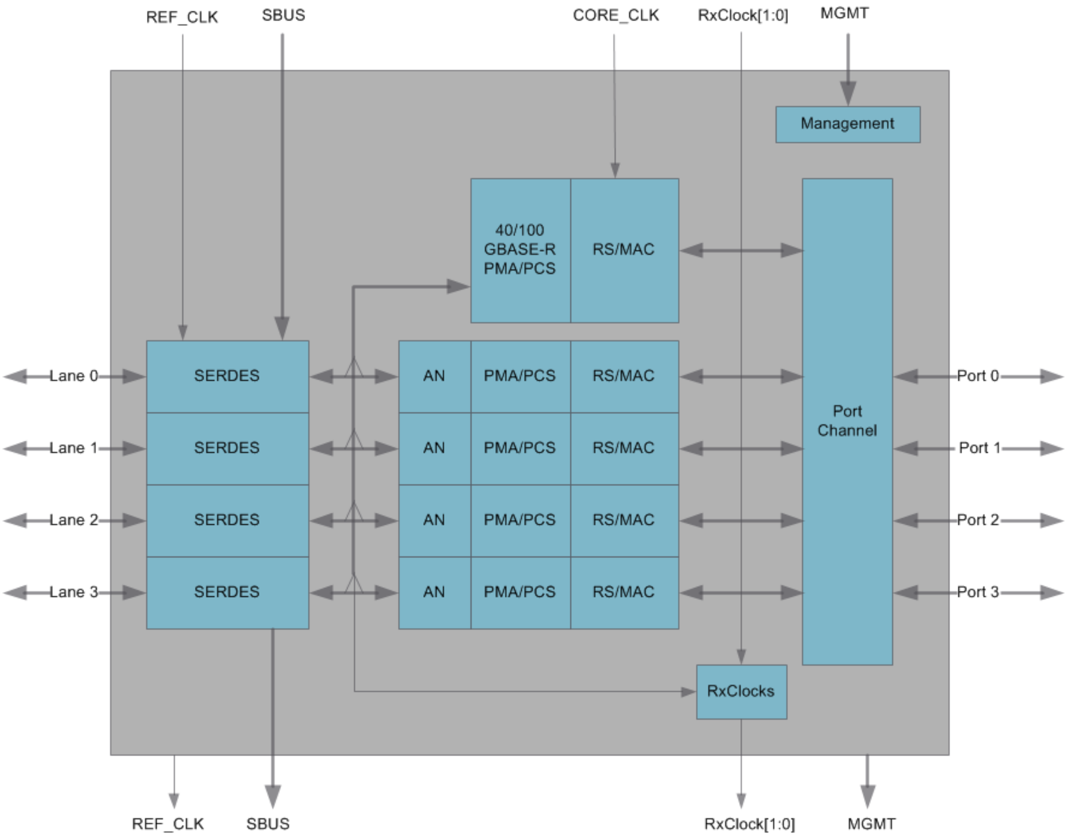


Figure 7-1 EPL Block Diagram

The Ethernet Port Logic includes the following elements:

SerDes/PCS

- Serial transmission and reception.
- 64b/66b encoding and decoding.
- 8b/10b encoding and decoding.
- Symbol locking.
- Lane polarity reversal.
- Lane alignment.
- Frame boundary decoding.
- Clause 73, 37 and SGMII auto-negotiation.
- BIST.

MAC

- Packet reception and transmission.
- CRC validation and CRC computation.
- Programmable handling of preamble.

Management interface

- SBUS interface to SPICO micro-controller.
- Management interface to processor.

Clocking

- One core clock (nominally 800 MHz, minimum 782 MHz).
- One reference clock to SerDes (156.25 MHz).
- Two recovered receive clocks.

The EPLs are daisy-chained together for management purposes as shown in Figure 7-2.

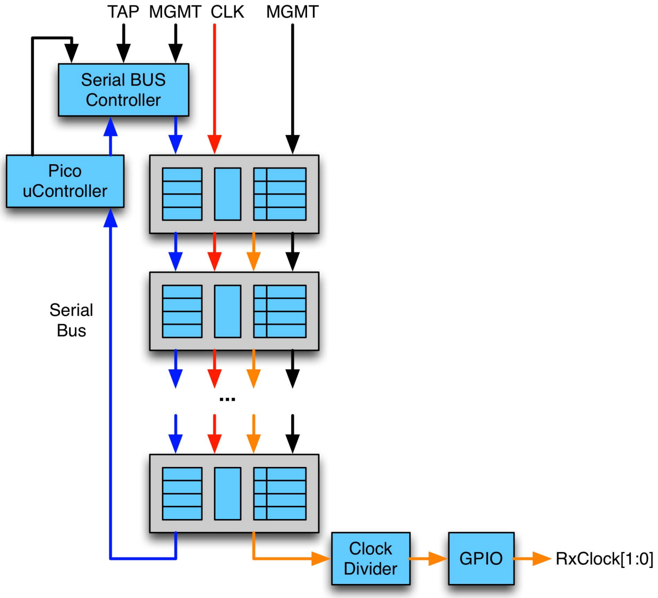


Figure 7-2 Multiple EPLs

There are two management accesses:

- Memory mapped for common Ethernet port registers.
- Serial bus for less frequently used registers.
  - — The serial bus is accessible via the SBUS controller (refer to Section 2.2).
  - — A micro-controller is also present in the chain to perform SerDes management.

7.2 Mode of Operation

The EPL supports the following modes of operation, and is compliant to the following standards:

Mode	Lanes	Standard	SerDes Bit Rate (Gb/s)	Encoding
SGMII	1	Cisco	1.25	8b/10b
1000Base-X	1	IEEE Clause 36,37	1.25	8b/10b
1000Base-KX	1	IEEE 802.3ap	1.25	8b/10b
10GBase-R (KR/CR/SR)	1	IEEE 802.3ap	10.3125	64b/66b
SFP+ 5M Direct Attach (10GBase-CR)	1	IEEE 802.3ae	10.3125	64b/66b
SFP+ SFI Interface (10GBase-SR)[1]	1	IEEE 802.3ae	10.3125	64b/66b
40GBase-KR4	4	IEEE 802.3ba	10.3125	64b/66b
QSFP 5M Direct Attach (40GBase-CR4)	4	IEEE 802.3ba	10.3125	64b/66b
QSFP PMD Service Interface (40GBase-SR4)	4	IEEE 802.3ba	10.3125	64b/66b
100GBase-KR4	4	IEEE 802.3bj	25.78125	256b/257b

Mode	Lanes	Standard	SerDes Bit Rate (Gb/s)	Encoding
100GBase-CR4	4	IEEE 802.3bj	25.78125	256b/257b
100GBase-SR4	4	IEEE 802.3bj	25.78125	256b/257b
2.5G non-standard	1	none	3.125	8b/10b
25G non-standard	1	none	25.78125	64b/66b

[1] Deterministic jitter generation is 150 mUI max verses SFI spec of 100 mUI max

The EPL can support any four single-lane modes of operation on different lanes simultaneously.

If the EPL is configured for multi-lane operation, any of the four lanes can be designated for auto-negotiation or for fall-back to single-lane operation. The lane polarity (+ or -) is always programmable per lane per direction.

7.3 Reference Clock

The Ethernet Port Logic requires one 156.25 MHz reference clock distributed to all EPLs.

7.4 SerDes Characteristics

The SerDes can operate at any of the following speeds:

- 25.78125 GbE, 64b/66b or 256/257-bit encoding
- 10.3125 GbE, 64/66-bit encoding
- 3.125 GbE, 8/10-bit encoding
- 1.25 GbE, 8/10-bit encoding

The following is a list of SerDes characteristics:

- Programmable pre and post emphasis.
- DFE (Decision Feedback Equalization).
- Symbol lock.
- Eye measurement.
- Auto adjustment via dedicated micro-controller.
- KR DFE training.
- Loopbacks.

7.4.1 DFE Tuning & Emphasis

The DFE tuning and transmit emphasis is static for 1.25 Gb/s. and 3.125 Gb/s

The DFE tuning is dynamic for 10.3125 Gb/s and 25.78125 Gb/s speeds. Dynamic tuning is controlled by the Spico micro-controller. For KR modes, the receiver DFE tuning and transmit emphasis can be executed automatically using the KR training sequence.

7.4.2 Pattern Generator/Comparator

Each SERDES lane has one Pattern generator and comparator. The patterns supported are:

- Polynomial
  - PRBS: x^7+x^6+1 (ITU V.29)
  - PRBS: $x^{15}+x^{14}+1$ (ITU-T O.151)
  - PRBS: $x^{23}+x^{18}+1$ (ITU-T O.151)
  - PRBS: $x^{31}+x^{28}+1$
  - PRBS: $x^{11}+x^9+1$ (IEEE Std 802.3ap-2007)
  - PRBS: x^9+x^5+1 (Fiber Channel)
- Fix pattern
  - 10/20/40/80 bits

Both link partners must be configured with the same pattern for comparator to work properly. Once configured, the comparator is capable of self-synchronizing to the incoming data, and also include a counter to count the number of errors detected. Software should first check synchronization has been detected and then reset the counter before monitoring a bit error rate.

7.4.3 Loopbacks

The SerDes supports two loopbacks:

- “Near” loopback where the TX serialized output is looped back into the RX serial input.
- “Far” loopback where the received parallel data is looped back into the transmit parallel data.
  - The data is placed in a small FIFO for clock boundary crossing but the TX must be traceable to the same clock source as the link partner for this feature to work properly.

7.4.4 Eye Measurement

The SerDes support eye measurement.

7.5 Recovered Clocks

The FM10000 offers support for up to two recovered clocks. The two recovered received clocks are daisy chained from one EPL lane to the next and passed finally to two clock dividers at the end of the chain and presented to two dedicated pins. The LANE_CFG register defines if the local clock is used or the previous one is passed through, the LANE_CFG also defines a divider (by 2 or by 4).

7.6 Auto-negotiation

Auto-negotiation as defined in clause 37 and clause 73, and the industry de-facto SGMII auto-negotiation mode. The clause 73 is a standalone layer while the clause 37/SGMII are attached to the PCS layer and controlled through the normal memory-mapped management interface.

7.6.1 Clause 73

Clause 73 uses a relatively low-frequency to encode the auto-negotiation messages. Each message is 48 bits long and is encoded in a 106-bit frames using a Manchester encoding scheme (see Figure 7-3). These frames are exchanged between link partners before the link comes up as the speed and mode of operation is not determined yet. The clause 73 requires a 125 MHz reference clock.

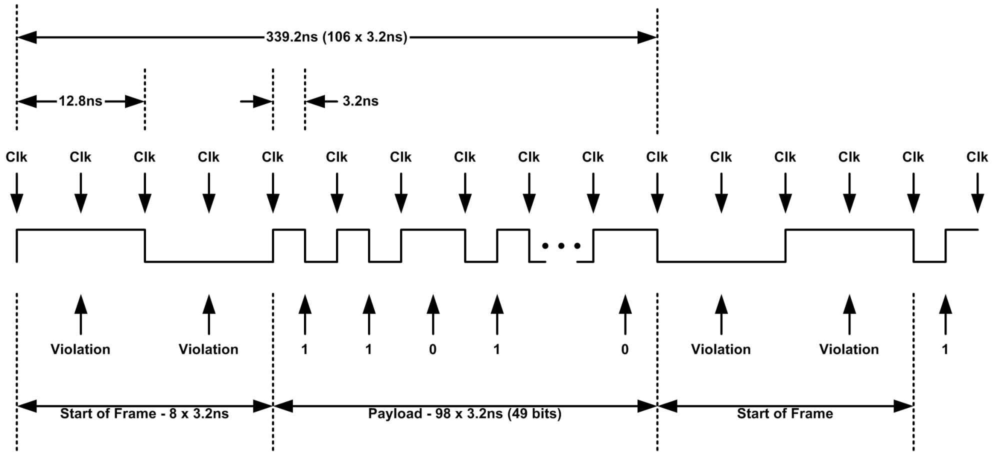


Figure 7-3 EPL Clause 73

The auto-negotiation process is detailed in clause 73 and summarized here:

1. The AN state-machine sends the mandatory control word frame repetitively setting the *ACK* bit to 0b. The control word includes an ID, which is a pseudo random number set by the hardware which is static for the duration of this negotiation. The control word includes an NP bit to indicate if the sender has more pages to send.
2. The AN expects to receive the mandatory control word and waits for 3 consecutive identical copies of the control word to be received.
3. The AN sends an acknowledge (*ACK* set to 1b) and echoed the ID from the link partner.

4. The AN waits for an *ACK* with an echoed ID equal to the one sent. The receiver checks for *NP* and if set enters the next page negotiation.
5. If neither the local end nor the link partner has more page to send, the AN is completed.
6. If one or both has next pages to send, they do so at that point, starting first by sending the *ACK* bit to 0b and then waiting for a valid next page before sending the next page. If one side as no next page to send, it sends the NULL page.

The control word format for Clause 73 is listed in Table 7-1.

Table 7-1 EPL Clause 73 Control Word Format

Bits	Name	Transmitter	Receiver
4:0	Selector	Fixed (00001). Cannot be set by software.	Checked by hardware.
9:5	Echoed Nonce	Transmitter echoes received "transmitted Nonce" in this field once three valid identical CW have been received.	Checked by hardware to be equal to the transmitted "transmitted Nonce" (if ACK is 1).
12:10	Pause Abilities	Set by software.	Pause abilities. Read and interpreted by software once negotiation exchange is completed.
13	Fault	Set by software.	Fault at link partner. Read and interpreted by software once negotiation exchange is completed.
14	ACK	Software presets if needed and hardware sets at appropriate time. Set by hardware once three valid identical CW have been received.	Check by hardware.
15	NP	Software presets if needed and hardware sets at appropriate time. Set by hardware once three valid identical CW have been received.	Check by hardware.
20:16	Tx Nonce	ID for this negotiation session. Set by hardware after each restart.	Link partner ID for this negotiation session.
45:41	Mode Abilities	Abilities of this transmitter. Set by software.	Abilities of link partner. Read and interpreted by software.
47:46	FEC Abilities	FEC abilities of this transmitter. Set by software.	FEC abilities of link partner. Read and interpreted by software.

The next page format for Clause 73 is listed inTable 7-2.

Table 7-2 EPL Clause 73 Next Page Format

Bits	Name	Transmitter	Receiver
10:0	Message	Set by software.	Read and interpreted by software once negotiation exchange is completed.
11	Toggle	Set by software.	Checked by hardware to be toggling.
12	ACK2	Set by software.	Read and interpreted by software once negotiation exchange is completed.
13	MP	Set by software.	Read and interpreted by software once negotiation exchange is completed.
14	ACK	Software presets if needed and hardware follows once three valid identical CW have been received.	Check by hardware.

Table 7-2 EPL Clause 73 Next Page Format [Continued]

Bits	Name	Transmitter	Receiver
15	NP	Software presets if needed and hardware follows once three valid identical CW have been received.	Check by hardware.
47:16	Message	Set by software.	Read and interpreted by software.

The details of the algorithm are as follows.

1. Software sets the MAC in AN73 mode. Software announces its local abilities by loading the control word into the AN_73_BASE_PAGE_TX register. The *NextPage* bit in the CW is set if there are other pages to transmit. Software enables AN, starts a timer and waits.
2. The switch exchanges the CW with its partner. The switch transmits the AN_73_BASE_PAGE_TX as the control word repetitively with *ACK* bit to 0b. The switch waits for three identical valid control words and stores them into the AN_73_BASE_PAGE_RX register. When three identical valid control words are received, the switch echoes the NONCE received into its own CW message, sets the *ACK* bit to 1b. If the switch detects the NONCE received is equal to the NONCE transmitted, it posts a An73TransmitDisable interrupt, and stops and waits for software intervention, which changes the NONCE to a new random value. If the NONCE are different, the switch waits for an ACK to be received. When an *ACK* is received, the switch posts an *An73CompleteAcknowledge* interrupt.
3. Software detects the *An73CompleteAcknowledge* interrupt which indicates that a page has been received and can now read the AN_73_BASE_PAGE_RX register (or AN_73_NEXT_PAGE_RX register is these are followed on pages). If there are next pages, software loads the AN_73_NEXT_PAGE_TX register with the next page to send and sets the *TxNextPageLoaded* bit. If there is no next page, go to Step 6.
4. The switch exchanges the NP with its partner. The switch detects the write to *TxNextPageLoaded* bit, clears it, and starts to transmit the next page. It first transmits with the *ACK* bit to 0b until the hardware receives three identical *NextPage* messages. The switch detects three identical NextPage message, then sets its ACK to 1b and stores the receive page into AN_73_NEXT_PAGE_RX register. The switch waits for three consecutive *ACK*s. The message in AN_73_NEXT_PAGE_RX is updated. When an *ACK* is received, the switch posts an *An73CompleteAcknowledge* interrupt.
5. Steps 3 and 4 are repeated until all pages have been exchanged.
6. The switch posts an *An73AnGoodCheck* interrupt to indicate that exchange of pages is completed and waits for the link to come up.
7. Software detects the interrupt and then applies the highest common denominator. Software is responsible to reprogram the SerDes at the proper speed, load the PCS mode negotiated into the AN73 PCS mode register and set appropriate timer for this mode to sync.
8. The switch posts an *An73AnGood* when the link comes up in time, or posts an *An73TransmitDisable* interrupt if the link does come up before the timer runs out.
9. Software detects *An73AnGood* and can now declare the AN completed. If it was an *An73TransmitDisable*, the state machine returns automatically to Step 2.

7.6.2 Clause 37

The Clause 37 is a sub-mode of the 1000Base-X operating mode. It allows exchanging abilities with the remote device such as duplex mode, flow control or fault signaling.

The auto-negotiation process is detailed in clause 37 and summarized here:

1. The AN state-machine sends the mandatory control word frame repetitively setting the *ACK* bit to 0b. The control word includes an *NP* bit to indicate if the sender has more pages to send.
2. The AN expects to receive the mandatory control word and waits for three consecutive identical copies of the control word to be received.
3. The AN sends an acknowledge (ACK set to 1b) and echoed the ID from the link partner. The message is sent for at least 10 ms.
4. The AN waits for an ACK to be received for at least 10 ms. The receiver checks for *NP* and if set enters the next page negotiation.
5. If neither the local end nor the link partner has more page to send, the AN is completed.
6. If one or both has next pages to send, they do so at that point, starting first by sending the *ACK* bit to 0b and then waiting for a valid next page before sending the next page. If one side as no next page to send, it sends the NULL page.

The 1000Base-X has the ability to enter into clause 37 either manually via software intervention or automatically upon link down detection or by detecting AN messages from remote devices. If programmed to start clause 37 upon link down, the clause 37 announces a “remote fault” and maintains the announcement until the link is up. The “remote fault” is cleared when the link becomes up and the handshake can now complete.

The control word format for Clause 37 is listed in Table 7-3:

Table 7-3 EPL Clause 37 Control Word Format

Bits	Name	Transmitter	Receiver
4:0	Selector	Set by software.	Checked by software.
6:5	Duplex	Set by software.	Duplex abilities. Read and interpreted by software once negotiation exchange is completed.
7:8	Pause	Set by software.	Pause abilities. Read and interpreted by software once negotiation exchange is completed.
9:11	Reserved	Set by software.	Checked by software.
12:13	Fault	Set by software or hardware.	For transmit, either set by software or hardware. For receive, always interpreted by software.
14	ACK	Software presets if needed and hardware sets at appropriate time.	Set by hardware once 3 valid identical CW have been received. Check by hardware.
15	NP	Software presets if needed and hardware sets at appropriate time.	Set by hardware once 3 valid identical CW have been received. Check by hardware.

The next page format for Clause 37 is listed in Table 7-4:

Table 7-4 EPL Clause 37 Next Page Format

Bits	Name	Transmitter	Receiver
10:0	Message	Set by software.	Read and interpreted by software once negotiation exchange is completed.
11	Toggle	Set by software.	Checked by hardware to be toggling.
12	ACK2	Set by software.	Read and interpreted by software once negotiation exchange is completed.
13	MP	Set by software.	Read and interpreted by software once negotiation exchange is completed.
14	ACK	Software presets if needed and hardware follows once 3 valid identical CW have been received.	Check by hardware.
15	NP	Software presets if needed and hardware follows once 3 valid identical CW have been received.	Check by hardware.

The control word format for SGMII is listed in Table 7-5:

Table 7-5 EPL Clause 37 SGMII Control Word Format

Bits	Name	Switch	PHY
0	1	Set by software to 1b.	Report 1. Validated by software.
9:1	Reserved	Set by software to 0b.	Report 0. Validated by software.
11:10	Speed	Set by software to 0b.	Report speed: 00b = 10 Mb/s - Can be applied by hardware. 01b = 100 Mb/s 10b = 1000 Mb/s 11b = Reserved
12	Duplex	Set by software to 0b.	Duplex mode. Validated by software. Hardware ignore this value and always assume full duplex.
13	Reserved	Set by software to 0b.	Checked by software.
14	ACK	Software presets if needed and hardware sets at appropriate time. Set by hardware once 3 valid identical CW have been received.	Check by hardware.
15	LinkUp	Set by software.	Checked by software.

The details of the algorithm are as follows:

1. Software sets the MAC in 1000BaseX mode. Software announce its local abilities by loading the control word into the AN_73_BASE_PAGE_TX register. The *NextPage* bit in the CW is set if there are other pages to transmit. Software enables AN37, starts a timer and waits.
2. The switch exchanges the CW with its partner. The switch transmits the AN_37_BASE_PAGE_TX as the control word repetitively with *ACK* bit to 0b. The switch waits for 3 identical valid control words and stores them into AN_37_BASE_PAGE_RX. When an *ACK* is received, the switch posts an *An37CompleteAcknowledge* interrupt.

3. Software detects the *An73CompleteAcknowledge* interrupt which indicates that a page has been received and can now read the AN_37_BASE_PAGE_RX register (or AN_37_NEXT_PAGE_RX register is these are followed on pages). If there are next pages, software loads the AN_37_NEXT_PAGE_TX register with the next page to send and flips the *ToggleNpLoaded* bit. If there are no next page, go to Step 6.
4. The switch exchanges the *NP* with its partner. The switch detects the change into the *ToggleNpLoaded* bit and starts to transmit the next page. It first transmits with the *ACK* bit to 0b until the hardware receives 3 identical *NextPage* messages. The switch detects 3 identical *NextPage* message, then sets its *ACK* to 1b and stores the receive page into AN_37_NEXT_PAGE_RX register. The switch waits for 3 consecutive *ACK*s. The message in AN_37_NEXT_PAGE_RX is updated. When an *ACK* is received, the switch post an *An73CompleteAcknowledge* interrupt.
5. Steps 3,4 is repeated until all pages have been exchanged.
6. The switch posts an *An73IdleDetect* interrupt to indicate that exchange of pages is completed.
7. Software detects the interrupt and then applies the highest common denominator.

The exchange can potentially be initiated by a link down event, the algorithm in this case is:

- Software has pre-configured the AN for this event.
- The switch detects a link down, starts to transmit the CW indefinitely with remote fault set.
- The switch detects a link up, clears the remote fault and start the process described above.

The exchange could potentially be initiated by the link partner. The algorithm in this case is:

- Software has pre-configured the AN for this event.
- The switch detects arrival of 3 consecutive AN37 messages.
- The switch starts the process described above.

7.6.3 SGMII

The SGMII is a de-generated mode of the 1000Base-X clause 37. It allows a 10/1000/1000BaseT PHY to announce to the switch the negotiated values with its link partner. In this mode, the switch is passive and only replies, it never initiates.

The auto-negotiation process is detailed in the Cisco SGMII specification and summarized here:

1. The AN waits for a message from PHY and operates normally until such message is received.
2. The AN state-machine detect reception of 3 consecutive identical messages from PHY.
3. The AN start a Tx timer and sends an ACK for the length of this timer.
4. The AN restart the Tx timer if the message receives changes but otherwise continue operation.
5. After the Tx timer completes, the AN checks if there is a change in the announcement versus the current mode. If yes, it post an interrupt to announce the change. If this is a speed change, the AN can be programmed to automatically apply the speed change without software intervention.

The control word format for SGMII listed in Table 7-6:

Table 7-6 EPL SGMII Control Word Format

Bit(s)	Name	Receiver
0	1	Set by software and PHY to 1b. Should be validated by s/w but otherwise ignored by the switch.
9:1	Reserved	Set by software and PHY to 0b. Should be validated by s/w but otherwise ignored by the switch.
11:10	Speed	Set by software to 0b for the switch. The PHY report the speed through this field: 00b = 10 Mb/s 01b = 100 Mb/s 10b = 1000 Mb/s 11b = Reserved The switch can be programmed to apply the speed change automatically. If the speed is not 1000 Mb/s, 100 Mb/s or 10 Mb/s, the AN is restarted.
12	Duplex	Set by software to 0b. PHY reports value negotiated with remote. Should be validated by software but otherwise ignored by the switch. The switch only supports full duplex and does not do anything different regardless of the value of this bit.
13	Reserved	Set by software and PHY to 0b. Should be validated by s/w but otherwise ignored by the switch.
14	ACK	Software sets to 0b and hardware sets at appropriate time. Check by hardware.
15	LinkUp	Set by PHY Checked by switch. Note that this bit is included in the overall link up / link down status reported by the MAC to software.

The algorithm is:

1. Software has pre-configured the AN for this mode. Software sets a transmit timer value (TX repeat time). Software enables AN and waits.
2. The switch operate normally until 3 identical messages are received. The switch starts a timer and transmits the CW for the duration of that timer. The *ACK* bit is set. The switch waits for the *ACK* bit to be received to 1b while running the Tx timer; the updated word is stored in AN_37_NEXT_PAGE_RX. After the Tx timer terminates, and if ACK was received, hardware posts an AN_DONE interrupt and either applies the speed (if programmed to do so) or post a link up/down event (if it changes).
3. Software detects the interrupt and applies the values detected if needed.

7.7 Physical Coding Sub-layer (PCS)

This section provides details on the Physical Coding Sub-layer (PCS).

7.7.1 100GBase-R4

The PCS layer for 100GBase-R4 (KR4/SR4/CR4) is designed to inter-operate with IEEE 802.3bj devices for backplane, cables, and multi-mode optics, provided the channels do not need RS-FEC corrector to reach 802.3 target BER targets.

The FM10000 100GBase-R4 encoder includes the Reed-Solomon Forward Error Correction code as required by the standard. The decoder supports the following two modes of operation:

- **Count RS-FEC errors (lowest latency)** — The RS-FEC is computed using incoming data and compared to the RS-FEC code in the data bock received, if there is a difference, the error is counted. No action is taken against any data if an error is detected, the frame CRC being considered good enough to protect against data error. This mode of operation is lower latency as the frame can be forwarded to the fabric while the FEC code is computed for a given block.
- **Count RS-FEC errors and mark frames affected as errored (higher latency)** — The RS-FEC is computed using incoming data and compared to the RS-FEC code in the data block received. If there is a difference, the error is counted and any frame having a portion of its content in the data block is marked as having a frame error, and eventually transmitted with a bad CRC when forwarded by the switch. This mode of operation requires the EPL to delay the forwarding of any frames by the size of the RS-FEC block and thus higher latency is expected.

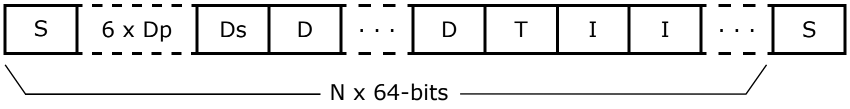
The FM10000 receiver implementation of 100GBase-R4 does not support RS-FEC correction, only encoding, decoding and checking. There are no attempts to correct any errors detected by the RS-FEC decoder.

The 100GBase-R4 supports automatic lane ordering.

7.7.2 40GBase-R4

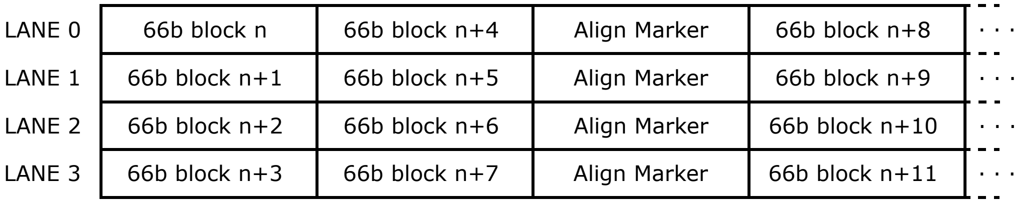
The PCS layer for 40GBase-R4 (KR4/SR4/CR4) is designed to meet the IEEE 802.3ba, clause 80, 81, 82, 83 and 84.

The frame format in 40 GbE mode follows. The value of Dp (Data preamble) is 55h (symbol D21.2), the value of Ds (Data start) is D5h (symbol D21.6). The only mode of operation is to accept frames that start with S, 6xDp and Ds as shown.



These frames are chopped into 64-bit chunks and the number of /I/ bytes added must be such that the 'S' byte always start a new 64-bit chunk.

The 64-bit chunks are then encoded into a 66-bit block which are distributed to four lanes using a round-robin distribution as shown below. An alignment marker is inserted after 16383 columns.



LANE 0	66b block n	66b block n+4	Align Marker	66b block n+8	. . .
LANE 1	66b block n+1	66b block n+5	Align Marker	66b block n+9	. . .
LANE 2	66b block n+2	66b block n+6	Align Marker	66b block n+10	. . .
LANE 3	66b block n+3	66b block n+7	Align Marker	66b block n+11	. . .

The addition of the alignment marker would cause a decrease of throughput, but is compensated by removing one column of IFG within the next 16383 columns.

The alignment marker includes block parity checking (BIP) that verifies the parity of one lane over every 16383 blocks. The transmitter always generates the correct BIP and the receiver verifies it and reports any error in the BIP error counter. The BIP checking does not start until the lane alignment is achieved.

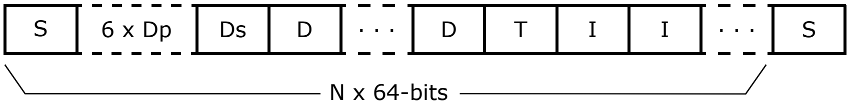
The lane ordering is automatically discovered using the alignment marker, and the mapping of lanes from an external device to the switch may follow an arbitrary lane mapping. The EPL has the capability to override the automatic lane ordering with a manual method.

The receiver lane de-skew logic is designed to handle lane skew up to 1856 bits (180 ns).

The normal lane line rate is 10.3125 Gb/s.

7.7.3 10GBase-R

The frame format in 10GBase-R (KR/SR/CR) serial mode follows. The value of Dp (Data preamble) is 55h (symbol D21.2), the value of Ds (Data start) is D5h (symbol D21.6). The default mode of operation is to accept only frames that starts with S, 6xDp and Ds as shown.

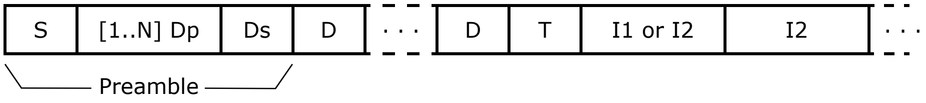


Configurable options in the MAC allow the preamble to be treated as data.

The normal lane line rate for this interface is 10.3125 Gb/s. Note that an alternative rate of 25.78125 Gb/s is also supported for this mode.

7.7.4 1000Base-X Frame Format

The frame format in 1000Base-X mode (or SGMII) follows.



The 1000Base-X does not allow preamble to be use as data. The preamble can be of any size on ingress but must at least have a /S/ symbol, one or more /Dp/ and a /Ds/ symbol. The receiver first checks for /S/, then skips all data bytes until /Ds/ is detected and then start to capture the frame. For egress, the PCS transmitter terminates the current idle sequence before starting a new frame. Therefore, the preamble is reduced by 1 byte (removing one /Dp/) if the MAC presented the frame to the PCS layer in the middle of an idle sequence. The minimum IFG configurable is 4 bytes, which can get reduced to 3 bytes (/S/, /Dp/, /Ds/) due to alignment of idle sequence.

In the 100 Mb/s mode, PCS searches for /S/ and then samples incoming data every 10 cycles. In the 10 Mb/s mode, PCS samples incoming data every 100 cycles.

7.7.5 Link Status

The PCS layer reports if a link is up or down. The link status are fully de-bounced with a different internal timer for link going down and link going up. An interrupt is posted whenever the link status changes. A link is declared up when:

- Symbol locks has been achieved.
- Lane alignment has been achieved.
- Valid idle sequences are received.
- No local faults are reported.
- No remote faults are reported.

Any of these conditions can be enabled or disabled by software. The default is for all conditions to be enabled.

7.7.6 FSIG

The 10 GbE and 40 GbE PCS function has the capability to enable transmission of a software configurable FSIG sequence. The same PCS have the ability to receive an FSIG sequence at any time regardless of whether the transmission of an FSIG sequence is enabled or not. LINK_IP[epl,port].*MacRxSequenceOrFsif* is generated for reception of FSIG, the last FSIG value received is simply saved in register RX_SEQUENCE.

7.7.7 IFGs

On transmit, it is necessary for the transmitter to modify the length of the interframe gap (IFG) to align the Start control character to the next boundary of 32 bits for 10 GbE (first octet of the frame must be on lane 0) and 64 bits for 40 GbE.

This is accomplished in one of the following two ways for 1 GbE and 10 GbE:

- **Guarantee minimum IFG** — The MAC inserts additional idle characters to align the start of preamble on a four byte boundary (0,1,2 or 3 bytes added) and then proceed in adding IFGs in group of 4. If the unit is configured for minimum of x12, the number of IFGs could actually be 12, 13, 14, 15 depending of frame size.

- **Meet average minimum IFG** — Alternatively, the transmitter may maintain the effective data rate by sometimes inserting and sometimes deleting idle characters to align the Start control character. When using this method the RS must maintain a Deficit Idle Count (DIC) that represents the cumulative count of idle characters deleted or inserted. The DIC is incremented for each idle character deleted, decremented for each idle character inserted, and the decision of whether to insert or delete idle characters is constrained by bounding the DIC to a minimum value of zero and maximum value of three. Note that this may result in inter-frame spacing observed on the transmit XGMII that is up to three octets shorter than the minimum transmitted inter-frame spacing specified in Clause 4; however, the frequency of shortened inter-frame spacing is constrained by the DIC rules. The DIC is only reset at initialization and is applied regardless of the size of the IPG transmitted by the MAC sublayer. An equivalent technique may be employed to control RS alignment of the Start control character provided that the result is the same as if the RS implemented DIC as described.

The devices support both methods.

For the “guarantee minimum IFG”, the minimum value supported by the transmitter is 8 for 10GBaseR/40GBaseR and 4 for SGMII/1000BaseX/10GBaseX. However, this value may not guarantee reliable operation as there may not be enough idles for the link partner to maintain integrity of the communication depending how the link partner implemented clock compensation. For the “meet average minimum IFG”, the minimum value supported by the transmitter is 9 for 10GBaseR/40GBaseR and 5 for SGMII/1000BaseX/10GBaseX. The recommended mode of operation is 12 with Deficit Idle Count enabled.

The 20/40 GbE interface includes the following extra functions:

- The Start control character must be aligned to the next 64-bit boundary. Therefore, the IFGs can only be added in increment of 8, the Deficit Idle Count is designed to handle this case.
- The addition of an alignment marker column periodically is compensated by deleting columns of /I/ between the markers. This is supported on receive and on transmit. For transmit, the compensation is done by configuration the clock compensation circuit to include deletion of idles at a rate corresponding to the insertion of markers.

Finally, an extra circuit for clock compensation is available that cuts IFGs periodically. The period is programmable between [1..65536].

7.7.8 Changing PCS Mode

PCS mode can be changed from any mode to any mode on a port by port basis. The PCS mode is configured in EPL_CFG_B register. The correct method is to first set to PCS_DISABLE and then change to the desired mode. The correct method for switching to or from 40 GbE or 100 GbE multi-lane is to first set the PCS mode to DISABLE on all four ports, then update the EPL_CFG_B.*QplMode* to the desired configuration and then update the EPL_CFG_B again with the desired PCS mode.

7.8 MAC

The MAC layer is responsible for the following actions:

- In the receive direction:
  - — Detect start of frame.
  - — Writes the frame into memory per Preamble and CRC optional processing.
  - — Compute CRC on frame payload.
    - The scheduler pushes segments pointers ahead of time.
    - MAC writes the packet to memory as it is received.
  - — Validates the CRC and frame length at the end of frame and indicate to the frame control if the frame received is good or bad. A frame is bad if any of the following condition are met.
    - Frame has a bad CRC.
    - Frame is too long, in this case the frame is terminated within a few bytes after at the maximum length configured has been reached, the MAC checks the CRC of oversized packets and records the statistics of how many long packets were received with correct CRC, and how many were received with bad CRC.
    - Frame is too short or has a symbol errors. In this case, the frame is terminated immediately and marked as bad, the MAC also enter a mode where it ignores the data received from the PCS until the minimum frame size has been reached. This prevents the MAC to overload the switch with tiny frames.
    - There was an overrun to the main data memory while the frame was written.
    - The link went down while a frame was received.
- In the transmit direction:
  - — The MAC receives the new frame (modified if needed) from the egress modifier.
  - — The transmitter first transmits the preamble, unless the preamble is part of packet payload, then transmits the packet payload and computes a new CRC as the frame is being transmitted. If the frame is shorter than the minimum allowed, the frame is padded with 0s to the minimum packet size, and then the last four bytes are replaced with CRC.
  - — The computed CRC replaces the original CRC in the frame. The CRC is inverted if the frame is flagged as bad by the egress modifier. This can happen if the frame was originally received bad but was already cut-through or if an irrecoverable ECC error occurred while the frame was retrieved from memory. Padding to minimum frame size is done regardless if the frame is being transmitted with a good or a bad CRC.

The MAC includes the following general transmit/receiver options:

- Transmitter:
  - — Force the port to drain all packets regardless of link status. The drain is applied immediately and may cut short an in-flight frame.

    Note: Packets can only be drained when the interface between switch and MAC are configured to compatible modes.
  - — Force the port to hold any packets regardless of link status. The hold is apply cleanly at the beginning of a frame.

- Port drains all packets automatically when link is down.
- Port holds transmission when link is down.

- Receiver:
  - Drain all packets.

7.8.1 Preamble and CRC Optional Processing

The MAC has optional features for the processing of the preamble and CRC computations allowing Intel® Ethernet Switch Family to support custom hardware. The options are configurable per MAC and only available for 10 GbE, 40 GbE, and 100 GbE interfaces. These options are not available for SGMII or 1000Base-X.

Preamble options:

- Allow the receiver to either strip the preamble (default) or pass it to fabric.
- Allow the transmitter to either generate preamble locally or pass through what comes from the fabric. If the selection is to pass through the preamble, the transmitter forces the first byte of the pre-amble to an SDF symbol.

CRC options:

- The CRC computation has two options; the CRC computation may be started at pre-amble or right after pre-amble. The default is after pre-amble as defined by IEEE. If the CRC is started at pre-amble, the first byte of the frame is replaced with a configurable fixed value before the CRC is computed. The option of starting CRC at preamble is only allowed if the preamble is captured and passed to the fabric.

7.8.2 Packet Generation

The MAC includes a CJAPT packet generation capabilities for testing. The CJPAT pattern is:

- START/PREAMBLE/SFD:
  - <FB> 55 55 55 55 55 55 D5
- Payload:
  - 0B for 1 Octet (lane 0)
  - 7E for 3 Octets (lanes 1, 2, 3)
  - 7E for 524 Octets — Low Density Transition Pattern
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump

  - AB for 4 Octets — Phase Jump
  - B5 for 160 Octets — High Density Transition Pattern
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - 7E for 528 Octets — Low-Density Transition Pattern
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - AB for 4 Octets — Phase Jump
  - B5 for 160 Octets — High-Density Transition Pattern
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
  - EB for 4 Octets — Phase Jump
  - F4 for 4 Octets — Phase Jump
- RC
  - BD 9F 1E AB
- IFG
  - <FD> followed by 11 idles

The CJPAT is only available for single lane operations (1 GbE, 2.5 GbE, 10 GbE, 25 GbE), it is not available for quad lane modes (40 GbE and 100 GbE).

7.8.3 Reception Errors

The receiver checks various conditions while it is receiving a frame, and posts an end-of-frame status to the switch fabric at the end of the frame. The end-of-frame status could be one of the following:

- The frame was received with a good CRC and no other errors were encountered
- The frame was received with a bad CRC
- The frame reception was aborted for one of the following reason:
  - Frame is smaller than minimum size
  - Oversized
  - Disparity decoding error
  - Invalid symbol
  - Unexpected control symbol
  - Malformed preamble
  - Link fault detected
  - Link went down
  - MAC mode change
  - Overflow

Some of the conditions previously listed can be individually enabled or disabled by software and may also be available as interrupts. Table 7-7 lists all conditions and what type of control is available for each one of them and if an interrupt is available to detect this particular condition.

Table 7-7 EPL Conditions (1)

Error Type	Interrupt	Control
Bad CRC	Yes	Software can disable CRC checking.
Undersized	No	Software can configure minimum size.
Oversized	No	Software can configure maximum size.
Disparity error[1]	Yes	Software can disable disparity error checking/
Invalid symbol[1]	Yes	Always enabled. Software can select between two dispositions: 1. Terminate frame immediately. 2. Replace data with 0xFE and continue to receive frame until end of frame detected. ***Note:*** If disposition 2, the CRC is most likely invalid.
Unexpected control symbol	Yes	Always enabled.
Link fault	Yes	Always enabled.
Preamble error	No	Always enabled.
Link fault detected	No	Always enabled.
MAC mode change	No	Always enabled.
Overflow	No	Always enabled.

[1] The disparity errors and symbol errors interrupts can also be raised during idle periods.

The switch includes a centralized statistics module which uses the end-of-frame condition and the frame length to maintain per-port counters such as runt, undersized, frame size binning, etc. However, due to the limited end-of-frame status posted to the fabric, some error conditions gets accounted in the MAC itself.

7.8.4 Counters

The MAC includes the following counters:

- Overflows (32-bit, rollover)
  - When the data cannot be pushed to the fabric.
- Underflows (32-bit, rollover)
  - When the data does not arrive fast enough from the fabric.
- Jabber Packets (32-bit, rollover)
  - Oversized packets which end up having a bad CRC or a framing error at the end.
- Oversized Packets (32-bit, rollover)
  - Oversized packets which end up having a good CRC at the end.
- Runt Packets / Drained packets (32-bit, rollover)
  - Runt packets which end up having a bad CRC or a framing error at the end.
  - Also used as a drained packets counters when the receiver is in force drain mode.
- Undersized Packets (32-bit, rollover)
  - Undersized packets which end up having a good CRC at the end.
- Number of No FAULT events (12-bit, rollover)
  - Transmission to No Fault.
- Number of LOCAL FAULT events: (10-bit, rollover)
  - Incremented every time fault state machine switches to a LOCAL FAULT condition.
- Number of REMOTE FAULT events: (10-bit, rollover)
  - Incremented every time fault state machine switches to a LOCAL FAULT condition.
- Decoding Error Counter (32-bit rollover)
  - Increments each time a column contains a Code Error symbol.
  - In 100G/40GBase-R, a column is 8 octets.
  - In others, a column is 4 octets.
- Multi-Purpose Error status (32-bit rollover)
  - For 40 GbE and 100 GbE only, counts the number of times the BIP checking failed (one for each lane).
  - For 10GBase-R, counts the number of BER errors.
  - For 10GBase-X and 1000Base-X or SGMII, counts the number of disparity/code errors.

The following counters are not counting when the switch is in reset:

- Overflows
- Underflows
- Oversized Packets
- Runt Packets
- Undersized Packets

Packets discarded while the switch is in reset are not counted.

7.8.5 Energy Efficient Ethernet

The FM10000 supports Energy Efficient Ethernet (EEE) for 1 GbE and 10 GbE only. It does not support EEE for quad lane modes (40 GbE or 100 GbE) or for 2.5 GbE and 25 GbE.

The FM10000 EEE implementation follows clause 78 and updates in the Ethernet clauses for 10GBase-R and 1GBase-X for support of low-power mode.

The following registers control this feature:

- Interrupts:
  - LINK_IP[epl,lane].*LpIdleIndicate* — Indicates RX state change.
  - LINK_IP[epl,lane].*EeePcSilent* — Indicates TX idle state change.
  - LINK_IP[epl,lane].*LpiWakeError* — Reports lacks of periodic LI codes.
- Control:
  - PCS_1000BASEX_CFG[epl,lane].*EnEee* — Enables response to link partner low power idle request.
  - PCS_10GBASER_CFG[epl,lane].*EnEee* — Enables response to link partner low power idle request.
  - MAC_CFG[epl,lane].*TxPcActTimeout/Timescale* — Define idle time to interrupt.
  - MAC_CFG[epl,lane].*TxLpiAutomatic* — Automatic exit of low power mode.
  - MAC_CFG[epl,lane].*TxLpIdleRequest* — Controls TX low-power mode.
  - MAC_CFG[epl,lane].*TxLpiTimeout/Timescale* — Defines transmit time after out of low power mode.
  - PCS10GBASER_EEE_CFG[epl,lane] — Low level timers for LP state machine.
  - PCS1000BASEX_EEE_CFG[epl,lane] — Low level timers for LP state machine.
- Status:
  - PORT_STATUS[epl,lane].*LpIdle* — Port logic status, receiver in low power.
  - PORT_STATUS[epl,lane].*EeeSlPcSilent*— Port status, transmitter is idle.

EEE is disabled by default.

If the feature is enabled (*EnEee*=1), the receiver enters in low power mode upon receiving a sequence of /LI/ codes (low-power idle codes), and exists low power mode upon receiving non-/LI/ codes (refer to IEEE clauses for exact state machine definition for 1GBase-X and 10GBase-R, timers are configured in PCS10GBASER_EEE_CFG and PCS1000GBASEX_EEE_CFG). The hardware posts interrupts

(LpIdleIndicate) upon changes of states (entering or leaving LP mode) and software could read the actual state at any time (*LpIdle*). Also, the hardware receiver watches for periodic wake up /LI/ codes, which are used to keep the synchronization alive. The receiver posts an interrupt (*LpiWakeError*) if it fails to see the periodic wake up codes. It is up to software to decide if the interrupts are masked, and what further actions are needed from a system perspective.

For the transmitter, the FM10000 monitors transmit activity on each port, it starts a timer whenever the transmitter gets into an idle state. If there is no activity for a certain time (*TxPcActTimeout*), the hardware posts an interrupt (*EeePcSilent*). Software could use the interrupt to force the hardware into an idle mode. There is no automatic way for the hardware to enter into low-power mode by itself. The SerDes low-power mode must be asserted by software by setting *TxLpIdleRequest* to 1b.

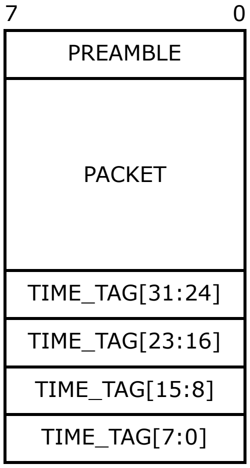
Software should also normally enable the TxLpiAutomatic mode. If a frame is received from the fabric to be transmitted on a port already in low-power, the hardware raises an interrupt (*EeePcSilent*), takes the SerDes out of low-power mode, waits a programmable interval (*TxLpiTimeout*), and starts frame transmission. Software could log the fact the port is no longer in low-power mode, perform other system tasks, but is not required to do any action on this port.

It is possible for software to not assert TxLpiAutomatic mode. In this case, if the transmitter is in low-power mode and a packet is received from the fabric, an interrupt is posted (*EeePcSilent*) indicating transmitter is not silent (*EeeSlPcSilent*=0) but the frame is maintained on hold until software disables the low-power mode (*TxLpIdleRequest* = 0), then hardware takes the SerDes out of low-power mode, waits a programmable interval (*TxLpiTimeout*), and starts frame transmission. This mode of operation is less common, as it imposes a longer waiting time for the frame in queue. This could be useful if software has to do extra operations at the system level and a fix time out period is not possible.

7.8.6 Time Stamping for IEEE1588

Each EPL has a 32-bit clock counter register synchronized to a master clock in the chip. This register is used for time stamping. All EPLs use the same clock source and are all phase locked to each other, the relative time is configurable in EPL_SYSTIME0. The time register is readable by software at any time.

On ingress, when enabled to do so, each MAC replaces the CRC with the time stamp for all packets received. The exact encoding is:



The time stamp is captured at the time the first byte of the preamble is received. The feature is optionally enabled using the MAC_CFG register.

On egress, the MAC replaces the last 4 bytes of the payload with a valid CRC (or a bad CRC if the frame was received with a bad CRC, or an uncorrectable memory error was detected for this packet). However, before doing so, if the MAC receives a command to capture the egress time for this packet (CAPTURE_TIME=1), the MAC captures the egress time of the first byte of the preamble of the packet and saves it in a temporary location. Then, at the end of the packet, the MAC extracts the last 4 bytes of the packet (which should contain the ingress time stamp), recovering the time captured when the packet ingressed. The two time stamps are then saved in two 32-bit registers, and an interrupt is raised. This interrupt allows a CPU to recover the precise time at which the packet entered and exited the switch.

There is only one set of time stamp registers per MAC, so only one capture command can be issued at a time, until software reads the captured time stamps and signals that it is ready for the next by clearing the ingress and egress time stamp registers (writing any value to MAC_1588_STATUS clears the registers).This procedure ensures that both time stamps are coherent and coming from the same packet.

7.8.7 Segment Rate Throttle

The MAC includes the capability to throttle its egress frame rate in such a way that a receiving FM10000 on the other end of the link uses a reduced scheduler allocation for that link without dropping frames. This can enable 50% of max frame rate for minsize frames, while still preserving 100% of bandwidth on longer frames.

The segment rate throttle configuration is specified in MAC_CFG, in the *TxSegSize*, *TxSegMinSpacing*, and *TxSegMaxCredit* fields.

- MAC_CFG.*TxSegMinSpacing* — Minimum average time per segment. Setting *TxSegMinSpacing*=0 disables the segment rate throttle.
- MAX_CFG.*TxSegMaxCredit* — Credit allowed for bursts above the max average rate.
- MAC_CFG.*TxSegSize* — Size of segments being throttled. This can be set to larger than a legal frame size, to throttle frame rate.

Each frame is divided into segments, the units of the receiver's schedule allocation. Segments are aligned to the start of each frame. All segments but the final segment of the frame have size *TxSegSize*, and the final segment has size less than or equal to *TxSegSize*.

The MAC maintains a counter, measured in units of byte-times (BT) of the current link rate. Credits increase at the link rate (regardless of whether data is currently flowing), and are decremented by *TxSegMinSpacing* on every start of segment. They saturate at *TxSegMaxCredit* (positive) and -32 KBT (negative). A frame may only be started when the credits are non-negative.

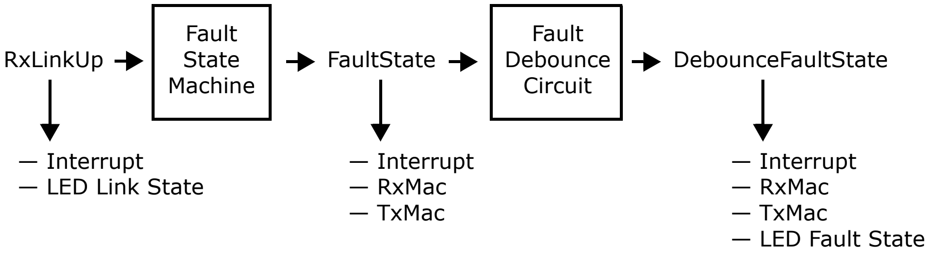
The segment rate throttle is not expected to be used in combination with DIC or clock compensation; if those features are enabled at the same time as the segment rate throttle, the behavior is unspecified. The segment rate throttle is not supported on links with rate less than 10 GbE

7.9 Status and Interrupts

- The MAC might produce the following interrupts/status:
- Signal present/absent on each lane.
- Symbol lock gain or lost on each lane.
- AN completion on each MAC.
- Link up/down on each MAC.
- Fault gain/lost on each MAC.

7.10 Link State and Fault Conditions

The EPL link state processing is as follows:



The *RxLinkUp* represents the status of the link as perceived by the PMA and PCS layer. The exact conditions included in this state are listed in Table 7-8, where some of these conditions are required while others can optionally be included or excluded by software.

Table 7-8 EPL Conditions (2)

Condition	Mode	Inclusion	Notes
SerDes Ready	All	Optional	
SerDes Signal Detect	All	Optional	Signal detection include a filter to smooth out possible glitches.
Symbol/Block Size	All	Required	
Lanes Alignment	40GBaseR 10GBase-X	Required	
Idle Detection	All	Optional	Detection of at least one idle per time period. This also includes LF and RF for 10G/40GBase-R
PHY Reports Link Up	SGMII	Required	
HiBer Detection	10G/40GBase-R	Optional	

The *RxLinkUp* state is then forwarded to the Fault State Machine which reacts in the following way:

- If the *RxLinkUp* is 0b (down), the Fault State Machine sets the FaultState to LOCAL FAULT.
- If the *RxLinkUp* is 1b (up), the Fault State Machine follows the IEEE defined fault state machine and set the FaultState to:

— LOCAL FAULT if receiving persistent local faults status from remote.

— REMOTE FAULT if receiving persistent remote faults status from remote.

— NO FAULT if no faults received from remote.

The *FaultState* condition then go through a debouncing circuit to clean up any temporary fault conditions and produce a *DebouncedFaultState*. The debouncing circuit defines two timeout (*UpTimeout* and *DownTimeout*) which are configured individually. The timer used depends on the transitions listed in Table 7-9:

Table 7-9 Debouncing Fault Timer

From	To	Timeout
Local Fault	Remote Fault	UpTimeout
Local Fault	No Fault	UpTimeout
Remote Fault	No Fault	Zero (instantaneous)
Remote Fault	Local Fault	DownTimeout
No Fault	Remote Fault	DownTimeout
No Fault	Local Fault	DownTimeout

Note: The usage of two timers allows the system to react at different time under fault or no-fault condition.

The *RxLinkUp*, *FaultState* and *DebouncedFaultState* are all available in the port status and can each be used as an interrupt source, though only the *DebouncedFaultState* should be really required from a software perspective.

The *FaultState* and *DebouncedFaultState* are also sent to the TxMAC and RxMAC which can be programmed to react automatically to the different fault conditions as per Table 7-10. For TxMAC in drain mode, packets from fabric are drained and not transmitted to the line, for TxMAC hold mode, packets from fabric are hold while the fault condition persist. The drain/hold mode can also be forced manually overriding the state conditions listed in Table 7-10.

Table 7-10 TxMAC Faults

FaultState	DebouncedFaultState	TxMAC	RxMAC
Local Fault	No	Send Remote Fault, hold/drain packets.	Drain any incoming frames.
No	Local Fault	Send Remote Fault, hold/drain packets.	Drain any incoming frames.
Remote Fault	Remote Fault	Send idles, hold/drain packets.	Drain any incoming frames.
Remote Fault	No Fault	Send idles, hold/drain packets.	Drain any incoming frames.
No Fault	Remote Fault	Not possible.	Not possible.
No Fault	No Fault	Send frames normally.	Accept incoming frames.

8.0 Tunneling Engine

8.1 Overlay Networks Overview

Figure 8-1 shows the different components of an Overlay Network.

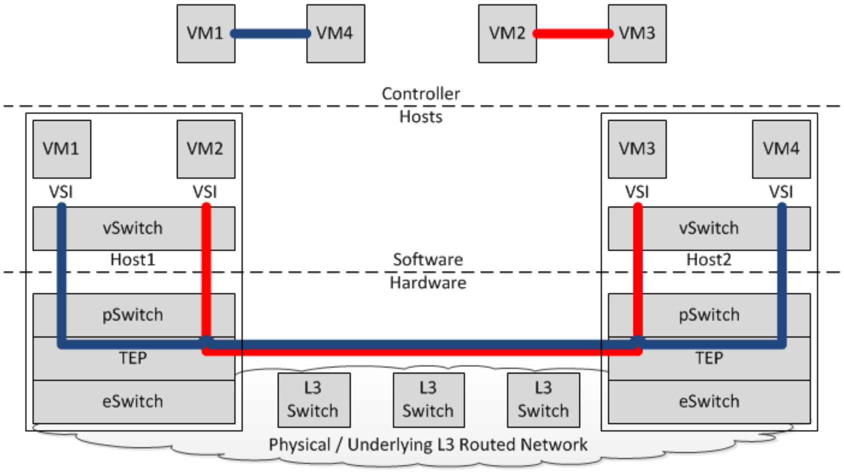


Figure 8-1 Overlay Networks Components

Terms:

- **Orchestration** — Software front end used to configure the system from a user perspective. This configuration is pushed into the controller to decide how to map this to real resources.
- **Controller** — Centralized software manager that has a view of the hosts in the system, and can place VMs into these hosts. Once placed, these VMs can be interconnected with virtual networks (VNs).
- **vSwitch** — Software switch running inside the host.
- **pSwitch** — Hardware switch controlled via vSwitch control plane, accelerating the vSwitch operation.
- **eSwitch** — Hardware switch controlled by the Switch Manager, and is the physical network switch.
- **VN** — Virtual network.
- **VNI** — Virtual network instance (24-bit in VXLAN and NVGRE, and 64-bit in GENEVE, like a VID for VLAN).
- **VSI** — Virtual switch interface (what a VM connects to).

- **VM** — Virtual machine.
- **TE** — Tunneling engine.
- **TEP** — Tunnel end point (typically one per vswitch/pswitch).
- **oDIP** — Outer DIP.
- **iDIP** — Inner DIP.
- **NAT** — Network Address Translation.
- **DeNAT** — Reverse NAT operation (for example mapping source IP addresses back to public addresses).

Orchestration layers (such as OpenStack) provide an abstract sandbox for users to create virtual machines and interconnect them arbitrarily using virtual networks. This causes certain problems/ opportunities within the network:

- Virtual machines in different virtual networks can be assigned any IP and MAC address the user wants. This means that the network can't assign these addresses by location (as IP subnetting would prefer to do).
- Virtual machines all within the same box may not be on the same virtual network and thus need to be isolated from each other. This mean that a domaining mechanism exactly like VLAN membership needs to be employed, but the flexibility of these virtual networks make it very hard to map this to actual VLANs.
- Orchestration layers are network agnostic- they don't care how the network is organized and prefer to leave that complexity to their IT department. Thus a solution that completely separates the physical network and all of its details from the virtual machines and their networks is preferred.
- Lastly, once virtual machines can live on separated virtual networks, the metaphor can be taken even further. Virtual routers, virtual load balancers, virtual firewalls, etc., anything that has physical wires in the data center can be virtualized on this platform.

To provide this separation, Overlay Networks insert tunnel endpoints (TEP) at the edge of the network, encapsulating the network header produced by the VM with an outer header with a set of addresses that have meaning to the physical (underlying) network. In addition, these tunnel headers also carry a virtual network identifier to provide the domain separation. The FM10000 supports the GRE, NVGRE, VXLAN and GENEVE tunnel formats. Since these tunnels exist at the edge of the network (primarily to abstract away the VM's headers from the physical network's addressing), when these tunnels are added and deleted in hardware it can greatly reduce or completely avoid the software overhead needed to implement this network layering.

The FM10000 must live in both the virtualized network as well as the physical network. For this reason, when doing tunnel endpoint functions the frame passes through the FM10000's processing path twice: once to implement the embedded switching portion that lives in the physical network (eSwitch), and once to implement the pseudo-virtual switching portion that lives in the virtual networks but is actually done in hardware (pSwitch). In between these two switching entities live the tunnel endpoints (TEPs) which add or delete tunnel headers to connect the two.

8.1.1 Host to Network

When a frame comes from for example VM1, it may be switched or modified in some way in the vSwitch, for functions that are not in hardware. Upon reaching the FM10000, the frame goes through the first pass of processing, which includes:

pSwitch Processing frames from the host:

- Determine the virtual switch interface (VSI) the frame came on, usually from the queue it originated on, or through a more complicated classification.
- Apply any network-centric functions on the frame (for example policing, counting, ACLs, mirroring, NAT, even routing).
- Determine that the frame needs to go through a tunnel endpoint, through some sort of N-tuple classification.

The pSwitch processing is split between one pass in the FM10000 switch pipeline and the tunnel engine pipeline. The tunnel engine has the capability to do an exact match classification (to save the switch pipeline for wild-card matches) as well as to do any address translation of the packet before it is pushed into a tunnel. After any address translation is complete, the tunnel engine then acts as the tunnel endpoint, (optionally) encapsulating the frame depending on the previous classifications.

TEP Processing frames from the host:

- As a result of the classification, the TEP determines which tunnel to push this frame into.
- A 5-tuple hash of the original frame's source/dest IP, source/dest L4 ports and IP protocol are used in certain encapsulation modes to expose the original frame's entropy to the physical network in a field in the outer header (for example the source L4 port in the outer UDP header in VXLAN).
- The tunnel determines the outer destination IP address to use, and may also determine the other tunnel attributes such as source address, IP TOS, TTL, etc.
- The virtual network instance (VNI) to put on the frame may be mapped from the VSI, or result from the same classification that sent this frame to this particular tunnel.
- The outer header's IP length is derived from the original frame's IP length. If the IP version of the outer does not match the inner, the difference in length calculations is accounted for.
- The outer header's IP checksum (if IPv4) is calculated on the fly and placed in the frame after the destination and source IP addresses have been chosen.

The frame then leaves the tunnel engine and passes through the switch's processing engine again, to apply the embedded switch functions.

eSwitch Processing frames from the host (after TEP):

- The eSwitch may route the outer header, choosing the NextHop DMAC according to the IP address that was encapsulated on the frame by the TEP.
- The eSwitch may also do other switch functions such as set the outer VLAN tag state, outer MAC forwarding, multi-chip tag forwarding, etc.

The eSwitch processing step may be extremely simple if the FM10000 is not doing any physical switching except for managing uplinks. If the FM10000 is acting as part of a distributed or L3 routed network, it does a complete switch processing step to send the packet to the right port and/or NextHop.

8.1.2 Network to Host

When a frame is received by the FM10000 connected to Host1, it goes through the first pass of switch processing.

eSwitch Processing frames from the network:

- Apply standard embedded switch functions, which may include forwarding the packet back out another Ethernet port (port to port switching).
- If the frame is tunneled, apply any security checks to make sure that this tunnel is coming from a known source.
- If this frame is tunneled and addressed to a local TEP, pass it to the tunnel engine for potential decapsulation.

In this decapsulation case the frame is then sent to the tunnel engine, where it is classified on its inner header.

TEP Processing frames from the network:

- Looking at the inner header, optionally translate changes to the DSCP or ECN bits in the outer header IP TOS fields to the inner IP TOS.
- Looking at the inner header, determine if this frame should have the outer header appended, deleted, or left alone.

Independent of whether the frame was completely decapsulated, the pSwitch analyzes the inner header using exact match classifications.

pSwitch Processing frames from the network:

- Looking at the inner header, determine if this frame needs address translation. If identified, translate these addresses.
- If the frame was decapsulated or appended, the frame may also be classified and processed using the second pass of the switch packet processor.

As a result of the pSwitch classification, the frame is pointed to the corresponding port or PCIe queue that maps to the frame's VSI.

Each TEP has a single physical network IP address:

- On reception from the network, oDIP -> my local TEP.
- On reception from the hosts, VSI is mapped to one TEP, or we have to derive TEP from the frame.
- On transmission, the FM10000 can set SIP per TEP, which can be mapped again from VSI or derived from the frame.

8.1.3 Virtual Network Functions

The FM10000 provides full switch/router capabilities and provides them to the host operating system to do certain network-centric accelerations (classify a packet, apply an ACL, count against a policer, etc.), however in order for these features to be useful in a network that is employing Overlay Networking it must also be fully tunnel aware. The FM10000 has the capability to apply network-centric processing on the inner and outer header of an encapsulated frame, and after processing optionally remove the outer header and pass it directly to the virtual machine. Additionally, the FM10000 has the ability to apply these network-centric accelerations and as a result modify the packet addresses before pushing the entire packet header into an encapsulating tunnel.

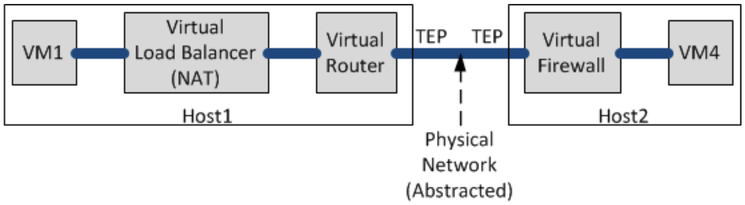


Figure 8-2 Overlay Networks Functions

The above picture shows a few example network functions that could be inserted in between VM1 and VM4. Since these functions are being done on the virtual plane, they are actually running inside the host that is hosting the VM (Host1 for VM1, Host2 for VM4). These functions are commonly implemented fully in software. With the FM10000, certain dataplane modifications can be done by the hardware, allowing the packet to flow through the hardware without having to be passed in between software services within the host.

In this example, a virtual load balancer is running on Host1, as well as a virtual router. The load balancer has made some forwarding decision for certain flows coming out of VM1, and has programmed the FM10000 to match on these flows, and translate their addresses to point to the destination VM. The virtual router is also programmed to match on IP addresses and do a L3 route. All of this modification happens on the packet coming from the VM, on top of the tunneling that is done by the TEP to carry the frame across the network.

The rules for the NAT and virtual router are flattened inside the hardware tables. Therefore, that the same classification can trigger both the NAT action (update L3/L4) as well as the corresponding L2 routing updates. This causes the NAT and route action to be performed as a single action by the tunnel engine, which updates the L2/L3/L4 fields in place. Additionally, the tunnel engine may also associate a tunnel with the same flow, encapsulating the newly updated frame with a tunnel header. The tunnel header is then processed by the embedded switch to forward it through the physical network.

Inside Host2 there is a software service running that is implementing a virtual firewall. It has programmed some set of hardware rules (ACLs) that allow it to inspect the packet and drop/log/trap/monitor the traffic as it is received of specific tunnel endpoints. If it is able to program these filters in the hardware, the frame does not need to forwarded to the virtual firewall's own software dataplane unless it is an exception packet (unknown flow, error packet, security exception, etc.).

When the frame is received by Host2, it is logically decapsulated by the TEP and its inner header fields are inspected. Here the inspection may happen using the TEP classifier (during the decap step), or if the outer header is removed inspected by the pSwitch's FFU operation. After these filter rules are applied, the inner header of the packet is inspected to switch the packet to VM4.

The FM10000 supports the capability to apply its network-centric functions to these frames sent to/from VMs as well as well as translate the addresses of the frame. The important functions used in this way are:

Inside the switch pipeline:

- **FFU Classification** — The ability to arbitrarily classify the frame, to implement ACL checks
- **Basic L2 Modification** — IP route, VLAN tag modify, and priority updates.
- **QoS** — Apply policers and switch scheduling/shaping to the frame.

Inside the Tunnel Engine:

- **N-tuple Classification (Exact Match)** — Similar to FFU matching, except that these rules are all exact match and can be done in more efficient hash table operations. This is most commonly used to classify frames in and out of tunnels, as well as classify for load balancer actions.
- **L2 Modification** — The ability to update the DMAC, SMAC and VLAN of a frame (to implement a virtual router).
- **L3 Modification** — The ability to update the L3 addresses (both Dest and Source, IPv4 and IPv6) to translate addresses between virtual machines (NAT). Additionally it is required to have the ability to control the TTL of a frame to detect hop counts in a virtual network. Lastly control of the IP TOS is used to classify a frame for priority, as well as translate color and ECN information that may have been affected by the transit through the physical network.
- **L4 Modification** — The ability to update the L4 addresses (both Dest and Source, for TCP and UDP) to translate ports between virtual machines (PAT).

These types of controls are similar to OpenFlow, and can be programmed in this way (using a flow-based match-action semantic) or the system can be programmed in a 'macro-flow' mechanism more similar to how a network switch is programmed today (using dedicated function tables for forwarding, learning, security, ACLs, etc. with wild-card matching).

8.2 Tunneling Engine Architecture Overview

The Tunneling Engine (TE) is a packet modifier engine capable of performing the following functions:

- Add/remove tunnels for VXLAN and NVGRE
- Perform network address translation (NAT)
- Keep statistics per flow or tunnel

NAT services are available even if encapsulation or decapsulation is not used.

Figure 8-3 provides an overview of the block.

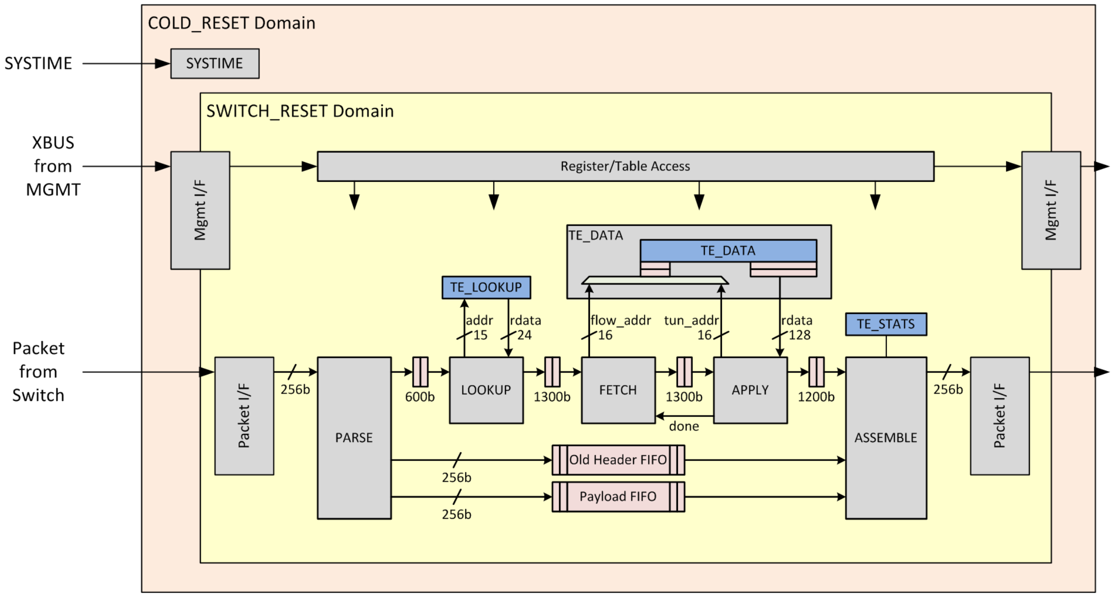


Figure 8-3 Tunneling Engine Block Diagram

The unit has three interfaces:

- Management interface (in/out) to configure and monitor the unit.
- Packet interface (in/out) to process packets.
- System time interface to implement an optional time reference.

There are two resets to the unit:

- Cold reset — Asserted to reset entire device, normally once at power up.
  - Reset entire block
- Switch reset — Asserted when the switch is reset.
  - Reset most of logic except system time and management interface. If switch reset is asserted, management writes are ignored and management reads return zeros, but the management still completes the transactions properly.

The unit supports up to 56 K tunnels (simplest form) and up to 256 VSIs (0..255).

The performance of the unit is:

- Up to 100 Gb/s, max throughput achieved for large packets.
- Up to 50 Mp/s for short packet for simple encapsulation/decapsulation.
  - Performance is lower for more complex modifications.

TE can process data at up to 100 Gb/s from the Ethernet switch using a 256-bit data path clocked at 500 MHz, and contains the following units:

- **Parser** — Extracts the interesting fields for packet.
- **Lookup** — Recovers pointer and length of a first list of flows.
- **Fetch** — Issues initial read to recover keys, flows and tunnels data structures.
- **Apply** — Compares keys and applies transforms.
- **Assemble** — Creates new and/or updated existing packet headers.

The parser parses the packet to extract useful fields, and pushes the packet into the "old-header" or "payload" FIFO. The parser can be configured in different ways to recover checksum and packet length, and thus might operate in cut-through mode or store-and-forward mode on a packet-by-packet basis. As an example, if the unit is configured to recover the length and checksum of the packet from headers (IP header for length, TCP/UDP header for checksum) and those headers are present on a given packet, the unit can operate in cut-through mode for this packet, as it doesn't have to wait for the end of packet.

If the packet is not an IP packet, or the packet is not UDP/TCP and a checksum is requested, the unit has no choice other than computing the checksum and length by walking through the entire packet before starting the process. The unit then operates in store-and-forward for this packet.

The lookup stage recovers an initial pointer to a data structures block using the TE_LOOKUP table, and the fetch stage recovers the actual data from the TE_DATA table. The apply stage applies the data transform requested on the first matching key. The assemble stage reconstructs the header and keeps statistics.

The memory block used during the transformation are:

- 32 K x 24-bits for initial pointer/size of flow/tunnel structure locations.
- 56 K x 128-bits holding flow/tunnel data.
- 4 K x 104-bits for statistics.

If an uncorrectable error is encountered in those memory blocks, the packet is dropped. All SRAM errors (correctable or not) are reported to the FM10000 centralized interrupt controller. There are no other interrupt sources than SRAM errors in this unit. If an uncorrectable error occurs on another small memory block, the tunnel engine stops processing packets until it is reset.

The management interface allows read/write into tables and registers while traffic is flowing, table changes must be done in a coherent order for the hardware to work properly.

8.3 Packet Disposition

TE supports the following packet processing:

- Normal packet processing (see next sections)
- Drop packet
- Trap packet

The TE drops packets under the following conditions:

- An uncorrectable error has been encountered.
- The data pointers point outside the memory (this flags a software programming error).

The dropped packets are simply counted in TE_DROP_COUNT. Packets received by the TE with bad frame indication are counted in TE_ERR_FRAMES_IN.

The TE traps a packet under the following conditions:

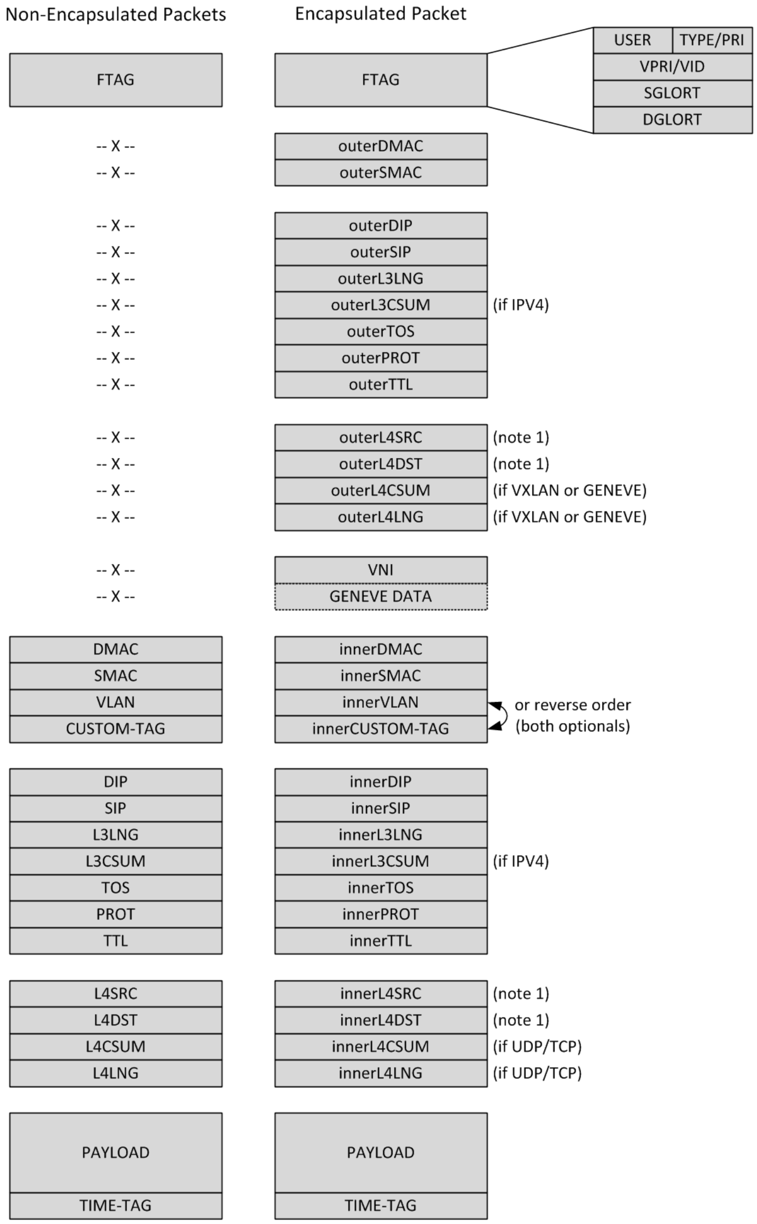
- Packet not matching any known flow.
- A NAT operation is requested on a packet that is neither TCP or UDP.
- Unknown DGLORT.

A trapped packet is forwarded unmodified except for the following changes:

- Time tag updated if requested by switch.
- dglort set to TRAP_DGLORT+N if enabled via TE_TRAP_CONFIG, else set to TE_DEFAULT_DGLORT if DGLORT_DEC.*setDglort* or no DGLORT_MAP match, else dglort on incoming frame.
- sglort set to TE_DEFAULT_SGLORT if DGLORT_DEC.*setSglort* or no DGLORT_MAP match, else sglort on incoming frame.
- Frame invalidated if requested via TE_TRAP_CONFIG.

8.4 Packet Format

The packet format supported is similar to the one supported by the PCI Express End Points. Fields of interest to TE are listed in Figure 8-4.



Note: For UDP and TCP, the L4SRC and L4DST are set according to the standard UDP and TCP headers format respectively. For any other protocols, the L4SRC and L4DST are set to 0b.

Figure 8-4 TE Packet Format

Figure 8-4 does not show the IP headers, TCP and UDP headers in all details, only the fields that are of particular interest for the TE operation. It is expected the reader is familiar with the full header format. The TE maintains compatibility with existing IP standards as new headers are created or modified.

The minimum packet size supported by TE is 27 bytes (FTAG(8) + DMAC/SMAC/ETYPE(14) + PAYLOAD(1) + CRC/TIMETAG(4) = 27). Any packet smaller than this is filtered out silently.

TE supports presence of one or two tags between the SMAC and the inner IP header. One of them is a standard VLAN tag, fixed size (4 bytes) and of type 0x8100. The second tag type and size is programmable (TE_EXVET) and could be defined to come before or after the VLAN tag.

TE skips over any IPv4 options, and the following types of IPv6 extension headers:

- IPv6 Hop-by-Hop Option (protocol number 0x00, length 8N+8 bytes).
- Routing Header for IPv6 (protocol number 0x2b, length 8N+8 bytes).
- Fragment Header for IPv6 (protocol number 0x2c, length 8 bytes; N is always 0).
- Destination Options for IPv6 (protocol number 0x3c, length 8N+8 bytes).
- Authentication Header (protocol number 0x33, length 4N+8 bytes – not the same as the other IPv6 extension headers).

Extension headers may appear in any order. The parser stops after the current header if the protocol number of the next header is not recognized. If an IPv6 extension header extends beyond 512 bytes from the start of packet (including FTAG), the packet is trapped.

Note: TE does not use FTAG.VID in hashing. When an 0x8100 tag is present (as configured via TE_EXVET) TE uses the VID from the L2 tag. If no tag is present, VID is 0.

Note: For UDP and TCP, the L4SRC and L4DST are set according to the standard UDP and TCP headers format, respectively. For any other protocols, the L4SRC and L4DST are set to 0.

TE detects presence of a VXLAN header by comparing UDP destination ports to TE_PORTS.*VXLAN*. No matching means that the packets are not VXLAN encapsulated.

The individual tunnel headers are as follows:

NVGRE:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-----------------+-----+-------------------------------+
|C|R|K|S|    Reserved     | Ver |     Protocol Type 0x6558      |
+-+-+-+-+-----------------+-----+---------+---------------------+
|          Virtual Subnet ID (VSID)        |        FlowID       |
+------------------------------------------+---------------------+
```

Where:

C = Checksum Present = 0

R = Reserved = 0

K = Key Present (set to indicate presence of VSID/FlowID word)

S = Sequence Number Present = 0

Reserved0 = Reserved = 0

Ver = Version = 0

Protocol Type = 0x6558 [Transparent Ethernet Bridging]

VSID = Virtual Subnet ID, Layer2 Network

FlowID = May provide entropy for flows with same VSID (0 if unused)

The FM10000 sets C,R,S,VER to zero and K to 1 during encapsulation, and can only perform decapsulation if {C,R,S,Ver} = 0.

If K = 0 during decapsulation, the VSID is treated as 0.

The FM10000 ignores FlowID during decapsulation, and sets FlowID to zero during encapsulation.

VXLAN:

<table>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>R</td><td>R</td><td>R</td><td>R</td><td>I</td><td>0</td><td>R</td><td>R</td><td colspan="24">Reserved</td></tr>
<tr><td colspan="23">VXLAN Network Identifier (VNI)</td><td colspan="9">Reserved</td></tr>
</table>

Where:

I = Indicates presence of VNI, required, must be 1 when encap, checked on decap.

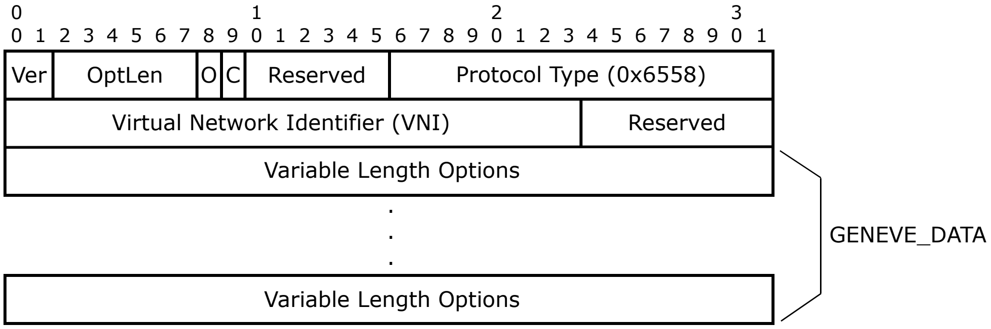
R = Reserved. Set to 0 when encap, ignored on decap.

Bit 5 is checked to be 0 (if GPE enabled)

8.5 GENEVE Packet Format

The FM10000 supports GENEVE (NGE) encapsulation. TE detects presence of a GENEVE header by comparing UDP destination ports to TE_PORTS.*NGE*. No matching means that the packets is not GENEVE encapsulated. The GENEVE header is shown next and inserted between the outer UDP header and the actual encapsulated frame.

GENEVE:



Where:

Ver = Set to 0 on encap. Checked to be 0 on decap.

O (OAM) = Set to 0 on encap. Checked to be 0 on decap.

C = Set to 0 when encap. Checked to be 0 on decap.

OptLen (6 bits) = The length of the options fields, expressed in four byte multiples, not including the eight byte fixed tunnel header. This results in a minimum total NGE header size of 8 bytes and a maximum of 8+248 bytes. However, the FM10000 only supports up to 16 DWORD (64 bytes) of options (OptLen=16). The options a referred to NGE_DATA.

Note: The TE_TUN_HEADER_CFG provides an option to either trap or pass through that are found to have either OAM or C bit set, option configurable for each case.

8.6 VXLAN-GPE/NSH Packet Format Support

FM10000 supports VXLAN-GPE/NSH Packet Format. The support of VXLAN-GPE/NSH is enabled using bit 2 of TE_TUN_HEADER_CFG.*EncapVersion*.

- If set to 0b, the TE supports NVGRE, GENEVE and VXLAN.
- If set to 1b, VXLAN-GPE/NSH and normal VXLAN support are enabled, while NVGRE and GENEVE are disabled.

A given TE cannot support all encapsulation protocols. However it is possible to configure one TE to support NVGRE, GENEVE and VXLAN, and a second to support VXLAN and VXLAN-GPE/NSH.

There are two parts to VXLAN-GPE/NSH: the VXLAN-GPE header itself and the NSH header. The VXLAN-GPE is a variant to the VXLAN encapsulation method:

VXLAN-GPE:

<table>
<tr><td>0</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>2</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td></tr>
<tr><td>R</td><td>R</td><td>V</td><td>V</td><td>I</td><td>P</td><td>R</td><td>O</td><td colspan="16">Reserved</td><td colspan="8">Next Protocol</td></tr>
<tr><td colspan="24">VXLAN Network Identifier (VNI)</td><td colspan="8">Reserved</td></tr>
</table>

Where:

I = Indicates presence of VNI. Must be 1b. Set to 1b on encapsulation. Checked to be 1b on decapsulation (trap to DECAP_BAD_TUN if found 0).

P = Indicates this is VXLAN-GPE. Set to 0b on encapsulation for normal VXLAN, and set to 1b for VXLAN-GPE. Checked to detect if VXLAN or VXLAN-GPE (if and only if GPE mode enabled).

O = Indicates OAM. Set to 0b on encapsulation for normal VXLAN and VXLAN-GPE. Checked to be 0b on reception (if and only if GPE mode is enabled and P=1b and *CheckOAM* is on).

VV = Indicates version. Set to 0 on encapsulation. Checked to be 0 on reception (if and only if GPE mode is enabled and P=1b and *CheckVersion* is on).

The Next Protocol values defined for the GPE header are:

Next Protocol = 1 — IPv4 (not supported)

Next Protocol = 2 — IPv6 (not supported)

Next Protocol = 3 — Ethernet (L2 header, supported)

Next Protocol = 4 — NSH header (supported)

TE detects presence of an VXLAN or VXLAN/GPE header by comparing UDP destination ports for either TE_PORTS.*NGE* or TE_PORTS.*VXLAN* ports. The following TE_PORTS configurations are supported:

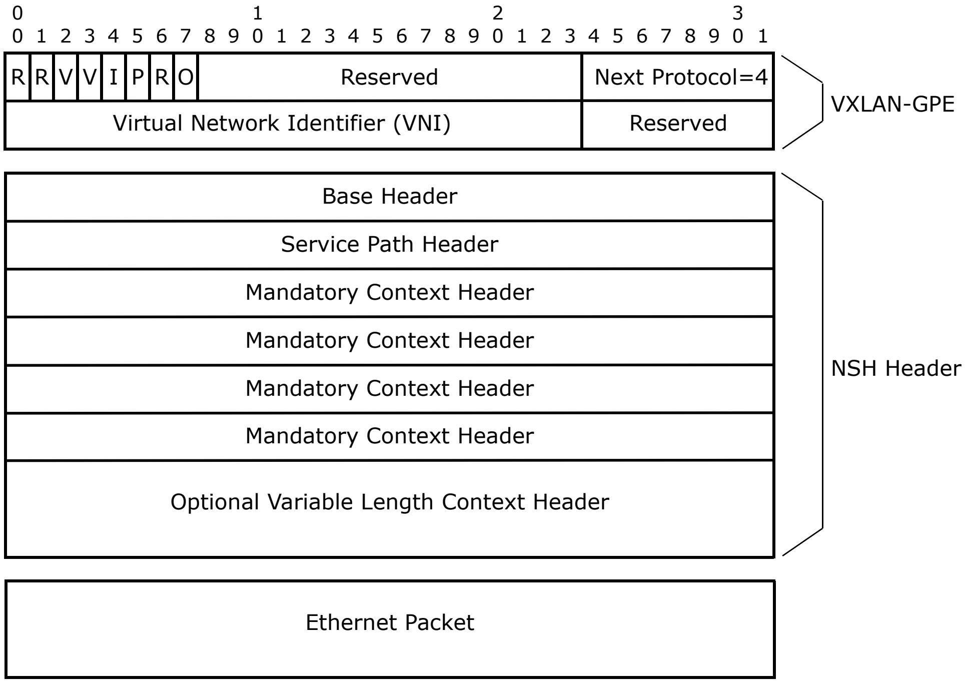
- To enable all VXLAN* packet types (VXLAN/+GPE/+GPE+NSH) on any of 2 ports, set two different non-zero values in TE_PORTS. Both ports are accepted equally (port 4789 normally used for VXLAN and 4790 for VXLAN-GPE).
- To enable only NSH packets on one port, set TE_PORTS.*VXLAN*=0, and TE_PORTS.*NGE* is the port number for NSH packets only.
- To enable only VXLAN and VXLAN+GPE without NSH on one port, set TE_PORTS.*NGE*=0, and TE_PORTS.*VXLAN* is the port number for non-NSH packets only.

When GPE mode is enabled, TE distinguishes the type of encapsulation using the P bit:

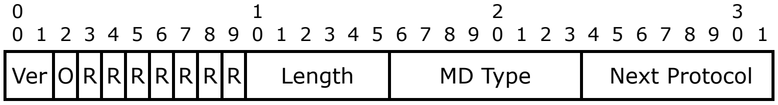
- If P = 0b, the packet is VXLAN and next header is Ethernet.
- If P = 1b and Next Protocol = 3, the next header is Ethernet, thus processing identical to normal VXLAN.
    - — The actual Next Protocol value (normally 3) must be programmed in TE_TUN_HEADER_CFG.*EncapProtocol*[7:0] for GPE/NSH to work properly.
- If P = 1b and Next Protocol = 4, the next header is NSH header, and the parser examines the first word of the NSH header to determine length of the NSH header.
    - — The actual Next Protocol value (normally 4) must be is programmed in TE_TUN_HEADER_CFG.*EncapProtocol*[15:8] for GPE/NSH to work properly.

The next group shows the VXLAN-GPE header and NSH header together:

VXLAN-GPE:



The Base Header structure is the following:



Where:

Ver = Header version. Only 0 is supported. Any other packet is trapped

O = OAM message. Trapped.

Length = Length in 4-byte QTY. Includes optional headers. Minimum is 6.

MD Type = Metadata type is 1 (not checked)

The Next Protocol values defined for the NSH Base Header are:

Next Protocol = 1 — IPv4 (not supported)

Next Protocol = 2 — IPv6 (not supported)

Next Protocol = 3 — Ethernet (L2 header)

- The actual Next Protocol value (normally 3) must be programmed in TE_TUN_HEADER_CFG.*EncapProtocol*[7:0] for GPE/NSH to work properly.

The Tunneling Engine parser (during decap) uses the *Length* field to determine the length of the NSH header, and also checks the Next Protocol. Only Ethernet is supported, so all other protocols cause the packet to be trapped.

In summary, if VXLAN-GPE is enabled, the TE parser checks the following during decapsulation:

- If P = 1b, check if O=0 and VV=0. Trap otherwise (trap DECAP_BAD_TUN).
- If P = 1b, check if Next Protocol in the VXLAN-GPE header is 3 or 4. Trap otherwise (trap DECAP_BAD_TUN).
  - If Next Protocol (in the GPE header) is 3, process as normal VXLAN header.
  - If Next Protocol (in the GPE header) is 4, check NSH Base Header for Ver=0, O=0, NextProtocol=3. Trap otherwise (trap DECAP_BAD_TUN). The MD type field is ignored.
  - If Ver=0, O=0, NextProtocol=3, use *Length* to determine location of L2 header

Notes:

- The checking of O can be enabled or disabled using the TE_TUN_HEADER_CFG.*CheckOam* bit.
- The checking of VV can be enabled or disabled using the TE_TUN_HEADER_CFG.*CheckVersion* bit.

For encapsulation, the entire VXLAN-GPE and NSH headers must be supplied using NGE data structure field using TunnelType=GENERIC(0). The first word of NGE_DATA is the first word of the VXLAN-GPE, the next word is second word of VXLAN-GPE, the third word is first word of NSH header, etc. The first words must include all the bits properly set (VV, I, P, O, Reserved). Contrary to GENEVE, TE does not compute the Length in the NSH base header, it must be explicitly called out as part of the NGE construct.

NSH headers of the maximum possible length (63 * 4 bytes) can be parsed and decapsulated. However, there is a limitation of 14 * 4 bytes on encap due to all tunnel words being supplied via NGE data. The total amount of NGE data is limited to 16 words (see NGE Encapsulation section below), of which the first 2 words are used as VXLAN header as described above.

8.7 Registers

The TE supports the following registers:

- TE_DATA[0..57343] (128 bits) — Contains flow/tunnel data.
- TE_LOOKUP[0..32767] (24 bits) — Contains flow pointers.
- TE_STATS[0..4095] (128 bits) — Contains stats (only 104 bits used).
- TE_USED[0..1023] (64 bits) — Track flow usage
- TE_DGLORT_DEC[0..7] (64 bits) — Defines how to recover flow-id, tep-id and the direction (encap/NAT or decap/DeNat).
- TE_SGLORT_DEC[0..7] (64 bits) — Defines how to derive VSI.
- TE_SYSTIME (40 bits) — Current system time.
- TE_SYSTIME0 (40 bits) — Defines default system time when a reset is received on SYSTIME interface.
- TE_SYSTIME_CFG (4 bits) — Defines system time increment for each tick on the SYSTIME interface.
- TE_DGLORT (32 bits) — Defines default DGLORT for each direction.
- TE_SGLORT (32 bits) — Defines default SGLORT for each direction.
- TE_SIP[0..255] (128 bits) — Default SIP during encapsulation per VSI.
- TE_VNI[0..255] (24 bits) — Default VNI during encapsulation per VSI.
- TE_DEFAULT_L4DST (32 bits) — Defines default destination port for VXLAN.
- TE_CFG (32 bits) — Defines global directives.
- TE_DROP_COUNT (32 bits) — Count dropped packets (counter only 16 bits).
- TE_TRAP_DGLORT (32 bits) — Defines DGLORT to use for trapped packets.
- TE_PORTS (32 bits) — Defines VXLAN and GENEVE ports.
- TE_EXVET (32 bits) — Defines custom Ethernet tag, size and location.
- TE_TUN_HEADER_CFG (32 bits) — Defines tunnel header format and version.
- TE_FRAMES_IN (32 bits) — Counts number of packets in.
- TE_FRAMES_DONE (32 bits) — Counts number of packets done (includes dropped packets).

GENEVE (NGE) also includes the following extra registers:

- TE_DEFAULT_NGE_DATA[0..15] (32 bits) — Defines default GENEVE data.
- TE_DEFAULT_NGE_MASK (16 bits) — Defines default GENEVE data present.

All registers above are 64-bit accesses regardless of their defined width. Software must maintain coherency by always modifying the set of flow/data structures in a scratch area and then updating the TE_LOOKUP and TE_DATA pointers to use the newly updated structures. Similarly, to recover unused space, the hardware first updates the pointers to get around the unused space before using this space again.

Note: The FM10000 supports two tunneling engines, registers listed above are per tunneling engine. All actual register declarations include the TE index, so TE_CFG becomes TE_CFG[0..1] and TE_SIP[0..255] becomes TE_SIP[0..1,0..255].

8.8 DGLORT/SGLORT Decoding

The switch provides the initial directives to the TE through the {DGLORT,USER} set. The TE_DGLORT_MAP and DGLORT_DEC registers are used to defined how the directives are recovered.

The registers TE_DGLORT_MAP[0..7] registers define value/mask pairs against which received {DGLORT,USER} are compared. The index of the last matching entry is then used to recover the directives.

```
dglortIdx = -1;
 foreach i (0..7)
     dv = TE_DGLORT_MAP[i].DGLORT_Value;
     dm = TE_DGLORT_MAP[i].DGLORT_Mask;
     uv = TE_DGLORT_MAP[i].UserValue;
     um = TE_DGLORT_MAP[i].UserMask;
     if ( (FTAG.DGLORT & dm) == dv && (FTAG.USRER & um) == uv )
     {
         dglortIdx = i;
     }
```

The TE_GLORT_DEC[dglortIdx] then defines if the request is encapsulation, decapsulation, or only NAT, and how to derive the flow information needed for the transformation. If there are no matching entries, the packet is passed transparently and not modified except for the following:

- DGLORT is set to TE_DEFAULT_DGLORT first, then possibly redirected if trap NO_DGLORT_MATCH asserted and program to redirect.
- SGLORT is set to TE_DEFAULT_SGLORT.
- Time tagging (refer to Section 8.13).

Similarly, the switch provides the source VM for the packet through the SGLORT, used for encapsulation/NAT only. The TE_SGLORT_MAP[0..7] registers define value/mask pairs against which the SGLORT is compared. The last matching entry defines how to extract the VSI from the SGLORT. If there are no matches, the packet disposition follows the TE_TRAP_CONFIG.*NO_SGLORT_MATCH* configuration.

```
// For encapsulation or NAT
 sglortIdx = -1;
 foreach i (0..7)
     value = TE_SGLORT_MAP[i].Value;
     mask = TE_SGLORT_MAP[i].Mask;
     if ( (FTAG.SGLORT & mask) == value )
     {
         sglortIdx = i;
     }
 if ( sglortIdx >= 0 )
 {
     vs = TE_SGLORT_DEC[sglortIdx].VSI_Start; // Max 15
     vl = TE_SGLORT_DEC[sglortIdx].VSI_Length; // Max 8
     vo = TE_SGLORT_DEC[sglortIdx].VSI_Offset; // Max 255
     vsi = ((SGLORT >> vs) & ((1<<vl)-1)) + vo; // Max 255
 }
```

If either DGLORT or SGLORT are no match, parsing stops, and Lookup and Apply stages are canceled. Also, if both DGLORT and SGLORT are no match, the TE_TRAP_CONFIG.*NO_SGLORT_MATCH* disposition takes precedence over the TE_TRAP_CONFIG.*NO_DGLORT_MATCH* disposition.

8.9 Flow Data, Tunnel Data, and Counters

TE stores initial pointers in the TE_LOOKUP table, flows and tunnels data in the TE_DATA table and counters in the TE_STATS table. TE maintains a pointer for flow, tunnel and counters along with a validity bit for each pointer:

- ***flowPtr*** — Location of flow data structure
- ***tunnelPtr*** — Location of tunnel data structure
  - — Optional. If not valid, no encapsulation is performed
- ***counterPtr*** — Location of counter structure
  - — Optional. If not defined, defaults to 0b

The TE_STATS table contains a set of frame and byte counters indexed by <*counterPtr*>. The frame count is the number of frames forwarded back to the switch (and thus not counting the dropped frames), and the byte counter is the number of bytes forwarded to the switch after modifications. The unit always counts all frames forwarded, and uses the default counter pointer of 0b if none specified. If both a flow counter and a tunnel counter are specified, the flow counter takes precedence.

The TE_DGLORT_DEC registers define how to compute the initial *flowPtr* and length of this structure.

There are 2 methods supported:

- From {DGLORT,USER} itself
  - — The matching TE_DGLORT_DEC[m] defines a table pointer (*Base*) and the location of the index (*IndexStart*,*IndexLength*) in the {DGLORT,USER} set. The TE then performs a lookup in the TE_LOOKUP at location Base+Index to recover a pointer to a data block and its length (last flag set to 1b in this case). A second lookup is then done in the TE_DATA to recover the data block itself.
- From keys within packet {id,DMAC,SMAC,VLAN,DIP,SIP,L4SRC,L4DST,PROT}
  - — The TE_DGLORT_DEC[m].*Base* defines the *Base* and *HashSize* of a hash table and which keys are used in the search. The TE computes a 16-bit hash value from the selected keys, and calculates the hash table index as:

    ```
    Index = (hash & ((2 << HashSize)-1))
    ```

    It performs a lookup in the TE_LOOKUP table at location Base+Index, to recover a pointer, length, and last-flag for a data block which contains a packed list of flow keys, flow data and possibly tunnel structures. The keys are compared to the keys extracted from the packed list (exact match only) and the matching flow is then used for flow and tunnel processing.

    - If the data block was not the last, the hardware presumes the first 32 bits contains a pointer to the next block along with a last flag and a length. This is in the same format as TE_LOOKUP, padded to 32 bits.
  - — For encap/NAT, the possible keys are {VSI,DMAC,SMAC,VLAN,DIP,SIP,L4SRC,L4DST,PROT}, the field TE_DGLORT_DEC[m].*HashKeyConfig* defines which keys are selected during the search. The VSI is recovered from SGLORT.
  - — For decap/DeNAT, the possible keys are {TEP-ID,VNI,inner(DMAC,SMAC,VLAN,DIP,SIP,L4SRC,L4DST,PROT)}, the field TE_DGLORT_DEC[m].*HashKeyConfig* defines which keys are selected. The TEP-ID is extracted from the {DGLORT,USER} using the TE_DGLORT_DEC.{*IndexStart*,*IndexLength*}, and the VNI is extracted from the packet.

The data structures are shown in Figure 8-5.

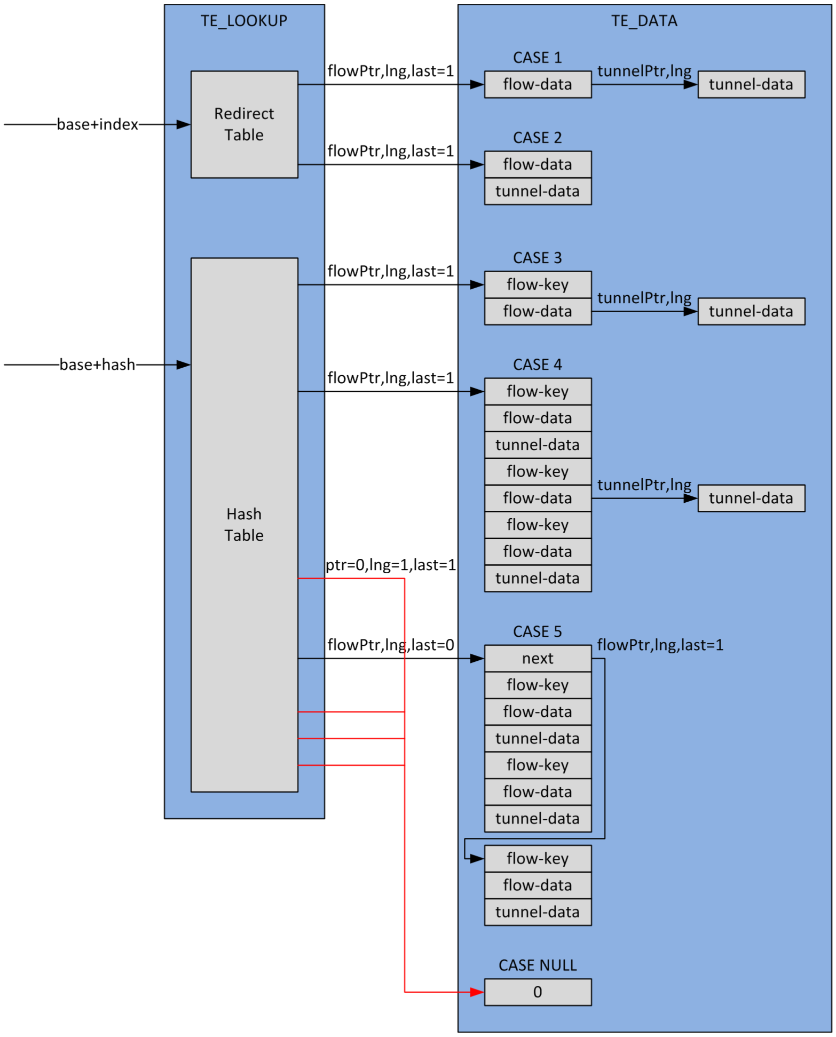


Figure 8-5 Encapsulation Flow Data Locations

The following cases are shown:

Case 1: The {DGLORT,USER} is used to recover a Base and an Index, the TE_LOOKUP[Base+Index] is then read to recover a pointer to a data block and its length. The data block contains a flow structure which contains a pointer to a tunnel data structure and its length. A second lookup for recovering tunnel data is done at a cost of an extra 8ns stall in the pipeline. The indirection allows sharing tunnel data for multiple flows.

Case 2: As for case 1, the {DGLORT,USER} is used to recover a Base and an Index, the TE_LOOKUP is then read to recover a pointer to a data block and its length. In this case, the data block contains a flow data structure and a tunnel data structure appended right after. This structure is faster and could be used when there is a specific tunnel for a given flow or if the flow data is empty.

Case 3: The {DGLORT,USER} is used to recover a Base pointer and HashSize, and defines which keys to use in the search. A 16-bit flow hash is then computed from the selected keys (masked as per TE_DGLORT_DEC.*HashKeyConfig* setting), and the LSB bits of this hash are used as the hash table index. Then, TE_LOOKUP[Base+Index] is read to recover a pointer of a data block and the length of this block. The hardware reads the memory in a burst of 128 bits of data, and compares to keys in the data structure to the key from packet. If it matches, the flow data following the keys is used. In this example, there is a second lookup to recover the tunnel data, but it is also possible to do as Case 2 and append the tunnel data to the flow structure. If there is no match, the packet is trapped.

Case 4: This is similar to case 3, except that the pointer and length recovered from TE_LOOKUP points to packed data block of *flow-keys*,*flow-data* structure which might contain tunnel data as well. As in case 1 and 2, tunnel data might be append to the flow data structure or might be recovered from a pointer in the flow data.

Case 5: This shows multiple data blocks chained together. Software is free to chop entire set of data structures into different blocks, a given data structure could be split across blocks.

Case "NULL" The recommended method when there are no flows for a given table entry in TE_LOOKUP is to reserve an entry for this purpose (probably entry 0) and set the content to 0b.

Pointers are to blocks of consecutive 128-bit data words in TE_DATA. The length of the block (in 128-bit words) is specified along with the pointer. Each block consists of one or more structures packed together, where each field is always aligned to a 16-bit boundary. The first 16 bits of each structure defines what fields are present in the structure. If the structures do not exactly fill a multiple of 128 bits, the block should be zero-padded; a 16-bit field of 0x0000 following a structure indicates that there are no further structures in the current block.

Each data block in a chain of blocks might be of variable size and might contain multiple {*flowKey*, *flowData*, *tunnelData*} sets, but must always start with a *flowKey*. This is shown in Figure 8-6.

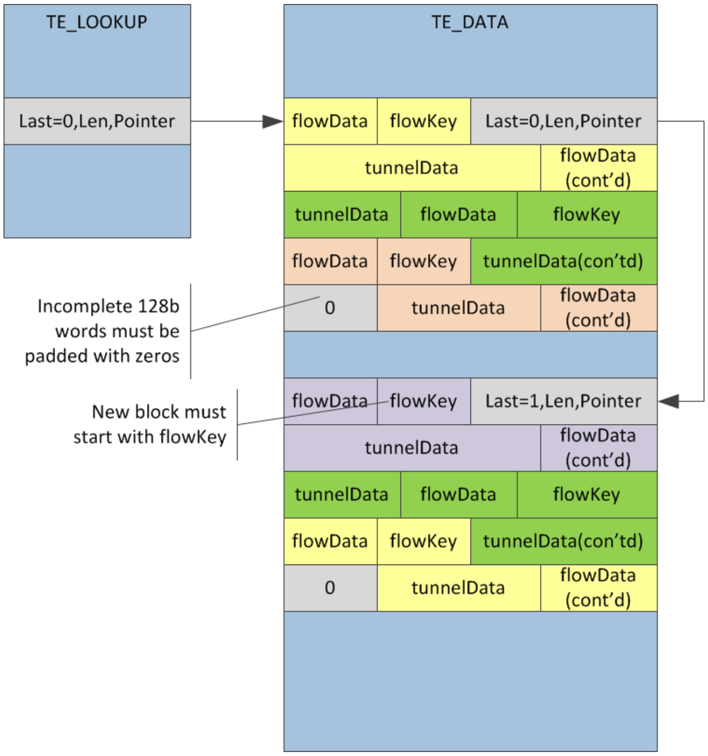


Figure 8-6 Key, Flow and Tunnel Data Packing

The Tunnel/Flow lookup starts a burst read in TE_DATA starting at the pointer provided by the TE_LOOKUP table, and continues bursting without stopping, while following the chain of blocks, and only stopping when the entire chain of data blocks are fully read, or when a match is found by the comparator. The comparator at the end of the TE_DATA stops comparing if the flow-key.mask is 0b, skipping to next block if any, or stopping completely (no match) if this was the last block.

Partitioning data structures in small blocks is useful to help software manage the memory pool. Updates to TE_DATA and TE_LOOKUP are permitted only between blocks read, this guarantees that a block cannot be altered while hardware is reading it.

8.10 Encapsulation/NAT

8.10.1 Data Structures

The following data structures are stored in the TE_DATA for encapsulation:

- nextPtr
- flowKey
- flowData
- tunnelData
- ngeData

The nextPtr is used to chain blocks together in the hash result. It uses the same format as TE_LOOKUP:

- *DataPointer* (16 bits, 15:0)
- *DataLength* (7 bits, 22:16)
- *Last* (1 bit, 23)
- Top 8 bits are ignored by hardware.

The flowKey defines following:

- mask: Defines which fields are present (TE_KEY)
  - Bit 0 = VSI present.
  - Bit 1 = Set to 0b.
  - Bit 2 = innerDMAC present.
  - Bit 3 = innerSMAC present.
  - Bit 4 = innerVID present. This is compared to the 12-bit value in the 8100 tag, not FTAG.VID.
  - Bit 5 = IP addresses are IPv6. Set only if innerDIP or innerSIP is included in hash.
  - Bit 6 = innerDIP present.
  - Bit 7 = innerSIP present.
  - Bit 8 = innerL4SRC present.
  - Bit 9 = innerL4DST present.
  - Bit 10 = innerPROT present.
  - Bit 11..13 = Any value (KeyExtra).
  - Bit 14 = Define if innerPROT=UDP is a match.
  - Bit 15 = Define if innerPROT=TCP is a match.
- VSI (16 bits, only lower 8-bits used, top 8-bits written as 0x00)
- innerDMAC (48 bits)
- innerSMAC (48 bits)
- innerVID (16 bits, VID is 12 bits only, innerVID[15:12] must be written as 0x0)

- innerDIP (32 bits/128 bits)
- innerSIP (32 bits/128 bits)
- innerL4DST (16 bits)
- innerL4SRC (16 bits)
- innerPROT (16 bits, only lower 8-bits used, top 8-bits written as 0x00)

The flowKey.mask defines which keys are present in the entry. The *hashMask* defines which packet fields are used in the search; it is copied from TE_DGLORT_DEC[m].*HashKeyConfig*, except for the *KeyIPv6* bit. hashMask.*KeyIPv6* is set if (KeyDIP | KeySIP) is set in *HashKeyConfig* AND the packet received is IPv6. The TE comparator will find a match if following conditions are met:

- if hashMask[13:0] matches the flowKey.mask[13:0],
- AND if all requested packet fields selected in the hashMask match the keys defined in the entry,
- AND if:
  - innerPROT == UDP and flowKey.mask[14] is constant for 1
  - OR innerPROT == TCP and flowKey.mask[15] is 1b
  - OR flowKey.mask[15:14] is 0b

Note: The bits 11 to 13 (*KeyExtra*) can be used to allow a shared hash table between multiple DGLORT decoder searches, enabling larger hash tables to more efficiently use the available resources.

The flowData defines:

- mask: Defines which fields are present (TE_ENCAP_FLOW_DATA)
  - Bit 0 = VNI present (TNI for GRE).
  - Bit 1 = counterPtr present.
  - Bit 2 = innerDMAC present.
  - Bit 3 = innerSMAC present.
  - Bit 4 = innerVID present.
  - Bit 5 = IP addresses are IPv6. Must equal flowKey.mask.KeyIPv6.
  - Bit 6 = innerDIP present. May be set only if *innerDIP* or *innerSIP* is included in flowKey.
  - Bit 7 = innerSIP present. May be set only if *innerDIP* or *innerSIP* is included in flowKey.
  - Bit 8 = innerL4SRC present.
  - Bit 9 = innerL4DST present.
  - Bit 10 = innerTTL present.
  - Bit 11 = ngeData present.
  - Bit 12 = tunnel present.
  - Bit 13 = tunnelPtr(1) or tunnelData(0).
  - bit 14 = loadTimetag in GENEVE data.
- VNI (32 bits) — VSID for NVGRE
  - VNI/VSID is actually 24 bits, so top 8 bits are ignored by hardware.

- counterPtr (16 bits)
- innerDMAC (48 bits)
- innerSMAC (48 bits)
- innerVID (16 bits, VID (12 bits) only. PCP/DEI are preserved.)
- innerDIP (32 bits/128 bits)
- innerSIP (32 bits/128 bits)
- innerL4DST (16 bits)
- innerL4SRC (16 bits)
- innerTTL (16 bits)
- ngeData (Nx32 bits)
  - First 16-bits is a mask indicating which of 16 x 32-bit words is present.
  - Next Nx32-bits is value of each word set in the mask.
- tunnelPtr (32 bits) or tunnelData (variable)
  - tunnelPtr has the same format as flowPtr. The last flag field is ignored by hardware.

The TE unit adds a tunnel header to the packet only if either tunnelPtr or tunnelData is provided.

The tunnelData defines:

- options (16 bits)
  - Bit 0..1 = Type of tunnel (00b=GENERIC, 01b=VXLAN, 10b=GENEVE, 11b=NVGRE)
  - Bit 2 = Tunnel is v6 or v4.
  - Bit 3 = outerSIP is present.
  - Bit 4 = outerTOS is present.
  - Bit 5 = outerTTL is present.
  - Bit 6 = outerL4DST is present.
  - Bit 7 = outerL4SRC is present.
  - Bit 8 = counterPtr is present.
  - Bit 9 = ngeData is present.
  - Bit 10 = loadTimetag in GENEVE data.
- outerDIP (required, 32 bits/128 bits)
- outerSIP (32 bits/128 bits)
- outerTOS (16 bits)
  - Only bottom 8 bits used. Top 8 bits are set to zero.
  - This field is interpreted as DS as per RFC2474, and equivalent to IPV6 TC.
- outerTTL (16 bits)
- outerL4DST (16 bits)
- outerL4SRC (16 bits)
- counterPtr (16 bits)

- ngeData (Nx32 bits)
    - First 16-bits is a mask indicating which of 16 words is present.
    - Next Nx32-bits is value of each word set in the mask.

The time tag is the TE_SYSTIME at the time the first word of the packet is received.

The counter data consist of:

- byte-counter (56 bits)
- packet-counter (48 bits)

8.10.2 Switch Roles for Encapsulation/NAT

Packets are switched twice:

- From CPU to TE (first pass)
- From TE to NETWORK (second pass)

The functions for each pass are:

- First pass:
    - ACL rule might set VID according to VSI.
    - If L2 switched:
        - L2Lookup(DMAC,VID) ⇒ DGLORT ⇒ TE
        - FFU(rule) ⇒ DGLORT ⇒ TE (dmac/smac not updated)
    - If IP routed:
        - FFU(dip) ⇒ Nexthop ⇒ DMAC,VID ⇒ L2Lookup ⇒ DGLORT ⇒ TE (dmac/smac updated)
    - Other rules apply normally.
- Second pass:
    - Normal physical router rules.

8.10.3 Packet Modifications

Table 8-1 lists the packet modifications for encapsulation/NAT. Unlisted fields are filled according to normal Ethernet/IP rules.

Table 8-1 Packet Modifications for Encapsulation/NAT

In	Out	Description
FTAG	FTAG	In: {SGLORT} ⇒ VSI {DGLORT,USER} ⇒ flowPtr Out: DGLORT: Passthrough or set. The option passthrough/set is defined by TE_DGLORT_DEC[m].*SetDGLORT*, the set value is defined in TE_DEFAULT_DGLORT if used. SGLORT: Passthrough or set. The option passthrough/set is defined by TE_DGLORT_DEC[m].*SetSGLORT*, the set value is defined in TE_DEFAULT_SGLORT if used. SWPRI,VPRI: Passthrough. VID: Passthrough. USER: Passthrough. FTYPE: Passthrough.
n/a	outerDMAC	Set to TE_DMAC,
n/a	outerSMAC	Set to TE_SMAC.
n/a	outerL3	DIP: From tunnelData. SIP: From tunnelData if present. From TE_SIP[vsi] otherwise. TOS/TC (8 bits): From tunnelData if present. Otherwise copy innerTOS or use a fix value (TE_CFG.*OuterTOS*) depending on TE_CFG.*DeriveOuterTOS* setting. The entire 8 bits is set. Note that this field has multiple names: TC in IPv6, renamed to DS in RFC2474. LENGTH: Derived from innerIP-Length or computed. CSUM: Updated from L4 header or computed or set to 0b. OPTIONS: None. FLAGS: Set to do-not-fragment. TTL: From tunnelData if present. Otherwise use TE_CFG.*OuterTTL* if different than 0. Otherwise copy *innerTTL* as received (pre-modifications by flow data directives). PROT: Derived from tunnel type (UDP for VXLAN, GRE for NVGRE). FLOW Set to 0b (IPv6 only).
n/a	outerL4	VXLAN: (UDP): L4SRC = From tunnelData if present, or hash(inner(DIP,SIP,L4SRC,L4DST,PROT)) using pre-modification data otherwise. L4DST = From tunnelData if present, or TE_DEFAULT_L4DST otherwise. Checksum = 0 or derived from inner checksum (derive only if inner is TCP or UDP, assume that inner TCP or UDP checksum is valid, passthrough is 0b). Length = Derived from inner L3 hdr length, fix value if not IP (TE_DEFAULT_LENGTH). NVGRE: No L4 header. Tunnel header: VNI 24-bit from flowData if present, or TE_VNI[vsi] otherwise. (GENEVE only) variable content from flowData & tunnelData.
L2 HDR	Inner L2 HDR	DMAC: From flowData if present, passthrough otherwise. SMAC: From flowData if present, passthrough otherwise. VLAN: Update VID from flowData if a new value is specified, and if VLAN present in packet (passthrough PCP/DEI unchanged). Otherwise, leave VLAN tag unchanged (present or absent).VLAN is detected as 0x8100. CUSTOM TAG: Leave unchanged, passthrough.

Table 8-1 Packet Modifications for Encapsulation/NAT [Continued]

In	Out	Description
L3 HDR	Inner L3 HDR	DIP: From flowData if present, passthrough otherwise. SIP: From flowData if present, passthrough otherwise. TTL: From flowData if present, passthrough otherwise. CSUM: Update if needed. Other: Passthrough.
L4 HDR	Inner L4 HDR	For UDP/TCP: L4SRC: From flowData if present, passthrough otherwise. L4DST: From flowData if present, passthrough otherwise. Checksum: Update if any inner(DIP,SIP,L4SRC,L4DST) have been updated.
Payload	Payload	Copy.
CRC/TIME-TAG	TIME-TAG	Updated if requested by switch.

8.10.4 Checksum and Packet Length Processing

During encapsulation, the tunneling engine needs to fill up the packet length and, potentially, also the VXLAN/GENEVE checksum. The tunneling engine supports the following engine (defined in TE_CFG).

- For non-IP packet:
  - TE computes the outer-IP packet from the actual length of the packet.
  - TE can be configured to set the checksum in the VXLAN or GENEVE header to 0b, or compute it from the actual original packet.
  - TE is in store-and-forward mode, as it has to compute packet length.
- For IP packet that are not UDP or TCP:
  - TE derives the outer-IP packet length from the inner-IP packet length.
  - TE can be configured to set the VXLAN or GENEVE checksum to 0b, or compute it from the packet modified payload (updated after NAT).
  - TE is in store-and-forward mode only if it needs to compute checksum from entire payload.
- For IP packet that are UDP or TCP:
  - TE derives the outer-IP packet length from the inner-IP packet length.
  - TE can be configured to set the VXLAN or GENEVE checksum to 0b, derive a new checksum from inner checksum in UDP or TCP header, or compute it from the packet modified payload (updated after NAT).
  - TE is in store-and-forward mode only if it needs to compute checksum from entire payload.

8.11 Decapsulation/DeNAT

The decapsulation/DeNat uses the following information:

- **FLOW-ID** — Identifies a packet flow that needs encapsulation
- **COUNTER-ID** — The tunneling end-point that performs the transport

8.11.1 Data Structures

The following data structures are stored in the TE_DATA for decapsulation:

- nextPtr
- flowKey
- flowData

The nextPtr is used to chain blocks together in the hash result. It uses the same format as TE_LOOKUP:

- *DataPointer* (16 bits, 15:0)
- *DataLength* (7 bits, 22:16)
- *Last* (1 bit, 23)
- Top 8 bits are ignored by hardware

The flowKey defines the following:

- mask: Defines which fields are present (TE_KEY)
  - Bit 0 = TEP-ID present.
  - Bit 1 = VNI present.
  - Bit 2 = innerDMAC present.
  - Bit 3 = innerSMAC present.
  - Bit 4 = innerVID present.
  - Bit 5 = IP addresses are IPv6. Set only if innerDIP or innerSIP is included in hash.
  - Bit 6 = innerDIP present.
  - Bit 7 = innerSIP present.
  - Bit 8 = innerL4SRC present.
  - Bit 9 = innerL4DST present.
  - Bit 10 = innerPROT present.
  - Bit 11..13 = Any value.
  - Bit 14 = Define if innerPROT=UDP is a match.
  - Bit 15 = Define if innerPROT=TCP is a match.
- TEP-ID (16 bits)
- VNI (32 bits)
  - VNI/VSID is actually 24 bits, so top 8 bits are ignored by hardware.

- innerDMAC (48 bits)
- innerSMAC (48 bits)
- innerVID (16 bits, VID is 12 bits only, innerVID[15:12] must be written as 0x0)
- innerDIP (32 bits/128 bits)
- innerSIP (32 bits/128 bits)
- innerL4DST (16 bits)
- innerL4SRC (16 bits)
- innerPROT (16 bits)

The flowKey.mask defines which keys are present in the entry. The *hashMask* defines which packet fields are used in the search; it is copied from TE_DGLORT_DEC[m].*HashKeyConfig*, except for the KeyIPv6 bit. hashMask.KeyIPv6 is set if (KeyDIP | KeySIP) is set in *HashKeyConfig* AND the inner header of the packet received is IPv6. The TE comparator finds a match if following conditions are met:

- If hashMask[13:0] matches the flowKey.mask[13:0],
- AND if all requested packet fields selected in the hashMask match the keys defined in the entry
- AND if:
  - — innerPROT == UDP and flowKey.mask[14] is 1b
  - — OR innerPROT == TCP and flowKey.mask[15] is 1b
  - — OR flowKey.mask[15:14] is 00b

The flow data defines:

- mask — Defines which fields are replaced (TE_DECAP_FLOW_DATA):
  - — Bit 2:0 = Outer header disposition

    000b = Delete outer header.
    001b = Leave outer header in place.
    010b = Move outer header to Trailer at end of packet, with an additional 8-byte Trailer Descriptor (refer to Section 8.11.3)
    All other values are reserved.
  - — Bit 3 = DGLORT.
  - — Bit 4 = innerDMAC present.
  - — Bit 5 = innerSMAC present.
  - — Bit 6 = innerVID present.
  - — Bit 7 = IP addresses are IPv6. Must equal flowKey.mask.KeyIPv6
  - — Bit 8 = innerDIP present. May be set only if *innerDIP* or *innerSIP* is included in flowKey.
  - — Bit 9 = innerSIP present. May be set only if *innerDIP* or *innerSIP* is included in flowKey.
  - — Bit 10 = innerTTL present.
  - — Bit 11 = innerL4SRC present.
  - — Bit 12 = innerL4DST present.
  - — Bit 13 = counterPtr present.

- DGLORT (16 bits)
- innerDMAC (48 bits)
- innerSMAC (48 bits)
- innerVID (16 bits, VID (12 bits) only, PCP/DEI are preserved,)
- innerDIP (32 bits/128 bits)
- innerSIP (32 bits/128 bits)
- innerTTL (16 bits)
- innerL4DST (16 bits)
- innerL4SRC (16 bits)
- counterPtr (16 bits)

The counterData structure consist of:

- byte-counter (64 bits)
- packet-counter (64 bits)

8.11.2 Packet Modifications

Table 8-2 lists the packet modifications for decapsulation/DeNAT.

Table 8-2 Packet Modifications for Decapsulation/DeNAT

In	Out	Disposition
FTAG	FTAG	In: {DGLORT,USER} ⇒ flow-id SWPRI,VPRI,VID,USER ⇒ passthrough Out: DGLORT: Update from flowData if present, passthrough/set otherwise. The option passthrough/set is defined by TE_DGLORT_DEC[m].*SetDGLORT*. The set value is defined in TE_DEFAULT_DGLORT if used. SGLORT: Passthrough or set. The option passthrough/set is defined by TE_DGLORT_DEC[m].*SetSGLORT*. The set value is defined in TE_DEFAULT_SGLORT if used. SWPRI,VPRI: Passthrough VID: Passthrough USER: Passthrough FTYPE: Passthrough
Outer Header		Follow delete/leave-as-is/move directive from flowData request. The disposition options are: • Delete the outer header. • Leave the outer header in place. In this case, update the L4 checksum if an inner header was updated (deNAT). If the original outer L4 checksum was 0x0000, leave it unchanged. • Move the outer header to Trailer at the end of packet, with an additional 8-byte Trailer Descriptor (refer to Section 8.11.3).
Inner L2 HDR	L2 HDR	DMAC: Update from flowData if present, passthrough otherwise. SMAC: Update from flowData if present, or passthrough otherwise. VLAN: Update VID from flowData if present and tag present, or passthrough otherwise.

Table 8-2 Packet Modifications for Decapsulation/DeNAT [Continued]

In	Out	Disposition
Inner L3 HDR	L3 HDR	DIP: Update from flowData if present, or passthrough otherwise. SIP: Update from flowData if present, or passthrough otherwise. TTL: Update from flowData if present, or passthrough otherwise. CSUM: Update checksum. TOS/TC: Interpreted as DS field (as per RFC2474), TE will update ECN bits of this field copying the one from the outer header if and only if the outer header is deleted or moved to the tail of the packet.
Inner L4 HDR	L4 HDR	For UDP/TCP only: SRC: From flow or leave as is. DST: From flow or leave as is. CSUM: Update checksum if any inner(DIP,SIP,L4SRC,L4DST) have been updated.
Payload	Payload	Copy.
n/a	Trailer	If requested in flowData mask bits 2:0, Trailer contains the saved outer header and an additional 8-byte Trailer Descriptor (refer to Section 8.11.3).
TIME-TAG	TIME-TAG	Update time tag.

8.11.3 Trailer for Saved Outer Header

The *flowData* mask bits 2:0 define the disposition of the outer header. In some applications, the hypervisor might need the outer header for further information on the packet. In this case, the outer header can be removed but saved. It becomes a trailer to the packet, inserted before the time tag, with an additional 8-byte Trailer Descriptor.

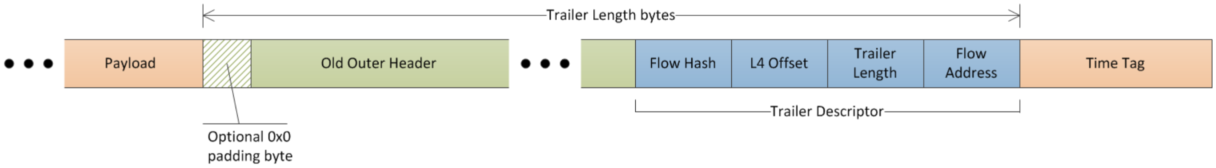


Figure 8-7 Trailer Descriptor

The trailer is:

- An optional 0x0 padding byte, inserted if Payload has odd length.
- Saved outer header (guaranteed to be an even number of bytes).
- Trailer Descriptor (8 bytes):
  - — Flow Hash (16 bits) — Full 16-bit flow hash value, as specified by TE_DGLORT_DEC.*HashKeyConfig*. This value is provided even when using direct flow lookup.
  - — L4 Offset (16 bits) — Number of bytes from the start of the saved outer header (after padding byte, if present) to the end of the saved L3 header.
  - — Trailer Length (16 bits) — Length of the trailer in bytes, including 8 bytes of additional data and padding byte if present. If odd, there is a padding byte.
  - — Flow Address (16 bits) — Index in TE_DATA where the *flowData* used for this packet is found. Specifically, for direct lookup this is TE_LOOKUP.*DataPointer*; for hash lookup this is the index of the 128-bit TE_DATA word that contains the mask of the matching *flowData*.

8.11.4 Checksum and Packet Length Processing

During decapsulation, the tunneling engine proceeds according to TE_CFG.*VerifyDecapCSUM*:

- If set to 0b, TE ignores the outer checksum regardless of its value. TE is operating in cut-through in this case.
- If set to 1b, TE validates the checksum if it is not 0. TE is operating in store-and-forward if the checksum is validated. If the checksum is incorrect, TE follows the TE_TRAP_CONFIG.*DECAP_BAD_CSUM* setting which shall be either PASS_WITH_ERROR or DROP (all other settings are undefined behavior).

8.12 NAT/DeNAT Only

Network Address Translation in either direction, without any encapsulation and decapsulation, is done using the Encapsulation function without providing any tunneling pointer or tunnel data.

8.13 Time Tagging

The TE unit supports time tagging. The TE receives a time reference clock from the FM10000 centralized module interface (SYSTIME) and keeps a local time reference (40 bits).

For each packet received, the TE captures local time at the moment the first word is received. At egress, the TE has an option to replace the last four bytes of the packet with the lower 32 bits of the time tag captured, or leave these bytes untouched. The update is controlled by the switch time request encoded into the port channel interface. The content is not changed if he switch has not requested a time capture.

8.14 Aging

The TE unit uses the TE_USED[0..1023] table to track which flows are still active. For each packet, if a flow matches the packet, the flowPtr associated with the matching key is used to set bit TE_USED[flowPtr/64].*Used*[flowPtr%64].

8.15 Updating Registers

Most registers in the tunneling engine could be changed any time while traffic is running without causing any adverse effect on the traffic itself except for a few critical registers:

- TE_DGLORT_MAP
- TE_DGLORT_DEC
- TE_DATA
- TE_LOOKUP
- TE_SRAM_CTRL
- TE_PORTS
- TE_TUN_HEADER_CFG
- TE_EXVET

Those registers requires special handling as described in the next sections.

8.15.1 Changes to TE_DGLORT_MAP/TE_DGLORT_DEC

The entries in the TE_DGLORT_DEC table are used multiple times in the pipeline. If entries in the DGLORT_DEC tables are changed while packets are using those specific entries, the actual packet processing might be unpredictable. It is recommended to only change entries in the DGLORT_MAP and DGLORT_DEC tables when there is a certainty they are not in use by the switch and tunneling engine at that moment. The most complete procedure is:

1. Change the switch to avoid forwarding traffic to the DGLORT set affected.
2. Ensure all packets are drained from the switch and the TE that might use these entries:
   - For switch, use CM_EPOCH registers.
   - For each TE affected, use following procedure:
     a. Read (in=TE_FRAMES_IN).
     b. Poll TE_FRAMES_DONE until ((TE_FRAMES_DONE-in) & 0x80000000) is 0.
3. Change the TE_DGLORT_MAP/TE_DGLORT_DEC tables.
4. Change the switch to re-enable traffic to use updated DGLORT entries

A quicker and simpler alternative method is to temporarily disable this DGLORT in TE:

1. Disable the usage of this DGLORT set by changing the DGLORT_MAP.
2. Wait for traffic to drain in TE:
   a. Read (in=TE_FRAMES_IN).
   b. Poll TE_FRAMES_DONE until ((TE_FRAMES_DONE-in) & 0x80000000) is 0.
3. Change DGLORT_DEC.
4. Re-enable DGLORT set by restoring DGLORT_MAP.

This second procedure is acceptable if and only if the DGLORT mapping and TE_TRAP_CONFIG is such that packets still in flight from the switch, and potentially using the affected entry, are dispatched in an acceptable manner from a network perspective while the change is being done.

8.15.2 Recovering Unused Data

The tunneling engine data path and management path are pipelined differently, which create a possible race condition when writing into TE_DATA and TE_LOOKUP. The hardware will perform TE_DATA writes between bursts, and always ensures writes to TE_DATA are done before a follow-on TE_LOOKUP write is done. This ensures that any packet processing has a consistent use of the data structures when new data structures are added to the dataset. However, there is a possible problem when data is removed and re-used right away. The following is the recommended procedure.

Adding new data blocks in TE_DATA

1. Write new data in TE_DATA.
2. Change pointers (in TE_LOOKUP or TE_DATA).

TE_DATA is now fully inserted.

Re-using data blocks in TE_DATA

1. Change pointers to bypass block to recover.
2. Read (in=TE_FRAMES_IN).
3. Poll TE_FRAMES_DONE until ((TE_FRAMES_DONE-in) & 0x80000000) is 0b.

TE_DATA is now unused.

8.15.3 Changes to TE_SRAM_CTRL

The TE_SRAM_CTRL is intended for debug purpose and could be used to simulate correctable or uncorrectable errors on traffic and debug software response to those events. This is described Section 10.0.

8.15.4 Changes to TE_PORTS/TE_TUN_HEADER_CFG/ TE_EXVET

These registers may not be changed while traffic is flowing. They are assumed not to change, and may be consulted more than once in the processing of each packet.

8.16 Hash Computation

The `hash ()` function used is the Ethernet CRC32 truncated to the lower 16 bits, with the CRC initialized to 0, and computed over a vector of bytes (`byteArray`) where the bytes are loaded from lowest significant byte to highest significant byte. The fields are padded with zeros when needed.

8.16.1 Flow Hash Computation

The flow hash computation is over the fields {KEY_MASK,ID,VNI,DMAC,SMAC,VID,DIP,SIP,L4SRC,L4DST,PROT}, with unused fields (those not indicated in KEY_MASK) zeroed out. For IPv4, only the least significant 4 bytes of DIP and SIP are nonzero. VID is zero-padded to 16 bits.

ID refers to VSI for encapsulation, or TEP-ID for decapsulation. VNI (for VXLAN/GENEVE decapsulation) and VSID (for NVGRE decapsulation) are treated equivalently here; KEY_MASK.*KeyVNI* must always be 0b for encapsulation.

```
  h = hash(byteArray[0..59])
```

where the byte array is set to:

```
byteArray[0] = KEY_MASK[7:0]
byteArray[1] = KEY_MASK[8:15]
byteArray[2] = KEY_MASK.KeyID ? ID[7:0] : 0 // ID is VSI (encap) or TEP-ID (decap)
byteArray[3] = KEY_MASK.KeyID ? ID[15:8] : 0
byteArray[4] = KEY_MASK.KeyVNI ? VNI[7:0] : 0 // VNI or VSID
byteArray[5] = KEY_MASK.KeyVNI ? VNI[15:8] : 0
byteArray[6] = KEY_MASK.KeyVNI ? VNI[23:16] : 0
byteArray[7] = 0
byteArray[8] = KEY_MASK.KeyDMAC ? DMAC[7:0] : 0
...
byteArray[13] = KEY_MASK.KeyDMAC ? DMAC[47:40] : 0
byteArray[14] = KEY_MASK.KeySMAC ? SMAC[7:0] : 0
...
byteArray[19] = KEY_MASK.KeySMAC ? SMAC[47:40] : 0
byteArray[20] = KEY_MASK.KeyVID ? VID[7:0] : 0
byteArray[21] = KEY_MASK.KeyVID ? VID[11:8] : 0 // VID is zero padded to 16b
byteArray[22] = KEY_MASK.KeyDIP ? DIP[7:0] : 0
...
byteArray[37] = KEY_MASK.KeyDIP ? DIP[127:120] : 0 // SIP/DIP[127:32] loaded as 0 for IPv4
byteArray[38] = KEY_MASK.KeySIP ? SIP[7:0] : 0
...
byteArray[53] = KEY_MASK.KeySIP ? SIP[127:120] : 0 // SIP/DIP[127:32] loaded as 0 for IPv4
byteArray[54] = KEY_MASK.KeyL4DST ? L4SRC[7:0] : 0
byteArray[55] = KEY_MASK.KeyL4DST ? L4SRC[15:8] : 0
byteArray[56] = KEY_MASK.KeyL4DST ? L4DST[7:0] : 0
byteArray[57] = KEY_MASK.KeyL4DST ? L4DST[15:8] : 0
byteArray[58] = KEY_MASK.KeyPROT ? PROT : 0
byteArray[59] = 0
```

8.16.2 Default Outer L4SRC Hash Computation

The default L4SRC hash computation is over the fields {DIP,SIP,L4SRC,L4DST,PROT}. For IPv4, only the least significant 4 bytes of DIP and SIP are non-zero.

```
h = hash(byteArray[0..37])
h |= 0xc000 // restrict range to >= 49152 (dynamic UDP port)
```

where the byte array is set to:

```
byteArray[0] = DIP[7:0]
byteArray[1] = DIP[15:8]
byteArray[2] = DIP[23:16]
byteArray[3] = DIP[31:24]
byteArray[4] = DIP[39:32] // SIP/DIP[127:32] load to 0 for IPv4
...
byteArray[15] = DIP[127:120] // SIP/DIP[127:32] load to 0 for IPv4
byteArray[16] = SIP[7:0]
...
byteArray[35] = L4DST[15:8]
byteArray[36] = PROT
byteArray[37] = 0
```

NOTE: This page intentionally left blank.

9.0 Peripherals

The following functions are defined in this chapter:

- Interrupt Controller
- Counter Rate Monitor
- SerDes Management and Temperature/Voltage Sensing
- I^{2}C Controller
- MDIO Controller
- GPIO Controller
- SPI Interface
- LED Controller
- System Time and IEEE 1588

9.1 Interrupt Controller

The interrupt controller controls interrupt dispatching across different modules. This section does not describe interrupt processing within PCI Express interface.

The interrupt controller includes the following registers:

- *XXX*_IP — Logs interrupts. Software reads to detect a particular event and writes 1b to clear the event.
- *XXX*_IM — Mask interrupts. Software writes 1b to prevent a selected event from being reported in the hierarchy.
- CORE_INTERRUPT_DETECT — Reports interrupts from manageable blocks.
- CORE_INTERRUPT_MASK — Selects which core interrupts are logged globally.
- GLOBAL_INTERRUPT_DETECT — Reports top level interrupts.
- INTERRUPT_MASK_*XXX* — Selects which interrupts are sent to which interrupt outputs.

Figure 9-1 shows the overall interrupt hierarchy.

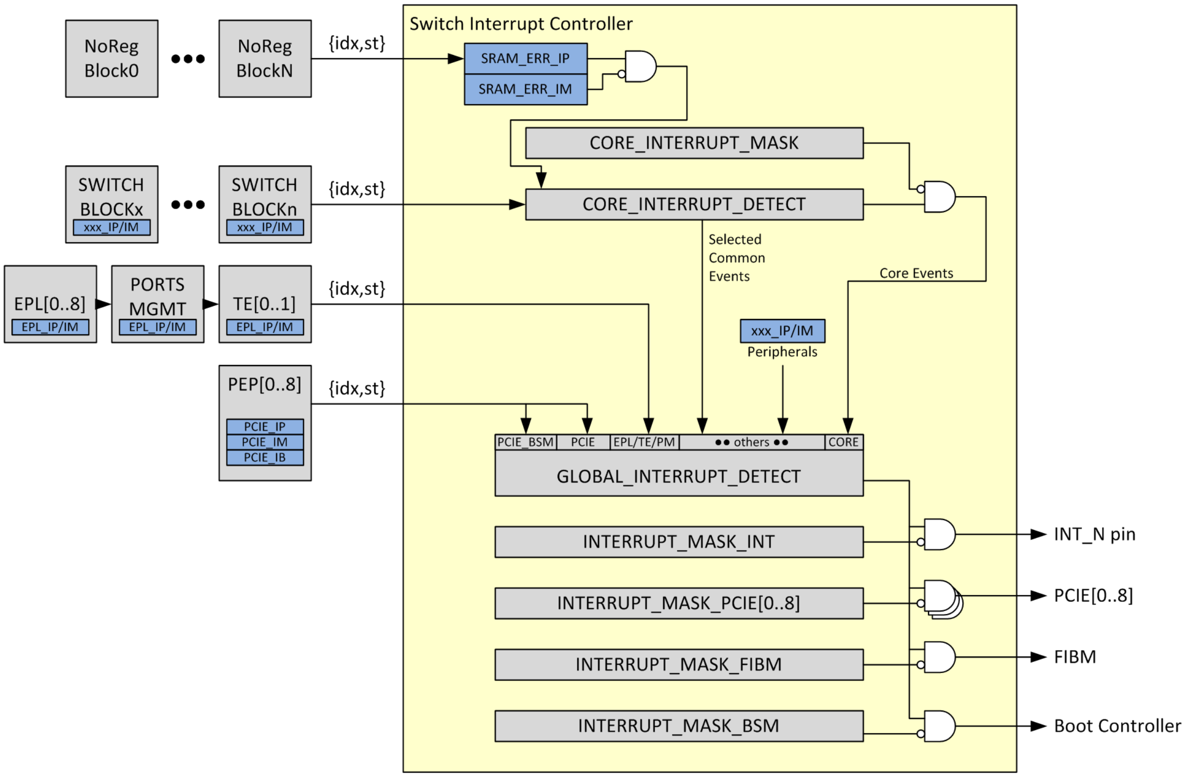


Figure 9-1 Interrupt Hierarchy

There are five types of sources:

- Ports (includes ports management, tunneling engines and EPLs)
- PCIEs
- Manageable blocks
- Unmanageable blocks
- Peripherals

There are 12 outputs:

- Interrupt pin (INT_N)
- 9 PCIe End Points
- 1 In-band Management
- 1 Boot State Machine

9.1.1 EPL, PORTS Management, Tunneling Engine and PCIe Interrupts

All EPL blocks, tunneling engines and Ethernet port management blocks are daisy chained through a serial chain. The vector posted ({index,status}) to the interrupt controller is an 8-bit vector with the lower 7 bits indicating the EPL number (only 0..8 are possible) and upper bit indicating if an unmasked interrupt is pending or not (that is, whether |(EPL_IP & ~EPL_IM) is true as an example).

All PCIE blocks are similarly daisy chained. In this case each PCIE host interface can generate 2 different indices, corresponding to its two different interrupt masks. Interrupts unmasked in PCIE_IM are reported in GLOBAL_INTERRUPT_DETECT.*PCIE*, while interrupts unmasked in PCIE_IB are reported in GLOBAL_INTERRUPT_DETECT.*PCIE_BSM*.

Each EPL and PCIE generate a vector immediately after they are taken out of reset, and also whenever its masked interrupt state becomes non-zero or go to zero. If the vector has not been sent yet and a new condition arises that requires sending an updated vector (this could happen if the chain is delayed temporarily and an event creates an interrupt or software just cleared all interrupts in this EPL or PCIE), the new vector replaces the one that has not left yet. This ensures that the latest state is reported to the MGMT module.

The interrupt controller, upon receiving the vector, sets or clears the corresponding bit in the GLOBAL_INTERRUPT_DETECT EPL[i], PCIE[i] or PCIE_BSM[i] fields to indicate presence on an interrupt in EPLs or PCIEs or not.

9.1.2 Switch Interrupts

All switch blocks (those having accessible registers) are daisy chained through an interrupt serial chain to report events logged into their respective *XXX*_IP/IM registers. The vector posted to the interrupt controller ({index,status}) is a 7-bit index and a 1-bit status indicating if the corresponding event has been cleared or not. The events are logged into the CORE_INTERRUPT_DETECT register (set to 1b or set to 0b depending on status), a mask (CORE_INTERRUPT_MASK) allows to select which one is reported in the GLOBAL_INTERRUPT_DETECT.*CORE*.

The FPP head/tail interrupts reported in CORE_INTERRUPT_DETECT are also replicated in GLOBAL_INTERRUPT_DETECT for convenience.

9.1.3 SRAM Errors

The processing of SRAM errors varies depending on the type of blocks:

- SRAM errors in the main packet memory are reported in SRAM_ERR_IP/IM. Unmasked interrupts are reported in CORE_INTERRUPT_DETECT.
- SRAM errors in the ingress/egress crossbars are in two categories:
    - Those affecting data cause the affected packet to be marked with having a framing error and are treated like other packet data errors.
    - Those affecting critical control data cause the crossbar to lock up and are reported in INGRESS_ERR and EGRESS_ERR events in the CORE_INTERRUPT_DETECT register. The switch manager must reset and restart the switch to repair these errors.

- SRAM errors in other function blocks are reported in their respective *XXX*_IP and *XXX*_IM registers:
  - Boot memory errors are reported in BSM_IP.
  - CRM memory errors are reported in CRM_IP.
  - FPP memory errors are reported in HEAD_IP or TAIL_IP.
  - Scheduler memory are errors reported in SCHED_IP.
  - Tunneling engine memory errors are reported in TE_IP.
  - MODIFY memory errors are reported in MODIFY_IP.

The errors reported could be correctable or uncorrectable. The software can detect the events and decide how to process them.

9.1.4 Peripherals Interrupts

The interrupt controller also reports peripherals interrupts directly in GLOBAL_INTERRUPT_DETECT. Those sources are GPIO, MDIO, I2C, SOFT, CRM, PINS, FIBM, and SBUS_PCIE.

9.1.5 Posting Interrupts

There are 12 possible interrupts outputs:

- To INT_N pin which can be used to interrupt an external processor
- To each PCIe end point which can generate an in-band MSI-X message
- To FIBM which will generate a packet
- To boot state machine which can trigger an NVM sequence

The interrupt controller provides 12 INTERRUPT_MASK_*XXX* registers for each of these outputs (9 for PCIEs, one for FIBM, one for BSM, one for the INT_N pin), allowing the user to select which interrupt is posted to which output.

9.2 Counter Rate Monitor

The FM10000 provides general counter rate monitoring and memory management functions to reduce the need for the CPU to perform repetitive periodic operations in the switch. This module's primary purpose is counter/state monitoring, but it includes other services as well. The services offered are:

- Monitor rate counter changes (too fast or too slow).
- Report state changes.
- Monitor queue sizes.
- Report max queue sizes.
- Copy memory blocks.
- Initialize memory blocks.
- Compute checksums.

The CRM structure is divided into four sets of registers:

- 2048 data set memory.
- 64 commands definition registers.
- Control register.
- Interrupt registers.

The control and status registers are used to control the CRM globally. The control register is used to start and stop the CRM and define the index range of the sequence of commands to execute. The status register is used to define the next command to execute (before start) or report current one being executed (while running) and the current state (running/stopped) of the CRM module.

Table 9-1 CRM Control/Status Registers

Register	Field	Bits	Definition
CRM_CTRL	*Run*	1	Start or stop the CRM. Stopping may require some time as the current command has to complete.
	FirstCommandIndex	6	Index of first command of the sequence.
	LastCommandIndex	6	Index of last command of the sequence.
	Continuous	1	Defines if the sequence of commands is executed continuously or only once. 0b = Once 1b = Continuously If executed only once, the run bit is cleared at the end of the sequence.
	TickPrescale	5	Defines (in power of 2) the prescaler to apply to the PCIE_REFCLK to produce the reference time for the CRM.
	TimeoutPrescale	5	Defines (in power of 2) the maximum time for a read/write to complete. An alarm is sent to the watchdog if this time is exceeded.
CRM_STATUS	*Running*	1	(READ ONLY) Indicates if CRM is running or stopped. 0b = CRM stopped. 1b = CRM running. ***Note:*** Stopping the CRM (turning 'Run' bit to 0b) may be delayed until the current command being executed completes.
	CommandIndex	6	If stopped by user, this is the index to the next command to execute. If stopped because the sequence completed, the command index is reset to the FirstCommandIndex. The value must be set to any command in the sequence before starting the CRM. If running, this is the index of the current command being executed.
CRM_TIME	*Tick*	32	Current tick counter.

The command registers are used to define commands. These commands can have any data size set associated with each one. When running, the CRM executes the sequence of commands defined in the CRM_CTRL register, once or repetitively. Each command has its own interrupt bit in CRM_IP which can be used to post an interrupt the processor. Each interrupt is maskable using the CRM_IM register.

It is possible for the CPU, at any time, to stop the current monitoring process and execute a special command (or series of command). The process would be:

- Stop CRM.
- Wait for CRM to indicate it has stopped.
- Save the index of the next command to execute.
- Load a a temporary list of commands (possibly by using unused command space).

- Launch execution of that new program (single execution), wait for its completion by polling corresponding CRM_IP bit for this command (or waiting for interrupt).
- Re-active the previous program at the place it was stopped.

The CRM can only be stopped at a command boundary. If a single command is going through a large data set, the CRM may take a while before stopping.

The commands register set includes the following registers.

Table 9-2 CRM Control/Status Registers

Register	Field	Bits	Definition
CRM_COMMAND[0..63]	*Command*	3	Defines the command to execute (see Table 9-3).
	DataIndex	11	Pointer of the data section for this command. Not used for all commands.
	Count	20	Defines the number of registers to walk through by this command. If the command needs a data section, the *Count* is limited to 2048 maximum (total size of data set). If the command does not need a data section (for example a memory set), the limit is 1,048,575. The count is the number of registers, not the number of bytes or words.
CRM_REGISTER[0..63]	*BaseAddress*	24	First register address.
	Size	2	Defines register size: 00b = 32-bit 01b = 64-bit 10b = 96-bit 11b = 128-bit
	BlockSize1Shift	4	Defines register block size in power of 2. Block size is 1<<*BlockSize1Shift*.
	Stride1Shift	4	Defines stride to next block in power of 2. Stride is 1<<(*Stride1Shift*+1).
	BlockSize2Shift	4	Defines second level register block size in power of 2. Block size is 1<<BlockSize2Shift.
	Stride2Shift	4	Defines second level stride to next block in power of 2. Stride is *1<<(Stride2Shift+1)*.
CRM_PERIOD[0..63]	*Interval*	32	Defines time interval before executing this command. The timebase ('CRM tick') is a prescale PCIE_REFCLK (1 to 65536 in power of 2). A value of 0 is as fast as possible.
	LastTick	32	Defines last time the command was executed.
CRM_PARAM[0..63]	*Data*	32	Defines the parameters for each command. Interpretation depends on the command.

The data set is a two dimensional array of 32-bit data, CRM_DATA[0..2047,0..1]. The first index is the data pointer (as defined in the CRM_COMMAND register) while the second is interpreted depending on the command used.

Table 9-3 lists the commands.

Table 9-3 CRM Command Definition

Command	Definition	CRM_PARAM	CRM_DATA[0]	CRM_DATA[1]	Interrupt
Set(0)	Initialize a memory block.	Value to set[1]	N/A	N/A	Set after initialization completes.
Copy(1)	Copy a memory block from one location to another location.	Destination register address	N/A	N/A	Set after copy completes.
CountRateTooFast(2)	Increment counter if a register changed by more than a certain limit. If the register size is greater than 32, CRM reads and uses only the least significant 32 bits for this operation.	Limit	Last value	Count	Set if at least one counter in the data set exceeded the last value.
MonitorChange(3)	Check for change and save it. If the register size is greater than 32, CRM reads and uses only the least significant 32 bits for this operation.	Mask	Last value	Tick at change	Set if at least one counter in the data set changed.
SaveMax(4)	Compare value to last and save if max. If the register size is greater than 32, CRM reads and uses only the least significant 32 bits for this operation.	Mask	Maximum value	Tick at change	Set when a new maximum is found.
CountRateTooSlow(5)	Increment counter if register change by less than a certain limit. If the register size is greater than 32, CRM reads and uses only the least significant 32 bits for this operation.	Limit	Last value	Count	Set if at least one counter in the data set is not incrementing fast enough.
CountGreaterThan(6)	Count if greater or equal. Increment counter if register is greater or equal to limit. If the register size is greater than 32, CRM reads and uses only the least significant 32 bits for this operation.	Mask	Limit	Count	Set if at least one counter in the data set is increased.
Checksum(7)	Compute Checksum. Read table and compute an XOR over the entire table. If the register size is greater than 32 bits, each 32-bit word is added to the XOR.	Expected checksum	N/A	N/A	Set if checksum computed does not match checksum expected.

[1] The CRM_PARAM register is a 32-bit register, for the SET MEMORY command, this value is replicated to match the size of the set (32,64,96,128).

A single command might be interested in a set of registers that is not contiguous in memory. For this reason, the register definition includes two levels of block sizes and strides that allow a single command to be performed efficiently over a large set of registers.

The basic pseudo code of the overall CRM module is:

```
while (1)

if ( Run )

# Check if time to run this command
if ( tick - CRM_PERIOD[CommandIndex].LastTick > CRM_PERIOD[CommandIndex].Interval )

    if ( tick - CRM_PERIOD[CommandIndex].LastTick > CRM_PERIOD[CommandIndex].Interval*2 )
        CRM_PERIOD[CommandIndex].LastTick = tick
    else
        CRM_PERIOD[CommandIndex].LastTick += CRM_PERIOD[CommandIndex].Interval

    baseAddress = CRM_REGISTER[CommandIndex].BaseAddress
    count0 = 0
    count1 = 0
    count2 = 0
    if (registerSize == 0)
        size0=1
    else if (registerSize == 1)
        size0=2
    else if (registerSize == 2)
        size0=4
    else if (registerSize == 3)
        size0=4
    size1 = 1 << CRM_REGISTER[CommandIndex].BlockSizeShift1
    size2 = 1 << CRM_REGISTER[CommandIndex].BlockSizeShift2
    stride1 = 1 << (CRM_REGISTER[CommandIndex].StrideShift1+1)
    stride2 = 1 << (CRM_REGISTER[CommandIndex].StrideShift2+1)

    foreach "i" in (0..CRM_COMMAND[CommandIndex].Count)

      # Compute address
      address = baseAddress + count0 * size0 + count1 * stride1 + count2 * stride2

      # Execute command at address "address" for register size selected
      switch (CRM_COMMAND[CommandIndex].Command)
        case 0:
          # Set memory
        case 1:
          # Copy memory
        case 2:
          # Monitor rate too fast
        case 3:
          # Monitor changes
        case 4:
          # Monitor max
        case 5:
          # Monitor rate too slow
        case 6:
          # Count changes greater than limit
        case 7:
          # Add register content to checksum

       # Advance counter
       count0++
       if ( count0 == size1 )
         count0 = 0
         count1++
         if ( count1 == size2 )
            count1 = 0
            count2++

# Pass to next command
if ( CommandIndex = LastCommandIndex )
    CommandIndex = FirstCommandIndex
else
    CommandIndex++
```

9.3 SerDes Management and Temperature/ Voltage Sensing

The FM10000 supports two Serial Bus rings linking SerDes into two chains; one chain for PCIe (33 SerDes) and one chain for Ethernet (36 SerDes). Each chain is controlled by a serial bus controller and include a local SPICO micro-controller and a ROM for initial software for the SPICO micro-controller. Each chain also includes a pair of resistor monitor (RMON) for automatic calibration and one ring oscillator (PMRO). The PCIe chain includes a thermal and voltage sensor which supports 1 local and 8 remote thermal sensors (located in different locations in the die) and 8 voltage sensors (also sensing different locations in the die).

Figure 9-2 shows the block diagram of the system.

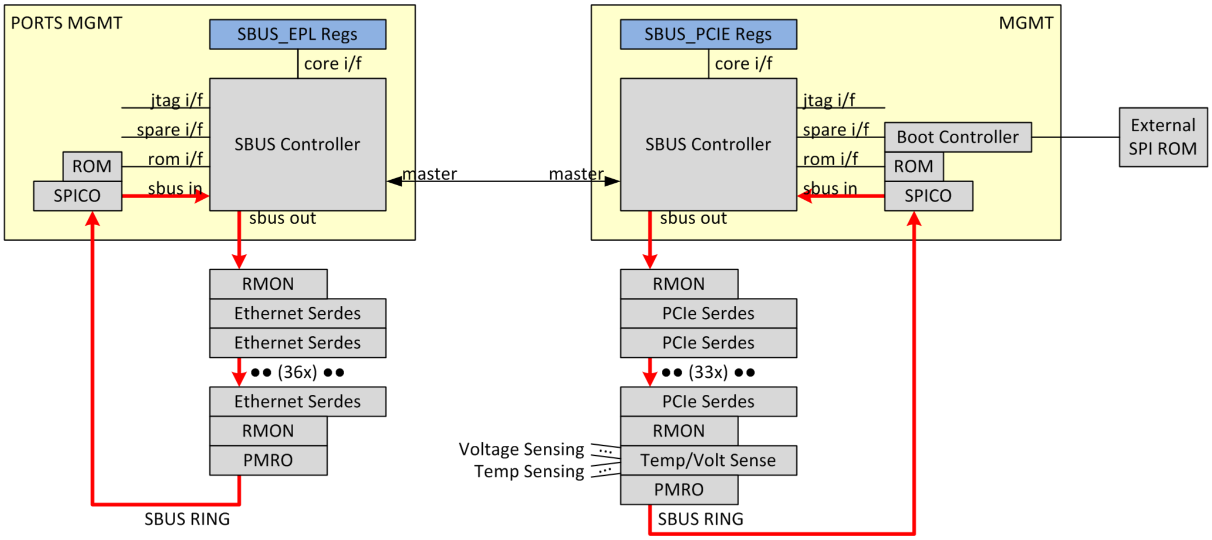


Figure 9-2 Serial Bus Overview

Each SBUS controller supports 6 interfaces:

- CORE interface
- JTAG interface
- ROM interface
- SPARE interface
- SBUS interface (in and out)
- MASTER-MASTER interface (linking the two chains)

The CORE interface is used to provide an access to the SBUS for software. The interface consist of the following set of registers:

- SBUS_EPL_*XXX* registers to access components on the Ethernet SBUS chain.
- SBUS_PCIE_*XXX* registers to access components on the PCIE SBUS chain.

The JTAG interface is connected to JTAG scan chain and used for testing.

The ROM interface is connected to a local ROM and is used to download a small diagnostic code to the micro-controller for production purpose. For normal operation, the micro-controller code is coming from either the switch manager software or an external Serial Boot ROM attached to the FM10000.

The SPARE interface is used on the PCIe chain only, it is connected to the Boot Controller to provide a fast path to PCIe SerDes download.

The SBUS interfaces (in and out) are connected to the SerDes, RMONs, thermal/voltage sense controller and SPICO.

The MASTER-MASTER interface links the two chains.

The SBUS is a slow bus (50 MHz for PCIe and 78.125 MHz for Ethernet) and is a long ripple chain. As a result the access time to any register in any SerDes is long, up to 1us. To prevent the internal management bus of the switch to be on hold for that long, the MGMT module and PORT MGMT modules provide each a command register that latches a command issued from a host (local or remote) and frees up the internal bus while the command is being executed. The host may poll the command register at a later time to see if the command has been executed. The host must wait for a previous command to complete before a new one can be issued.

The command register contains:

- Device (0..N)
- Register (0..255)
- Operation (read, write, reset)
- Execute (transition from 0b to 1b starts the command)
- Busy (0b=idle, 1b=executing a command)

The transaction on the serial bus is started as soon as the command is written, a read back from the register returns a busy bit to indicate that the command is being executed. The busy bit is cleared when the command execution completes.

The software access for an SBUS Read or Write command (for XXX either EPL or PCIE) is:

```
sbus_xxx_command (device, register, command, data)
{
 // Write data
 SBUS_XXX_REQUEST = data;

 // Start command
 SBUS_XXX_COMMAND = (1 << 24) /*Execute*/
                  + (((command==Write) ? 0x21 : (command==Read) ? 0x22 : 0x0) << 16)
                  + (device << 8)
                  + (register);

 // Wait for command to complete
 while (SBUS_XXX_COMMAND.Busy) yield();

 // Check result status (optional; failure indicates invalid device number)
 result = SBUS_XXX_COMMAND.ResultCode;
 if ( command == Write && result != 0x1 ) error;
 if ( command == Read && result != 0x4 ) error;

 // Clear register for next command
 SBUS_XXX_COMMAND = 0;

 // Read data
 data = SBUS_XXX_RESPONSE;
 return data;
}
```

Writes to command registers are ignored while a command is being executed. Reading any register returns its current value, however the SBUS_*XXX*_RESPONSE content is undefined while a READ command is executing.

The switch management bus supports multiple masters in the system (PCIe, EBI, FIBM, etc...) and a problem may arise if two masters try to read different SerDes registers. Here is an example:

Time	Master 1	Master 2
0	SBUS_*XXX*_COMMAND = read(A)	
1		SBUS_*XXX*_COMMAND = read(B)
2	x = SBUS_*XXX*_RESPONSE	
3		y = SBUS_*XXX*_RESPONSE

In this case, the command read(B) is ignored and the variable “y” contains the result of read(A), which is not the desired objective. This problem is a common for many registers in the switch and there is no logic in the switch to prevent this. To avoid this problem, only one master should issue a read or write command at a time and complete it before any other master issues a read command.

The SBUS_*XXX*_CFG register defines the reset state of the SBUS controller. The SBUS controller should be configured for a SBUS chain clock equal to 1/2 of the reference clock:

- • Reference clock is 100 MHz for PCIe. The SBUS should be configured for 50 MHz.
- • Reference clock is 156.25 MHz for EPL. The SBUS should be configured for 78.125 MHz.

9.3.1 SPICO Micro-Controller

The SBUS_EPL_CFG.*SBUS_ControllerReset* and SBUS_PCIE_CFG.*SBUS_ControllerReset* registers control the SPICO controller on each chain. Each SPICO controller can be in any of the following 3 states at any time:

Reset

SPICO controller is in reset and all internal circuits reset to the their default state. Software cannot be downloaded while the SPICO controller is in this state.

Disabled

SPICO controller is out of reset but the micro-processor is not running. Software may be downloaded while the SPICO is in this state.

Enabled

SPICO controller is out of reset and executing code.

9.3.2 SerDes Registers

The SerDes registers are documented in the register section. The operations for each type of SerDes, PCIe or Ethernet, are documented in the PCIe and Ethernet Port Logic chapter respectively.

9.3.3 Device Address to SerDes Map

Table 9-4 illustrates the SBUS to SerDes/EPL mapping as well as the SBUS ordering on the serial bus. Both the EPLs and the PCIe module have 4 SerDes each. The PCIe SBUS is the first one on the ring, therefore covers the SBUS address range 1 to 4, followed by EPL[1] SBUS, which covers the address range 5 to 9 and so on.

The list of devices and their respective individual and broadcast SBUS addresses for the PCIe and EPL chains are listed in Table 9-4 and Table 9-5, respectively. Do not use a broadcast address for a Read command that targets multiple receivers.

Table 9-4 PCIe SerDes Map

SerDes	SBUS Address	SBUS Address (PCS)	Broadcast Address	Broadcast Address (PCS)
RMON0	1	n/a	-	-
PCIE0.LANE[0]	2	3	0xFF	0xF7
PCIE0.LANE[1]	4	5	0xFF	0xF7
PCIE0.LANE[2]	6	7	0xFF	0xF7
PCIE0.LANE[3]	8	9	0xFF	0xF7
PCIE0.LANE[4]	10	11	0xFF	0xF7
PCIE0.LANE[5]	12	13	0xFF	0xF7
PCIE0.LANE[6]	14	15	0xFF	0xF7
PCIE0.LANE[7]	16	17	0xFF	0xF7
PCIE1.LANE[0]	18	19	0xFF	0xF7
PCIE1.LANE[1]	20	21	0xFF	0xF7
PCIE1.LANE[2]	22	23	0xFF	0xF7
PCIE1.LANE[3]	24	25	0xFF	0xF7
PCIE1.LANE[4]	26	27	0xFF	0xF7
PCIE1.LANE[5]	28	29	0xFF	0xF7
PCIE1.LANE[6]	30	31	0xFF	0xF7
PCIE1.LANE[7]	32	33	0xFF	0xF7
PCIE2.LANE[0]	34	35	0xFF	0xF7
PCIE2.LANE[1]	36	37	0xFF	0xF7
PCIE2.LANE[2]	38	39	0xFF	0xF7
PCIE2.LANE[3]	40	41	0xFF	0xF7
PCIE2.LANE[4]	42	43	0xFF	0xF7
PCIE2.LANE[5]	44	45	0xFF	0xF7
PCIE2.LANE[6]	46	47	0xFF	0xF7
PCIE2.LANE[7]	48	49	0xFF	0xF7
PCIE3.LANE[0]	50	51	0xFF	0xF7
PCIE3.LANE[1]	52	53	0xFF	0xF7
PCIE3.LANE[2]	54	55	0xFF	0xF7
PCIE3.LANE[3]	56	57	0xFF	0xF7
PCIE3.LANE[4]	58	59	0xFF	0xF7

Table 9-4 PCIe SerDes Map [Continued]

SerDes	SBUS Address	SBUS Address (PCS)	Broadcast Address	Broadcast Address (PCS)
PCIE3.LANE[5]	60	61	0xFF	0xF7
PCIE3.LANE[6]	62	63	0xFF	0xF7
PCIE3.LANE[7]	64	65	0xFF	0xF7
PCIE4.LANE[0]	66	67	0xFF	0xF7
RMON1	68	n/a	-	-
THERMAL SENSE	69	n/a	-	-
PMRO	70	n/a	-	-
SPICO	253	n/a	-	-
SBUS Controller	254	n/a	-	-

Table 9-5 Ethernet SerDes Map

SerDes	SBUS Address	Broadcast Address
RMON2	1	
EPL[0].LANE_A	2	0xFF
EPL[0].LANE_B	3	0xFF
EPL[0].LANE_C	4	0xFF
EPL[0].LANE_D	5	0xFF
EPL[1].LANE_A	6	0xFF
EPL[1].LANE_B	7	0xFF
EPL[1].LANE_C	8	0xFF
EPL[1].LANE_D	9	0xFF
EPL[2].LANE_A	10	0xFF
EPL[2].LANE_B	11	0xFF
EPL[2].LANE_C	12	0xFF
EPL[2].LANE_D	13	0xFF
EPL[3].LANE_A	14	0xFF
EPL[3].LANE_B	15	0xFF
EPL[3].LANE_C	16	0xFF
EPL[3].LANE_D	17	0xFF
EPL[4].LANE_A	18	0xFF
EPL[4].LANE_B	19	0xFF
EPL[4].LANE_C	20	0xFF
EPL[4].LANE_D	21	0xFF
EPL[5].LANE_A	22	0xFF
EPL[5].LANE_B	23	0xFF
EPL[5].LANE_C	24	0xFF
EPL[5].LANE_D	25	0xFF

Table 9-5 Ethernet SerDes Map [Continued]

SerDes	SBUS Address	Broadcast Address
EPL[6].LANE_A	26	0xFF
EPL[6].LANE_B	27	0xFF
EPL[6].LANE_C	28	0xFF
EPL[6].LANE_D	29	0xFF
EPL[7].LANE_A	30	0xFF
EPL[7].LANE_B	31	0xFF
EPL[7].LANE_C	32	0xFF
EPL[7].LANE_D	33	0xFF
EPL[8].LANE_A	34	0xFF
EPL[8].LANE_B	35	0xFF
EPL[8].LANE_C	36	0xFF
EPL[8].LANE_D	37	0xFF
RMON3	38	-
PMRO	39	-
SPICO	253	-
SBUS Controller	254	-

9.3.4 Firmware Download

The host software or SPI boot controller is required to download a firmware into each SerDes and the master SPICO controller before the SerDes can be used. Each firmware consists of a set of 10-bit words, and there are five firmware images to download:

- PCI-Express Master SPICO Firmware
- PCI-Express SerDes Firmware
- Ethernet Master SPICO Firmware
- Ethernet SerDes Firmware Swap File
- Ethernet SerDes Firmware

There are two methods to download the firmware images for the PCI-Express branch:

- via the LOAD instruction of the SPI Boot Controller.
- via software.

The Ethernet firmware images can only be downloaded by software.

9.3.4.1 Firmware Download by Software

The software method consists of first placing the target in a state to receive firmware, then downloading the firmware (3x10-bit words at a time) using the SBUS_*XXX* registers until all the firmware code is fully loaded, and then restarting the target.

The firmware download into SerDes uses a broadcast method to upload all SerDes at the same time.

Note: SerDes download can and does occur while the PCIE and EPL are in reset, and does not need to be repeated unless the device is fully reset.

The firmware download procedure for PCIe and Ethernet is very similar. The primary difference is that the Ethernet SerDes firmware cannot be downloaded all at once; instead, part of the Ethernet SerDes firmware is placed in a swap file that is loaded into the master SPICO memory. The master SPICO is responsible for pushing the swap file to the Ethernet SerDes at the appropriate time.

First, set the SBus to 1/3 of 100 MHz clock frequency (33 MHz) for PCIe, and 1/2 of 78.125 MHz (39.0625 MHz) for EPL).

```
// Set SBus frequency to refclk/3 for PCIe
sbus_pcie_command (0xFE, 0x0A, Write, 0x02);
sbus_pcie_command (0xFE, 0x0A, Write, 0x82);

// Set SBus frequency to refclk/2 for ethernet
sbus_epl_command (0xFE, 0x0A, Write, 0x01);
sbus_eth_command (0xFE, 0x0A, Write, 0x81);
```

The procedure to download firmware to the master SPICO is the following. For Ethernet, the SerDes swap file is appended to the end of the master SPICO firmware and downloaded as though a single file. As above, “xxx” represents either “epl” or “pcie”.

```
// Place Master SPICO into Reset and Enable off
sbus_xxx_command (0xFD, 0x01, Write, 0x0000_00C0);

// Remove Reset, Enable off, IMEM_CNTL_EN on
sbus_xxx_command (0xFD, 0x01, Write, 0x0000_0240);

// Set starting IMEM address for burst download
sbus_xxx_command (0xFD, 0x03, Write, 0x8000_0000);

// Write firmware image to Master SPICO
// data_word[0..n-1] = N x 10b of data.
for (i=0;i<n;i++)
    if ( n-i >= 3 )
        {num_writes = 3;}
    else if ( n-i >= 2 )
        {num_writes = 2;}
    else
        {num_writes = 1;}

    // Set [31:30] to number of words,
    // Set [29:20] equal to 3rd data_word,
    // Set [19:10] equal to 2nd data_word,
    // Set [9:0] equal to 1st data word
    sbus_xxx_command (0xFD, 0x14, Write, (num_writes << 30 |
                                         (data_word[i+2]& 0x3FF << 20) |
                                         (data_word[i+1]& 0x3FF << 10) |
                                         (data_word[i] & 0x3FF));
    i += 3;
}

// Start the Master SPICO controller
sbus_xxx_command (0xFD, 0x01, Write, 0x0000_0040); // Set IMEM_CNTL_EN off
sbus_xxx_command (0xFD, 0x16, Write, 0x000C_0000); // Turn ECC on
sbus_xxx_command (0xFD, 0x01, Write, 0x0000_0140); // Set SPICO_ENABLE on
```

The procedure to download firmware to all the SerDes is the following. This procedure uses the 0xFF broadcast address to all SerDes in the SBUS ring.

```
// Place SerDes in Reset & disable SPICO
sbus_xxx_command (0xFF, 0x07, Write, 0x0000_0011);

// Remove SerDes Reset
sbus_xxx_command (0xFF, 0x07, Write, 0x0000_0010);

// Assert IMEM override
sbus_xxx_command (0xFF, 0x00, Write, 0x4000_0000);

// Write firmware image to SerDes
// data_word[0..n-1] = N x 10b of data.
for (i=0;i<n;i++)
    if ( n-i >= 3 )
        {num_writes = 3;}
    else if ( n-i >= 2 )
        {num_writes = 2;}
    else
        {num_writes = 1;}

    // Set [31:30] to number of words,
    // Set [29:20] equal to 3rd data_word,
    // Set [19:10] equal to 2nd data_word,
    // Set [9:0] equal to 1st data word
    sbus_xxx_command (0xFF, 0x0A, Write, (num_writes << 30 |
                                          (data_word[i+2]& 0x3FF << 20) |
                                          (data_word[i+1]& 0x3FF << 10) |
                                          (data_word[i] & 0x3FF));
    i += 3;
}

// Start SerDes controller
sbus_xxx_command (0xFF, 0x00, Write, 0x0000_0000); // IMEM override off
sbus_xxx_command (0xFF, 0x0B, Write, 0x000C_0000); // Turn ECC on
sbus_xxx_command (0xFF, 0x07, Write, 0x0000_0002); // Turn SPICO Enable on
sbus_xxx_command (0xFF, 0x08, Write, 0x0000_0000); // Enable core and hardware interrupts
```

9.3.4.2 Firmware Download by BSM LOAD Command

This method is only available for PCI Express firmware, and only from the Boot State Machine (BSM). It should be used to reduce the delay of PCIe bring-up. The method is the same as for software download, except that the loop over all data words of the image is replaced by a single BSM LOAD instruction (refer to the BSM Instruction format in Section 9.7.5.3 for more details). The data format used in the payload of the LOAD command is the same as in the software data loop above (3 x 10-bit firmware words per 32 bits).

For the PCIe Master SPICO, the LOAD instruction uses the special address 0x000000, with offset selector 0 and address increment 0. For broadcast to the PCIe SerDes, the LOAD instruction uses address 0x000001, again with offset selector 0 and address increment 0.

After the LOAD command completes, there must be a delay of 10us before any subsequent `sbus_pcie_command()` or any other write to the SBUS_PCIE_COMMAND register.

9.3.4.3 Firmware Verification

A CRC check may be performed on the downloaded firmware, and the revision number of the firmware may be checked. This ensures that the download completed without error.

```
// Check the firmware downloaded is correct
sbus_xxx_check_crc();
sbus_xxx_check_version(<master firmware version #>, <serdes firmware version #>);
```

The `sbus_xxx_check_crc()` and `sbus_xxx_check_version()` operations are performed by interrupts issued to the master SPICO and SerDes. Interrupts can be issued via the SBus chain or the core interfaces over management. Since the SBus interrupts have more options and PCIe SerDes do not have their core interrupt interface connected to management, all interrupts are performed over the SBus.

The interrupts to perform a CRC check on the downloaded firmware take a significant amount of time, and can be performed in parallel across the master SPICO and all the SerDes.

```
sbus_xxx_check_crc()
{
    // Execute Master SPICO CRC check interrupt
    sbus_xxx_command (0xFD, 0x02, Write, 0x0000_0002);
    sbus_xxx_command (0xFD, 0x07, Write, 0x0000_0001);

    // Execute SerDes CRC check interrupts (broadcast)
    sbus_xxx_command (0xFF, 0x03, Write, 0x003C_0000);

    // wait (MicroSeconds);

    // Poll Master SPICO for completion (only check bit[15])
    poll ((result = sbus_xxx_command (0xFD, 0x08, Read, 0x0000_0000))[15] == 0 );

    // Check Master SPICO result
    if (result[31:16] != 0x0001) fail; // (0x0001 is pass, 0xFFFF is fail)

    for each serdes
    {
        // Poll SerDes for completion (only check bit[16])
        poll ((result = sbus_xxx_command (serdes, 0x04, Read, 0x0000_0000))[16] == 0 );

        // Check SerDes result
        if (result[15:0] != 0x00) fail; // (0x00 is pass, 0xFF is fail)
    )
}

sbus_xxx_check_version(master_version, serdes_version)
{
    // Execute Master SPICO Revision check interrupt
    sbus_xxx_command (0xFD, 0x02, Write, 0x0000_0000);
    sbus_xxx_command (0xFD, 0x07, Write, 0x0000_0001);

    // Poll Master SPICO for completion (only check bit[15])
    poll ((result = sbus_xxx_command (0xFD, 0x08, Read, 0x0000_0000))[15] == 0 );

    // Check result
    if (result[31:16] != master_version) fail;

    // Execute SerDes Revision check interrupts (broadcast)
    sbus_xxx_command (0xFF, 0x03, Write, 0x0000_0000);

    for each serdes
    {
        // Poll SerDes for completion (only check bit[16])
        poll ((result = sbus_xxx_command (serdes, 0x04, Read, 0x0000_0000))[16] == 0 );

        // Check SerDes result
        if (result[15:0] != serdes_version) fail;
    )
}
```

9.3.5 PCIe SerDes Initialization

After firmware has been downloaded to the PCIe SerDes, and they have been brought out of reset, a series of interrupts are broadcast to all PCIe SerDes to initialize their refclk frequency and equalization.

```
// Initialize PCIe SerDes, in parallel
sbus_pcie_command (0xFF, 0x03, Write, 0x0006_0050); // RX refclk ratio
sbus_pcie_command (0xFF, 0x03, Write, 0x0005_1032); // TX refclk ratio
sbus_pcie_command (0xFF, 0x03, Write, 0x0026_000E); // seed CTLE equalization parameter HF
sbus_pcie_command (0xFF, 0x03, Write, 0x0026_0102); // seed CTLE equalization parameter LF
sbus_pcie_command (0xFF, 0x03, Write, 0x0026_0238); // seed CTLE equalization parameter DC
sbus_pcie_command (0xFF, 0x03, Write, 0x0026_5202); // run iCal on second equalization
sbus_pcie_command (0xFF, 0x03, Write, 0x0026_5B01); // disable pCal
sbus_pcie_command (0xF7, 0x0F, Write, 0x2204_1604); // calibrate Gen1/Gen2 amplitude
sbus_pcie_command (0xF7, 0x12, Write, 0x0000_9604); // apply amplitude calibration on power up
sbus_pcie_command (0xFF, 0x03, Write, 0x002B_0000); // set RX termination to AGND
```

The PCIe SerDes interrupt interface for individual SerDes lanes may also be accessed via the PCIE_SERDES_CTRL register.

9.3.6 Temperature and Voltage Sensing

The device support 9 temperature sensing points and 8 voltage sensors.Those are all addressed at address 69 on the PCIe SBUS chain. The code to read them all is the following:

```
/* Start sensing */
sbus_pcie_command (69, 0, Write, 3);
v = sbus_pcie_command (69, 0, Read, 0);
sbus_pcie_command (69, 1, Write, 25);

/* Read all 17 sensors */
foreach i in (1,2,3,4,5,6,7,8,11,12,13,14,15,16)
{
    sbus_pcie_command (69, 3, Write, 1<<i);
    sbus_pcie_command (69, 0, Write, 2);
    wait (1.0us);

    /* Read sensor; bit 15 indicate success, failure */
       v = sbus_pcie_command (69, 65+i, Read, 0);

    /* Save data */
    if ( i < 9 )
    {
       /* First 9 are temperatures sensors */
       if ( v & 0x8000 ) temperature[i-1] = (v & 0x7FFF)/8;
       else temperature[i-1] = 0;
    }
    else
    {
       /* Next 8 are voltage sensors */
       if ( v & 0x8000 ) voltage_sense[i-11] = (v & 0xFFF)/2;
       else voltage_sense[i-11] = 0;
    }
    sbus_pcie_command (69, 0, Write, 2);
}
```

The temperature sensors are precise at +/- 5 °C. The voltage sensors are precise +/- 10 mV.

The locations are:

- TEMPERATURE[0]: PCI Host Interface #0
- TEMPERATURE[1]: Switch temperature 0

- TEMPERATURE[2]: Switch temperature 1
- TEMPERATURE[3]: Switch temperature 2
- TEMPERATURE[4]: Switch temperature 3
- TEMPERATURE[5]: Switch temperature 4
- TEMPERATURE[6]: Ethernet Port Logic #8
- TEMPERATURE[7]: Tunneling engine

The voltage sensing points are:

- VOLTAGE_SENSE[2]: Switch sense 0
- VOLTAGE_SENSE[3]: Switch sense 1
- VOLTAGE_SENSE[4]: Switch sense 2
- VOLTAGE_SENSE[5]: Switch sense 3
- VOLTAGE_SENSE[6]: Switch sense 4
- VOLTAGE_SENSE[7]: Switch sense 5

The “switch” sensing points are going to provided the most extreme read out under load. To reduce access time, it is suggest to only poll the temperature sensor #2.

9.4 I^{2}C Controller

The FM10000 contains one I^{2}C controller. The I^{2}C controller supports the following features:

- 100 KHz or 400 KHz operation.
  - The speed is programmable through the I2C_CFG register and the switch supports a larger set.
  - It is recommended to use 400 KHz as the default for I^{2}C operation.
- Slave mode allowing external masters to read/write registers in the chip.
  - All registers are accessible.
  - Configurable I^{2}C slave address.
- Master mode allowing the switch to access external I^{2}C devices.
- 7-bit addressing mode.
- Partial SMB compatibility.

The I^{2}C controller does not support the following features found in some I^{2}C devices:

- General call address.
- 10-bit addressing mode.
- Start byte.
- Mix-speed mode.
- High-speed mode.
- Arbitration

The I^{2}C controller doesn't support the following SMB features:

- 35 ms time out, the time out is hard-coded to 100 ms.
- PEC

Figure 9-3 shows the basic read and write accesses.

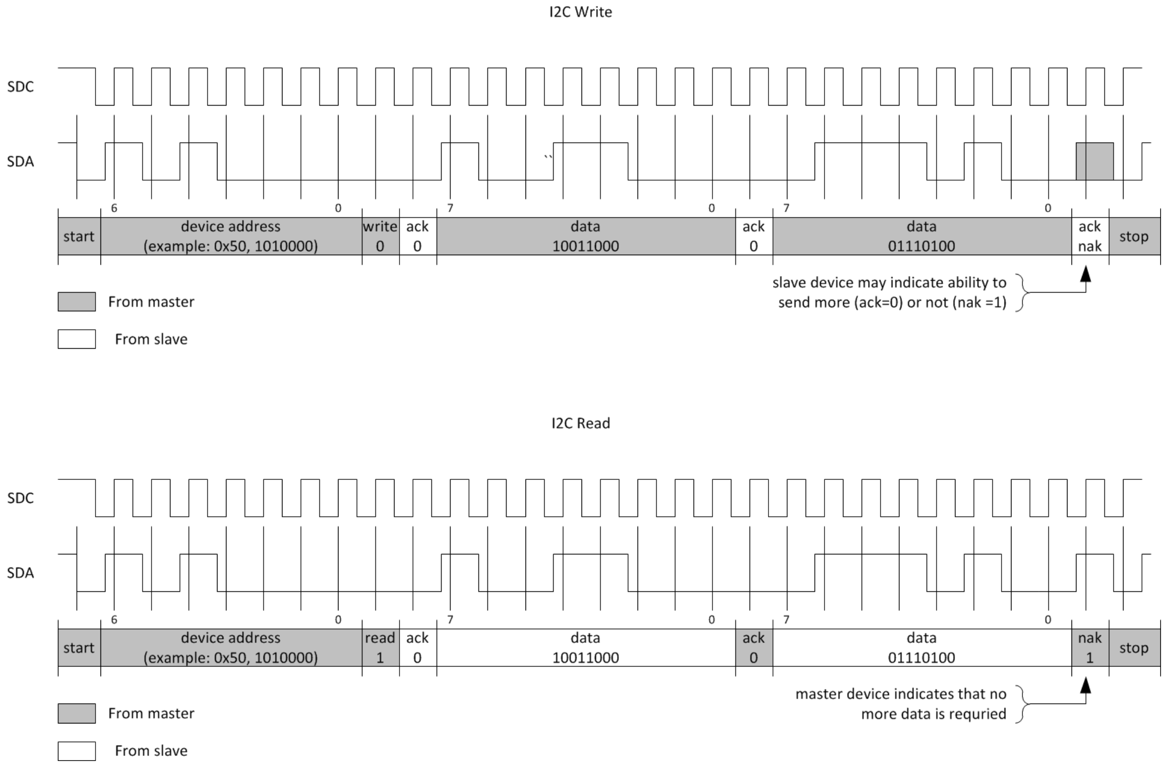


Figure 9-3 Basic I^{2}C Read/Write Accesses

The timing diagram is shown in Figure 9-4.

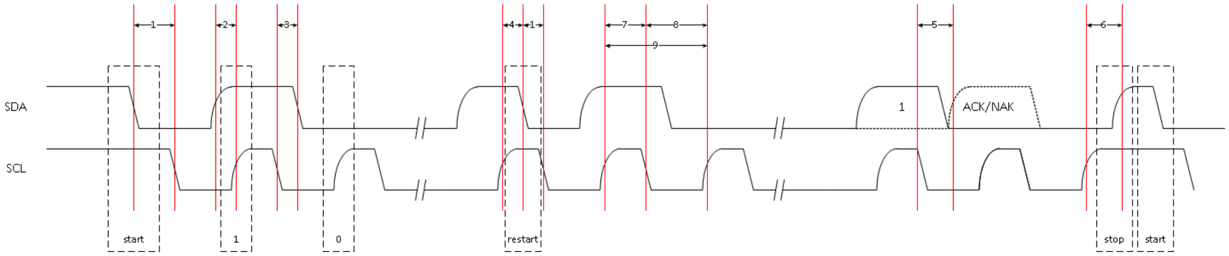


Figure 9-4 I^{2}C Timing Diagram

Table 9-6 I^{2}C Timing Constraints

Reference	Name	Description	100 KHz	400 KHz
1	tHD;STA	Restart/restart setup.	≥ 4.0 µs	≥ 0.6 µs
2	tSU;DAT	Data setup to clock high.	≥ 250 ns	≥ 100 ns
3	tHD;DAT	Data hold from clock low.	>0	>0
4	tSU;STA	Clock high to Restart.	≥ 4.7 µs	≥ 0.6 µs
5	tVD;DAT tVD;ACK	Data/ack response from slave.	≤ 3.45 µs	≤ 0.9 µs
6	tSU;STO	Stop setup.	≥ 4.0 µs	≥ 0.6 µs
7	tHIGH	Clock high.	≥ 4.0 µs	≥ 0.6 µs
8	tLOW	Clock low.	≥ 4.7 µs	≥ 1.3 µs
9	fSCL	Frequency.	≤ 100 KHz	≤ 400 KHz

The pins SDA and SDC are open drain pins with external pull-ups. This structure has the disadvantage of creating slow rising edges which can potentially be incorrectly interpreted as a double transition. To prevent this, the FM10000 device incorporates a digital filtering circuit to ensure that only complete transitions of either SDC or SDA are detected. The filter operates by sampling the SDA and SDC using the LSM default clock rate and only reports transitions that are stable for at least N cycles, where N is defined in the I2C_CFG.DebounceFilterCountLimit register.

The I^{2}C controller supports a timeout of 100 µs (1/10 of the I^{2}C clock rate) that aborts the current cycle and returns all pins to 1.

The I^{2}C register is used to configure the controller:

- I2C_CFG:
  - — *Enable* (1 bit) — Defines if the switch will answer to an I^{2}C address from an external master or not.
  - — *Address* (7 bits) — Defines the address to which the switch will answer.
  - — *Divider* (12 bits) — Defines clock rate as a divider of the PCIE_REFCLK base clock.
    - Use 52 decimal (0x34h) for 100 KHz.
    - Use 10 decimal (0xAh) for 400 KHz.
    - Warning: Default divider is 100 decimal (0x64h) and results in slower than normal conventional operating clock frequency.
  - — *InterruptMask* (1 bit) — Defines the interrupt masking (used in Master mode only).
  - — *DebounceFilterCountLimit* (7 bits) — Defines the number of clock for data to be stable before being recognized. This register is used to filter glitches on I^{2}C with bad slew rate.

9.4.1 Slave Mode

An external agent on the I^2C bus can access any register of the chip as the CPU does; this is the slave mode.

The slave mode has the following characteristics:

- Slave address is user configurable and defaults to 'x100 000x'.
- All write accesses start with a 3-byte address field to indicate which address (register or table) to access followed by N x 4-byte data words. Address and data must be sent MSB first. Data words are written into consecutive addresses starting with the address given. The address is incremented at the end of the data transfer. The master is at liberty to write any number of words and it is assumed that the master never attempts to write into an illegal address. It is possible for a write access to only include the 3-0 byte address without any data words. This is used to load an address into the chip for an eventual read (note that the address is saved only if 3 address bytes are sent).
- All read accesses return consecutive 4-byte words starting with the address currently latched. The actual register is latched just before the first byte of the word is sent. The master is at liberty to read any number of words and terminates with a NAK on the last byte desired, which could any byte.
- The controller supports restart cycles (a stop-start).

This is shown in Figure 9-5.

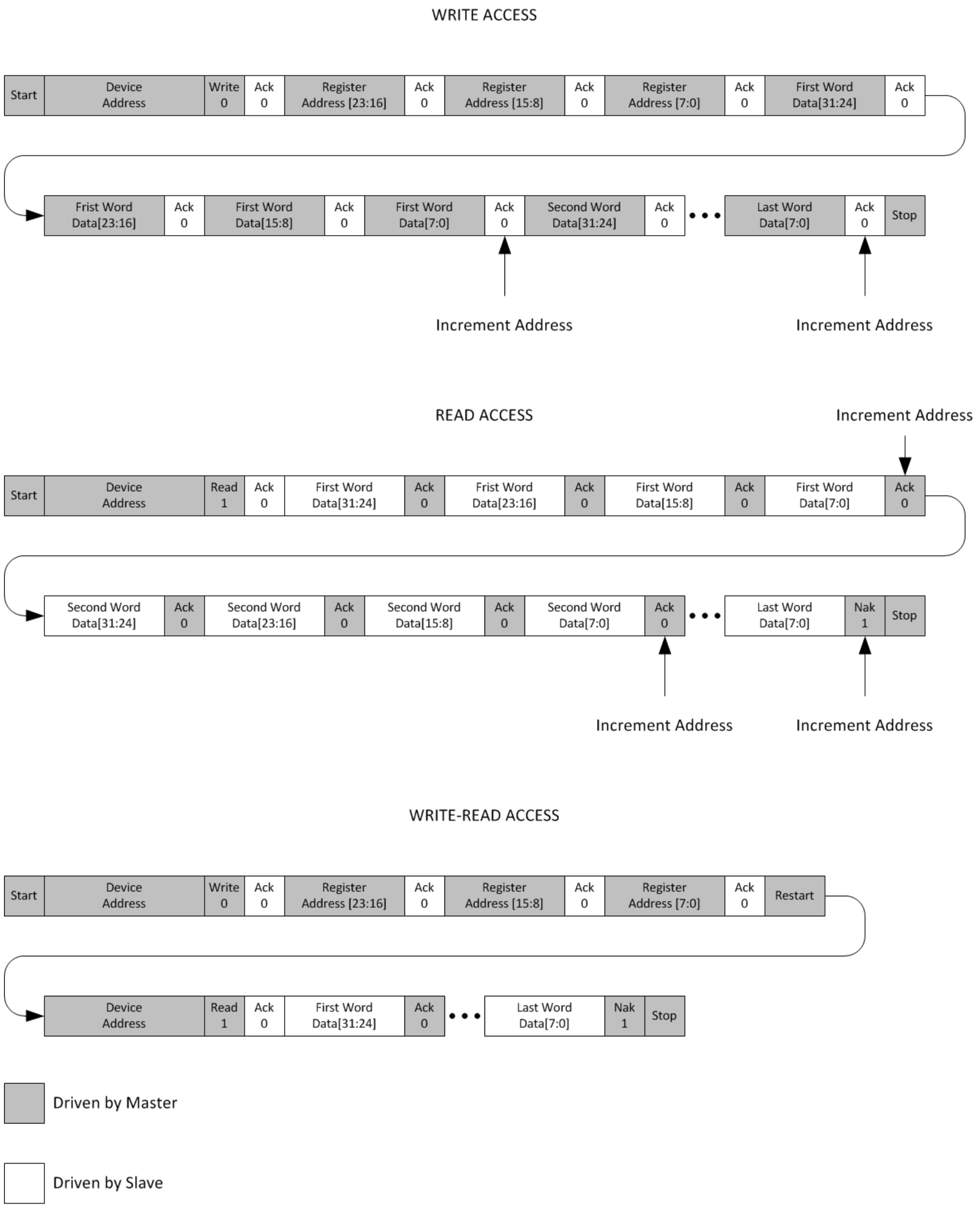


Figure 9-5 I^{2}C Slave Access

SMB hosts can use this interface to read/write registers provided the following methods are used.

- For writing a single register, the SMB host shall issue SMBus write with 3-byte address first followed 4-byte data second.
  - — The SMB requires a “command” field which is be set to the MSB of the address, followed by next 2 bytes of address and 4 bytes of data

- For reading a single register, host shall proceed in two steps:
    1. Host issues a 3-byte SMBus write command, where command is the MSB of the address and followed by 2 bytes of addresses. This is to latch the register address to read.
    2. Host issues a 4-byte SMBus read command, where command is a dummy byte, followed by 4-byte read. The single dummy byte is ignored by the switch.

9.4.2 Master Mode

A host attached to PCIe can issue I^2C transactions on the I^2C interface to reach devices on this bus, this is the master mode. In this mode the I^2C controller is capable of issuing automatic I^2C accesses:

- **write** — Can write up to 12 bytes per transaction.
- **write-read** — Can write up to 12 bytes and receive up to 12 bytes.
- **read** — Can read up to 12 bytes.

Note: A 'write' with a length of 0 can be used as a peek to determine if a device is present at the slave address requested.

The I^2C registers used are:

- I2C_DATA[0..2] (12 bytes) — Contains the data for 'write' or 'write-read' commands or 'read' commands. The register contains the data to write before the command is executed (if write command is executed) and contains the data from read after the command is executer (if read command executed).
- I2C_CTRL — Used when I^2C controller is master:
    — *Addr* (8 bits) — Defines the address of the device to access (lower bit ignored).

    — *Command* (2 bits) — Execute command 'write', 'write-read', 'read', 'null'.

    — *LengthW* (4 bits) — Defines the number of bytes to send (0..12). Minimum is 0. Hardware behavior for length greater than 12 is undefined.

    — *LengthR* (4 bits) — Defines the number of bytes to read (1..12). Hardware behavior for length of 0 or greater than 12 is undefined.

    — *LengthSent* (4 bits) — Indicates the number of bytes actually written (0..LengthW). This field is shorter than LengthW if and only if a NAK is received prematurely.

    — *CommandCompleted* (4 bits) — Indicates the status of the command (running,completed, aborted, no_device, etc...).

    — *InterruptPending* (1 bit) — Indicates if an interrupt is pending. Automatically set when the command completes. Writing 1b clears the interrupt.

    The possible transactions are shown in Figure 9-6.

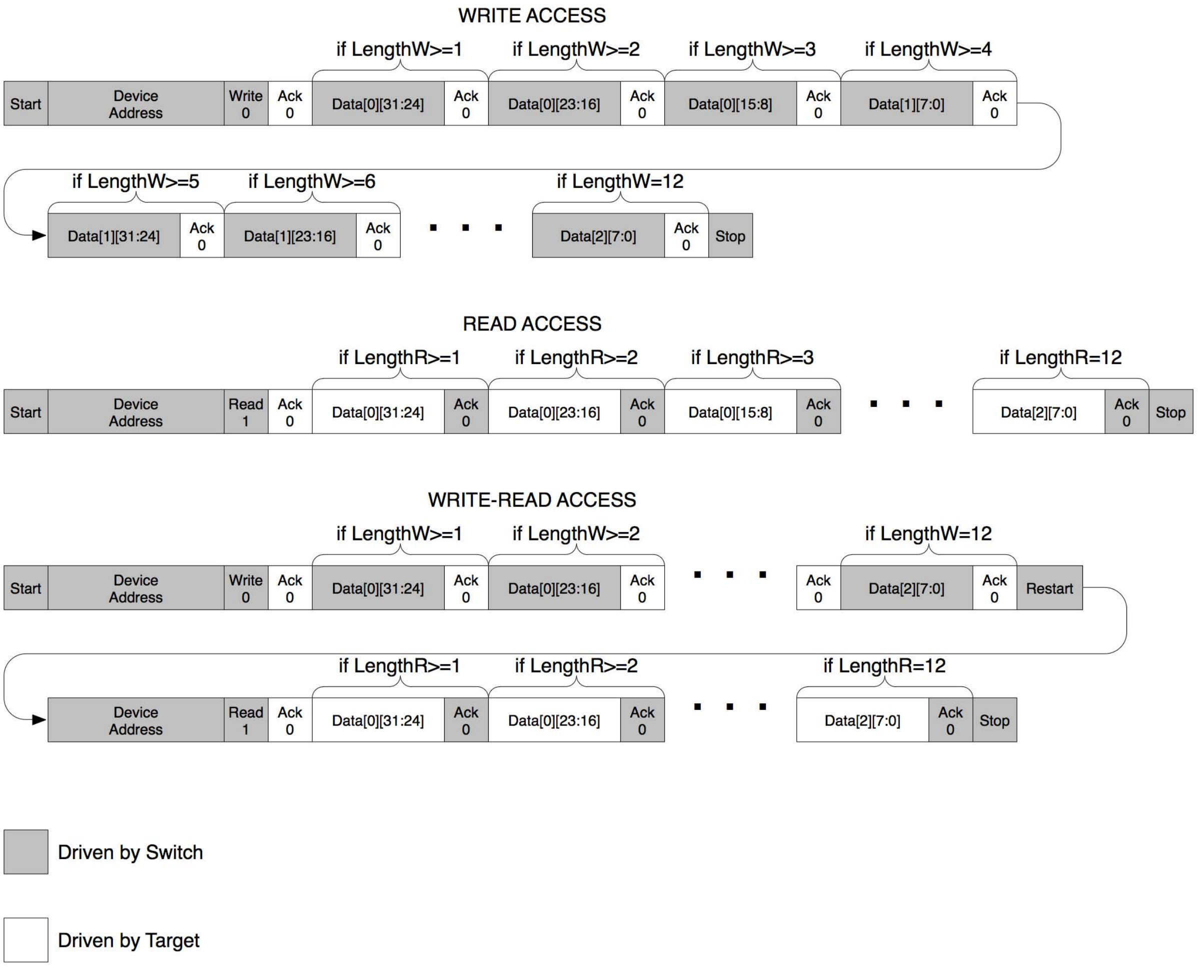


Figure 9-6 I^{2}C Multiple Accesses

9.5 MDIO Controller

The FM10000 supports one MDIO controller. The MDIO controller is designed to allow the CPU to access MDIO devices through the switch. The features are:

- Support for clauses 22 and 45.
- Master only. The switch cannot be the target for any access.
- 3.3V/2.5V level compatibility only. An external level converter is required for access to 1.2V MDIO as defined in IEEE802.3ae specification.

The clause 22 and clause 45 timing diagrams are shown in Figure 9-7 and Figure 9-8, respectively.

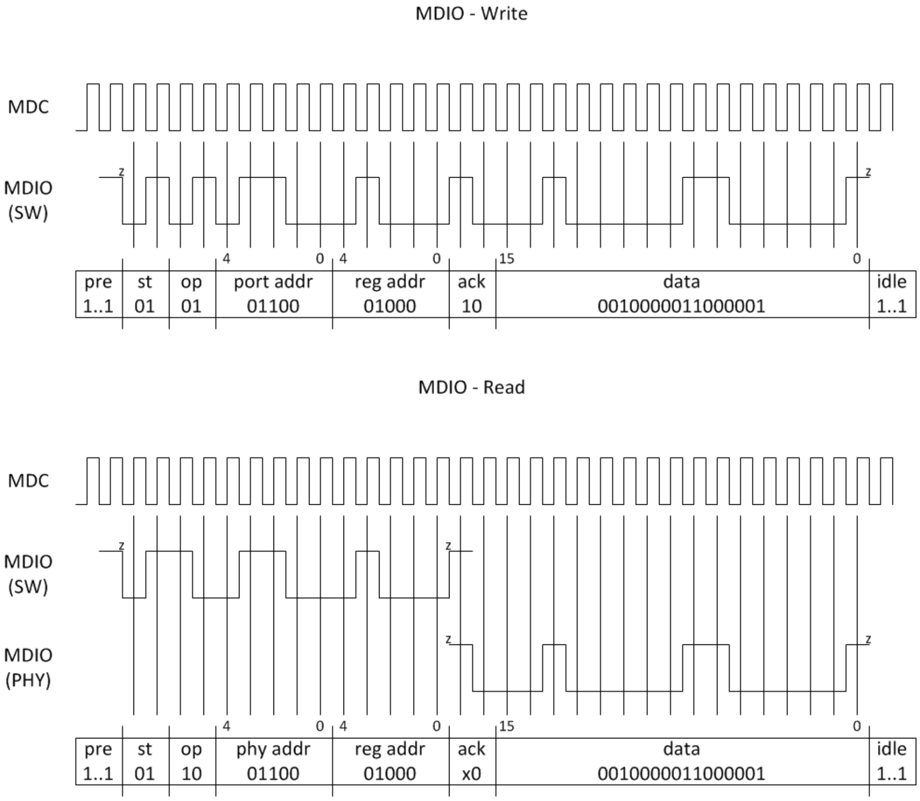


Figure 9-7 MDIO Clause 22 Timing Diagram

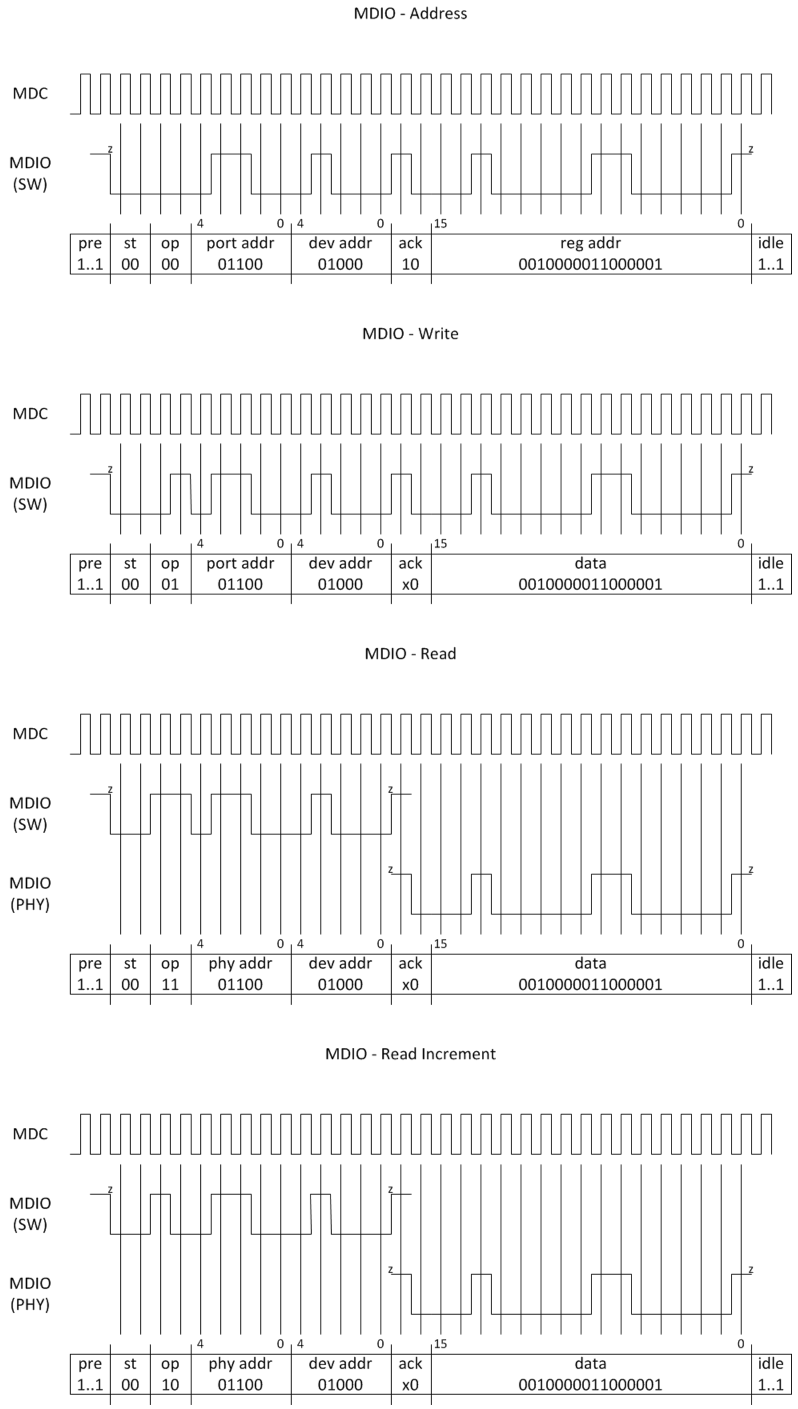


Figure 9-8 MDIO Clause 45 Timing Diagram

The register settings that support these transactions are the following:

- MDIO_CFG:
  - — *Divider* (12 bits) — Defines the clock divider (from PCIE_REFCLK clock)
  - — *Preamble* (1 bit) — Defines if the 32-bit pre-amble is always sent or not.
  - — *InterruptMask* (1 bit) — Defines if the MDIO interrupt is masked or RW passed (0) to the interrupt hierarchy.
- MDIO_DATA — The data sent or read. Only 16 bits.
- MDIO_CTRL:
  - — *Register* (16 bits)
    - Clause 22: Not used.
  - — *Device* (5 bits) — Sub-device to address.
  - — *PHYAddress* (5 bits) — Physical device to address.
    - Clause 45: Device Address (5 bits)
    - Clause 22: Re-purposed as the "register" field.
  - — *Command* (2 bits)
    - 00b = Null — Do nothing.
    - 01b = Write
      - — Clause 45: Send register address frame and then write frame.
      - — Clause 22: Send write frame.
    - 10b = Read
      - — Clause 45: Sequential read, send read command frame only.
      - — Clause 22: Send read command frame only.
    - 11b = Random-read
      - — Clause 45: Send register address frame followed by a read frame.
      - — Clause 22: Invalid.
  - — *DeviceType* (1 bit) — Defines if the frame format is compatible with clause 22 (0b) or clause 45 (1b).
  - — *Status* (2 bits) — Returns the status of the command.
  - — *InterruptPending* (1 bit) — Indicates if a command has been completed. Writing '1' clears the interrupt.

9.6 GPIO Controller

The General Purpose IO (GPIO) controller is 16-bits wide and supports:

- Input, output, open-drain.
- Interrupts per bit
  - Configurable for low to high or high to low or both.

The following registers are defined for the controller:

- GPIO_DATA
- GPIO_CTRL
- GPIO_IP
- GPIO_IM

All GPIO pins are defaulted as inputs after reset.

9.7 SPI Interface

9.7.1 Overview

The SPI interface is a serial interface which can be used by the switch to upload a default configuration after reset, or to load commands to handle certain other conditions. The SPI interface also offers a management interface allowing a CPU, local or remote, to initiate transactions on the SPI bus. The SPI interface supports normal read mode (single pin), a dual-pin mode and a quad-pin mode. Speed can be up to 50 MHz (1/2 of 100 MHz reference clock). The default speed is 1 MHz.

The list of signals is defined in Table 9-7.

Table 9-7 SPI Signals

Pin	Direction	Description
SPI_CS_N	OUT	SPI Chip Select (Active Low)
SPI_CLK	OUT	CLOCK for SPI interface. Maximum is 50 MHz.
SPI_MOSI/IO0	OUT	Serial Data Output (MOSI, Master-Out- Slave-In, since the FM10000 is master). Connect to serial data input of serial EEPROM/FLASH. Also used as a serial data input (IO0) when operating in dual-pin mode.
SPI_MISO/IO1	IN	Serial Data Input (MISO, Master-In-Slave-Out, since the FM10000 is master). Connect to serial data output of serial EEPROM/FLASH. Also used as a second serial data input (IO1) when operating in dual-pin mode.
SPI_IO2	I/O	Serial Data IO2, used as a third serial data input in quad-pin mode or general purpose pin when not.
SPI_IO3	I/O	Serial Data IO3, used as a fourth serial data input in quad-pin mode or general purpose pin when not.
SPI_BOOT	IN	Defines if booting from SPI is enabled (asserted) or disabled (de-asserted).

9.7.2 Requirements

An SPI EEPROM or Flash must be connected to each FM10000 and provide boot code for that FM10000. Boot should then continue over PCIe.

The FM10000 Boot Image is designed to fit on a 1 MB (8 Mb) memory device. The device must therefore be at least 1 MB in size. All addresses beyond 1 MB are unused.

The SPI Memory device must be erasable in chunks (either sector or page) of 4 KB or smaller to support serial number sector updates or the double banking update procedure.

The SPI Memory device must be addressable in binary power of two pages (such as 256-byte or 512-byte). The reason is that the SPI Boot Image contains pointers to locations into the SPI memory device. The behavior is not defined for non-power of two pages. For example, 512-byte pages are legal while 528-byte pages are illegal for interfacing with the FM10000.

To support updates of the SPI Memory device contents through the FM10000 SPI Master, the write protection needs to be disabled. For devices with a write protect pin, this pin should either be connected to one of the FM10000 GPIOs or forced into a disabled state.

Notes:

1. The FM10000 uses the read command, 0x3 and 3-byte addressing when reading the boot image. The chosen Flash must support this.
2. The FM10000 uses 2.5 V LVCMOS signaling. The chosen Flash must either be compatible with these levels or use a translator.

9.7.3 Protocol

The SPI configuration upload at boot time is a two-step process:

1. Read at location 0x000000 to retrieve the options (mode and clock divider) and the address of the boot image. This is commonly running at lower speed using the most commonly-supported read command (0x03). The user may program the next mode and clock divider desired in the first word retrieved to accelerate future download.
2. Read and execute boot image at the location recovered in step 1 at the mode and speed specified. If there is a jump taken in the instruction stream. Step 2 is re-executed until the program terminates. Refer to Section 9.7.5 for details about image format.

Figure 9-9 shows the first step.

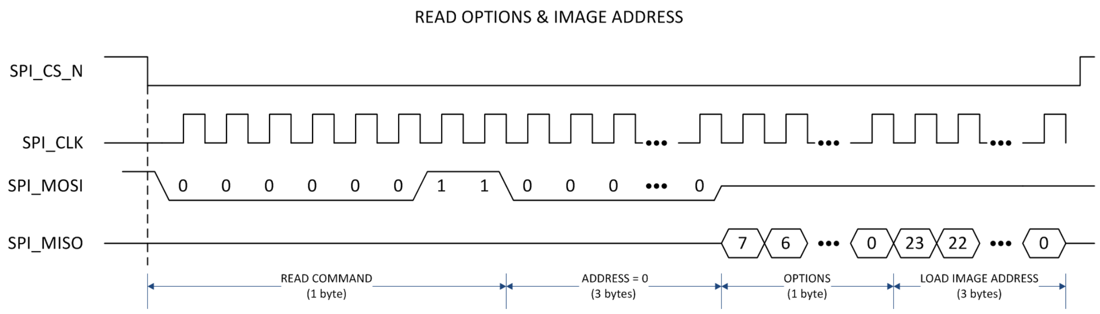


Figure 9-9 Serial Configuration Upload - First Access

The 8-bit option field is encoded as follows:

Bit(s)	Description
2:0	DON'T CARE
5:3	SPEED 000b = 0.39 MHz 001b = 0.78 MHz 010b = 1.56 MHz 011b = 3.125 MHz 100b = 6.25 MHz 101b = 12.5 MHz 110b = 25.0 MHz 111b = 50.0 MHz
7:6	MODE 00b = single (command code = 0x03) 01b = dual (command code = 0x3B) 10b = quad (command code = 0x6B) 11b = single_fast (command code = 0x0B)

Figure 9-10 shows the transactions under the four different modes supported.

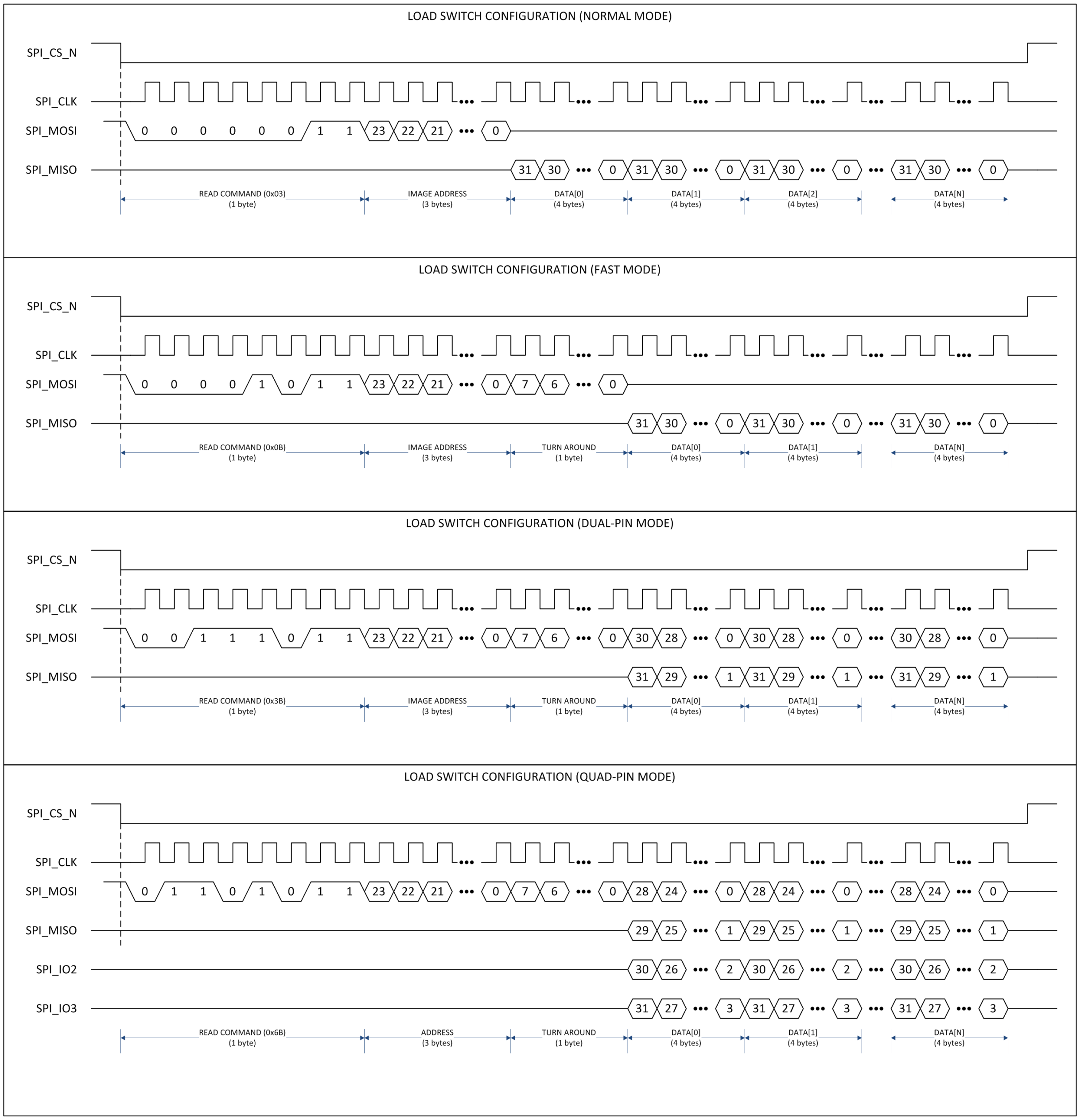


Figure 9-10 Serial Configuration Upload - Image Read

The process is:

1. Switch is reset, SPI_BOOT is asserted.
2. Switch asserts SPI_CS_N and send a read command (0x03) at address 0.
    a. Data is driven on the negative edge the clock.
    b. Most significant bit is sent first.
    c. Rate is 1/256th of reference clock (390 KHz).
3. Switch reads 32-bit word to recover base address of image selected.
    a. First 8 bits defines speed (bit 5-3) and mode (7:6).
    b. Next 24 bits defines address of load image.
4. Switch de-assert and re-assert SPI_CS_N and send a read command at address recovered previously.
    a. Command is 0x03 for single-pin read, 0x0B for fast-read, 0x3B for dual and 0x6B for quad.
    b. Send address to read from.
    c. Send a dummy byte (turn around byte) for commands 0x0B, 0x3B and 0x6B.
5. Switch reads data words (32 bits at the time) until end of configuration or jump.
    a. Data is sampled on the positive edge of the clock.
6. De-activate SPI_CS_N, SPI_CLK.
7. Terminate if program is terminated, or restarting fetching if it is a jump.

9.7.4 Management

The SPI controller also include capability for the host (local or remote) to issue commands on the SPI bus. The SPI_CTRL register controls the interface:

- *ENABLE* — Indicates if the controller is enabled or not.
- *FREQ* — Define speed of operation.
- *CMD*[3:0] — Command to execute.
- *SHIFT_METHOD* — Single, dual or quad.
- *DATA_SIZE* — 1,2,3,4 bytes (4 bytes coded as '00').
- *HEADER_SIZE* — 1,2,3,4 bytes (4 bytes coded as '00').

The register SPI_HEADER contains the header, the register SPI_TX_DATA contains the data to shift out, the SPI_RX_DATA contains the data shift in. The shift method is only applicable for data read.

The command is a 4-bit structure:

Bit 0 = Indicates if a header must be shifted out.

Bit 1 = Indicates if an 8-bit idle must be shifted out (used for turn-around in read-fast, read-dual, read-quad modes).

Bit 2 = Indicates if data must be shifted in/out.

Bit 3 = Indicates if the SPI_CS_N must be de-asserted or left asserted (set to 0b to leave asserted).

Figure 9-11 shows a management request with all four steps enabled, each step can potentially be disabled.The command must be cleared (all 4 bits set to 0b) before a new command can be issued.

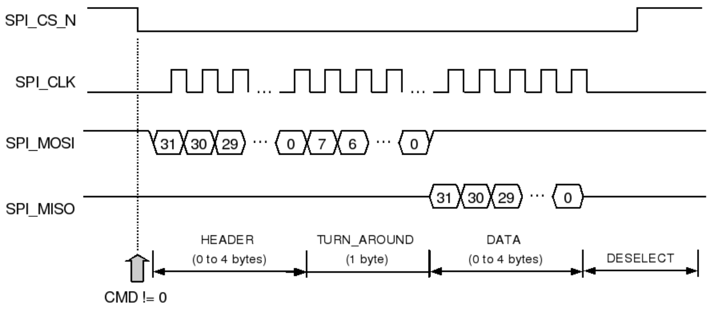


Figure 9-11 SPI Timing Diagram

As an example, to read at a random address in an SPI-based EEPROM, the instruction to send is 0x03 followed by the address to read (total 32 bits) and followed by 32 bits of data read. This would be executed as follows:

```
SPI_HEADER = 0x03000000 + <addr>;       # Set command (high byte) and address
SPI_TX_DATA = <don't care>;             # Set command (high byte) and address
SPI_CTRL.Enable = 1;                    # Enable controller
SPI_CTRL.Freq = 0;                      # Max rate (50MHZ)
SPI_CTRL.ShiftMethod = 0;               # Single pin mode
SPI_CTRL.HeaderSize = 0;                # 4 bytes header
SPI_CTRL.DataSize = 0;                  # 4 bytes data size
SPI_CTRL.Cmd = 4'b1101;                 # Send Header, Get Data, Deselect when done
---wait 1.28us--- (64b/50MHZ)
data = SPI_RX_DATA;                     # Data read
SPI_CTRL.Cmd = 4'b0000;                 # Clear command
```

To read sequence of 4 words in quad-bit mode without de-selecting between words:

```
SPI_HEADER = 0x6B000000 + <addr>; # Set command and address
SPI_CTRL.Enable = 1;              # Enable controller
SPI_CTRL.Freq = 0;                # Max rate
SPI_CTRL.ShiftMethod = 2;         # Quad pin mode
SPI_CTRL.HeaderSize = 0;          # 4 bytes header
SPI_CTRL.DataSize = 0;            # 4 bytes data size
SPI_CTRL.Cmd = 4'b0111;            # Send Header, Idle 1 byte, Get Data, Keep selected when done
---wait 0.96us--- ((32+8+32/4)/50MHZ)
data = SPI_RX_DATA;               # Data word 0
SPI_CTRL.Cmd = 4'b0000;           # Clear command
SPI_CTRL.Cmd = 4'b0100;           # Get more data
---wait 0.16us--- ((32/4)/50MHZ)
data = SPI_RX_DATA;               # Data word 1
SPI_CTRL.Cmd = 4'b0000;           # Clear command
SPI_CTRL.Cmd = 4'b0100;           # Get more data
---wait 0.16us--- ((32/4)/50MHZ)
data = SPI_RX_DATA;               # Data word 2
SPI_CTRL.Cmd = 4'b0000;           # Clear command
SPI_CTRL.Cmd = 4'b1100;           # Get more data, deselect when done
---wait 0.16us--- ((32/4)/50MHZ)
data = SPI_RX_DATA                # Data word 3
SPI_CTRL.Cmd = 4'b0000;           # Clear command
```

Figure 9-12 shows the timing diagram for this sequence.

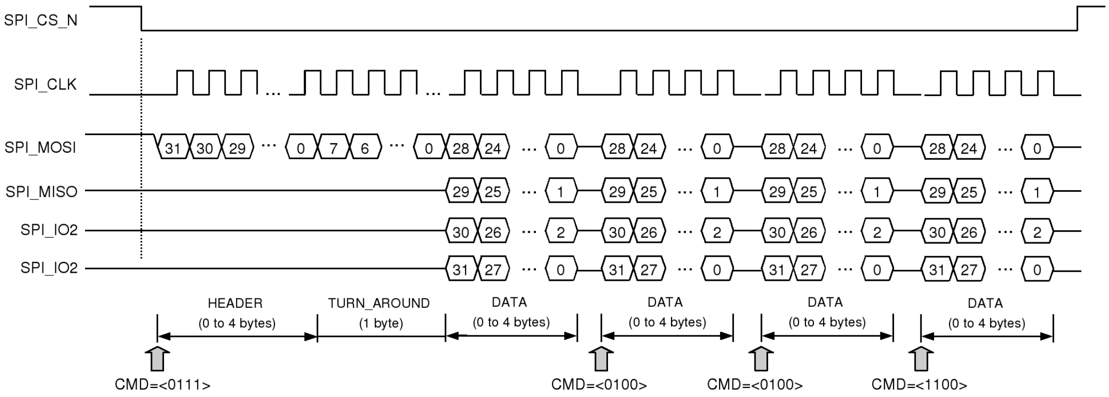


Figure 9-12 SPI Timing Diagram - Quad Mode

9.7.5 Boot State Machine

There are three different ways to trigger the boot state machine to begin executing commands from the serial ROM:

1. **On boot** — After MASTER_RESET is de-asserted and the fusebox has been read, and if the SPI_BOOT pin is asserted. On boot, the SPI options and starting address are read from location 0x000000 of the SPI image.
2. **On command** — When BSM_CTRL.Command is written to LOAD_FROM_SPI from some other value. In this case the SPI options and starting address are taken from BSM_ARGS.
3. **On interrupt** — When there is a global interrupt directed to the BSM (i.e. |(GLOBAL_INTERRUPT_DETECT & ~INTERRUPT_MASK_BSM) is true) and BSM_CTRL.*Command* is set to NONE. In this case the SPI options and starting address are taken from BSM_ARGS.

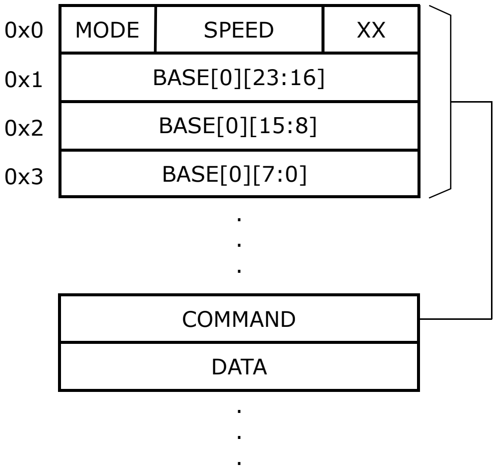
In any of these cases, the boot state machine will continue executing commands until it reaches an END command, or an execution error, or BSM_CTRL.*Command* is set to STOP_BOOT. No other conditions terminate BSM execution; in particular, the BSM is not interrupted while it is already executing. If the BSM was triggered by interrupt, and the interrupt bit is not cleared or masked before the BSM halts, when it halts the interrupt immediately re-triggers the BSM.

Note: If the FM10000 is reset during boot, the EEPROM should also be reset.

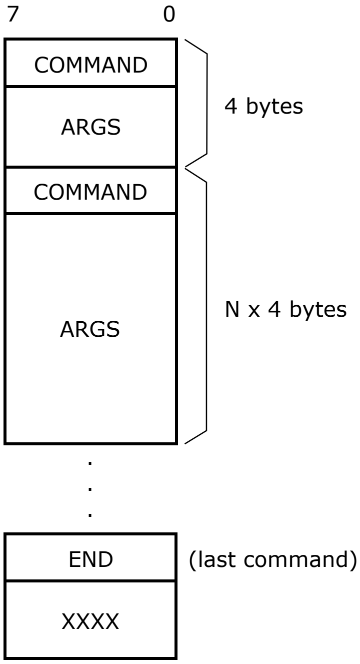
9.7.5.1 Image Format

The first four bytes of the serial ROM define the starting address of the boot instructions, as well as the operating mode and speed of the interface. These four bytes are only used on boot. If execution is triggered by interrupt, or by the LOAD_FROM_SPI command in the BSM_CTRL register, then mode, speed, and base address are instead taken from the BSM_ARGS register.

The first byte defines the SPI operating mode and speed. Its format is defined under SPI Protocol above. The next three bytes are the ROM address of the first instruction to execute.



The remainder of the serial ROM image format is a series of instructions encoded as shown below. The size of each instruction is at least 4 bytes long and always a multiple of four bytes. The first byte defines the command, including options, and the remaining bytes are arguments to the command.



9.7.5.2 Overview of BSM Operation

Addresses in the FM10000 memory space are expressed relative to an optional base offset in BSM_ADDR_OFFSET. The offsets used are chosen by the 2-bit values AA or BB in the instruction encoding in Table 9-8. Thus, if the offset selector is AA, `m[ADDR]` represents `mem[ADDR + BSM_ADDR_OFFSET[AA]]`. There are 3 programmable offsets, BSM_ADDR_OFFSET[1..3]; BSM_ADDR_OFFSET[0] is always treated as 0x0.

Instructions that can modify register values have the same semantics and side effects that management writes to those registers would have, depending on the target register. Thus, for example, a SET of an RO register field has no effect; COPYing a CW1 register to itself clears the register.

There are two 16-bit counter registers, BSM_COUNTER[0..1], that are used by the LOOP instruction for optimized fixed-length loops. These can be loaded via the WRITE or COPY instructions.

9.7.5.3 Instruction Table

The command encoding is detailed in Table 9-8. Here AA and BB define the base offsets for ADDR and ADDR_B, as described in Section 9.7.5.2. XX means an option field whose use is described in the table.

The data and addresses are always encoded most significant byte (MSB) first. Examples illustrating the byte ordering are:

```
WRITE ADDR=0x40104 DATA=0x19 => [00 04 01 04 00 00 00 19]

WAIT 10 usec => [FE 00 04 E2]
```

If the boot state machine reads an illegal command byte (one not defined in Table 9-8), it halts with an error indicated in BSM_CTRL.*EepromError*.

Table 9-8 Boot ROM Command Encoding

Command	Encoding	Description	Agrs & Data

WRITE	0b00XXXXAA	Write up to 16 words of fixed data, starting at ADDR.
		Variable length instruction. XXXX is the number of words of DATA minus 1.
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = DATA[0][31:24]
			ARGS[4] = DATA[0][23:16]
			ARGS[5] = DATA[0][15:8]
			ARGS[6] = DATA[0][7:0]
			ARGS[7] = DATA[1][31:24]
			.
			.
			.
			ARGS[4*XXXX+3] = DATA[XXXX][31:24]
			ARGS[4*XXXX+4] = DATA[XXXX][23:16]
			ARGS[4*XXXX+5] = DATA[XXXX][15:8]
			ARGS[4*XXXX+6] = DATA[XXXX][7:0]

COPY	0b01XXBBAA	Copy a block of up to 4 words, starting at ADDR (offset AA), to ADDR_B (offset BB).
		XX is the number of words to copy minus 1.
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = not used
			ARGS[4] = ADDR_B[23:16]
			ARGS[5] = ADDR_B[15:8]
			ARGS[6] = ADDR_B[7:0]

Table 9-8 Boot ROM Command Encoding [Continued]

Command	Encoding	Description	Agrs & Data

LOAD	0b11000XAA	Initialize a large block of memory with fixed code, starting at ADDR.
		X is the address increment used for successive words of DATA. Thus, setting X=0 allows repeated writes to the same ADDR.
		There are special addresses to be used for loading PCIe Spico firmware. Following a LOAD to one of these special addresses, there must be a delay of 10 µs before any subsequent write to the SBUS_PCIE_COMMAND register.
		• ADDR[23:0] = 0x00 & AA = 0b00: For PCIe SBus Master Spico.
		• ADDR[23:0] = 0x01 & AA = 0b00: For PCIe SerDes Spico
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = not used
			ARGS[4] = COUNT[23:16]
			ARGS[5] = COUNT[15:8]
			ARGS[6] = COUNT[7:0]
			ARGS[7] = DATA[0][31:24]
			ARGS[8] = DATA[0][23:16]
			ARGS[9] = DATA[0][15:8]
			ARGS[10] = DATA[0][7:0]
			.
			.
			.
			ARGS[4*COUNT+3] = DATA[COUNT-1][31:24]
			ARGS[4*COUNT+4] = DATA[COUNT-1][23:16]
			ARGS[4*COUNT+5] = DATA[COUNT-1][15:8]
			ARGS[4*COUNT+6] = DATA[COUNT-1][7:0]

INIT	0b11001XAA	Initialize a large block of memory with an increment pattern, starting at ADDR.
		X is the address increment used for successive words of DATA; thus setting X=0 allows repeated writes to the same ADDR.
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = not used
			ARGS[4] = COUNT[23:16]
			ARGS[5] = COUNT[15:8]
			ARGS[6] = COUNT[7:0]
			ARGS[7] = DATA[31:24]
			ARGS[8] = DATA[23:16]
			ARGS[9] = DATA[15:8]
			ARGS[10] = DATA[7:0]
			ARGS[11] = DATA_ADD[31:24]
			ARGS[12] = DATA_ADD[23:16]
			ARGS[13] = DATA_ADD[15:8]
			ARGS[14] = DATA_ADD[7:0]

CALC	0b100XXXAA	Perform a 2-input calculation and write the result:
		`m[ADDR_D] = m[ADDR_S] OP m[ADDR_T]`
		All addresses use the same base AA. XXX defines the operation:
		0 = AND
		1 = OR
		2 = ADD
		3 = SUBTRACT
		4 = SHIFT LEFT (if m[ADDR_T] > 31, result is 0x0)
		5 = SHIFT RIGHT
		6 = SHIFT RIGHT ARITHMETIC (sign extend)
			ARGS[0] = ADDR_D[23:16]
			ARGS[1] = ADDR_D[15:8]
			ARGS[2] = ADDR_D[7:0]
			ARGS[3] = not used
			ARGS[4] = ADDR_S[23:16]
			ARGS[5] = ADDR_S[15:8]
			ARGS[6] = ADDR_S[7:0]
			ARGS[7] = not used
			ARGS[8] = ADDR_T[23:16]
			ARGS[9] = ADDR_T[15:8]
			ARGS[10] = ADDR_T[7:0]

CALC_IMM	0b101XXXAA	Perform a 2-input calculation with an immediate value, and write the result:
		`m[ADDR_D] = m[ADDR_S] OP IMM`
		All addresses use the same base AA. XXX
		defines the operation, with the same encoding
		as CALC.
			ARGS[0] = ADDR_D[23:16]
			ARGS[1] = ADDR_D[15:8]
			ARGS[2] = ADDR_D[7:0]
			ARGS[3] = not used
			ARGS[4] = ADDR_S[23:16]
			ARGS[5] = ADDR_S[15:8]
			ARGS[6] = ADDR_S[7:0]
			ARGS[7] = IMM[31:24]
			ARGS[8] = IMM[23:16]
			ARGS[9] = IMM[15:8]
			ARGS[10] = IMM[7:0]

Table 9-8 Boot ROM Command Encoding [Continued]

Command	Encoding	Description	Agrs & Data

BRANCH	0b11010XAA	Conditionally jump to JUMP_ADDRESS, or continue to next instruction, depending on a masked value. X indicates the sense of the test:
		0 = Jump if:
		`(m[ADDR] & MASK) != VALUE`
		1 Jump if:
		`(m[ADDR] & MASK) == VALUE`
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = VALUE[31:24]
			ARGS[4] = VALUE[23:16]
			ARGS[5] = VALUE[15:8]
			ARGS[6] = VALUE[7:0]
			ARGS[7] = MASK[31:24]
			ARGS[8] = MASK[23:16]
			ARGS[9] = MASK[15:8]
			ARGS[10] = MASK[7:0]
			ARGS[11] = not used
			ARGS[12] = JUMP_ADDRESS[23:16]
			ARGS[13] = JUMP_ADDRESS[15:8]
			ARGS[14] = JUMP_ADDRESS[7:0]

POLL	0b11011XAA	Repeatedly poll a value, as in BRANCH, until either the value is found or max retry is reached. If it is max retry, the jump is taken. It the value is found, continue to the next instruction.
		RETRY_INTERVAL is in PCIE_REFCLK clock cycles.
		X indicates the sense of the test:
		0 = Continue if
		`(m[ADDR] & MASK) == VALUE`
		1 = Continue if
		`(m[ADDR] & MASK) != VALUE`
			ARGS[0] = ADDR[23:16]
			ARGS[1] = ADDR[15:8]
			ARGS[2] = ADDR[7:0]
			ARGS[3] = VALUE[31:24]
			ARGS[4] = VALUE[23:16]
			ARGS[5] = VALUE[15:8]
			ARGS[6] = VALUE[7:0]
			ARGS[7] = MASK[31:24]
			ARGS[8] = MASK[23:16]
			ARGS[9] = MASK[15:8]
			ARGS[10] = MASK[7:0]
			ARGS[11] = MAX_RETRY[15:8]
			ARGS[12] = MAX_RETRY[7:0]
			ARGS[13] = RETRY_INTERVAL[15:8]
			ARGS[14] = RETRY_INTERVAL[7:0]
			ARGS[15] = not used
			ARGS[16] = JUMP_ADDRESS[23:16]
			ARGS[17] = JUMP_ADDRESS[15:8]
			ARGS[18] = JUMP_ADDRESS[7:0]

LOOP	0b1110000X	Loop until BSM_COUNTER[X] reaches 0. If the counter is nonzero, decrement it and jump to JUMP_ADDRESS.	ARGS[0] = JUMP_ADDRESS[23:16]
			ARGS[1] = JUMP_ADDRESS[15:8]
			ARGS[2] = JUMP_ADDRESS[7:0]

JUMP	0b11101000	Jump to JUMP_ADDRESS.	ARGS[0] = JUMP_ADDRESS[23:16]
			ARGS[1] = JUMP_ADDRESS[15:8]
			ARGS[2] = JUMP_ADDRESS[7:0]

RETURN	0b111100AA	Jump to address in m[JUMP_PTR].	ARGS[0] = JUMP_PTR[23:16]
			ARGS[1] = JUMP_PTR[15:8]
			ARGS[2] = JUMP_PTR[7:0]

SET	0b111110AA	Set field of register:
		`m[ADDR] = (m[ADDR] & ~MASK)
			VALUE`	ARGS[0] = ADDR[23:16]
				ARGS[1] = ADDR[15:8]
				ARGS[2] = ADDR[7:0]
				ARGS[3] = VALUE[31:24]
				ARGS[4] = VALUE[23:16]
				ARGS[5] = VALUE[15:8]
				ARGS[6] = VALUE[7:0]
				ARGS[7] = MASK[31:24]
				ARGS[8] = MASK[23:16]
				ARGS[9] = MASK[15:8]
				ARGS[10] = MASK[7:0]

WAIT	0xFE	Delay. TIME is in PCIE_REFCLK clock cycles.	ARGS[0] = TIME[23:16]
			ARGS[1] = TIME[15:8]
			ARGS[2] = TIME[7:0]

END	0xFF	End of program.	Not used.

9.8 LED Controller

The LED interface consists of 5 signals; LED_CLK, LED_DATA0, LED_DATA1, LED_DATA2, and LED_EN. Those 5 signals are used to transmit 3 bits of status data per MAC over the time multiplexed data pins.

The LED interface is controlled via the LED_CFG register which defines:

- LED_FREQ (24 bits) — A divider to derive the LED_CLK from the PCIE_REFCLK (100 MHz).The exact frequency is equal to PCIE_REFCLK/(LED_FREQ+1)/2. A value of 0 is not supported.
- LED_ENABLE (1 bit) — Controls if the LED's are enabled (1b) or disabled (0b).

The overall timing diagram is shown Figure 9-13. Each MAC is identified with a number X.Y where 'X' is the EPL number (0 through 8) and 'Y' is the MAC number within that EPL (0 through 3).

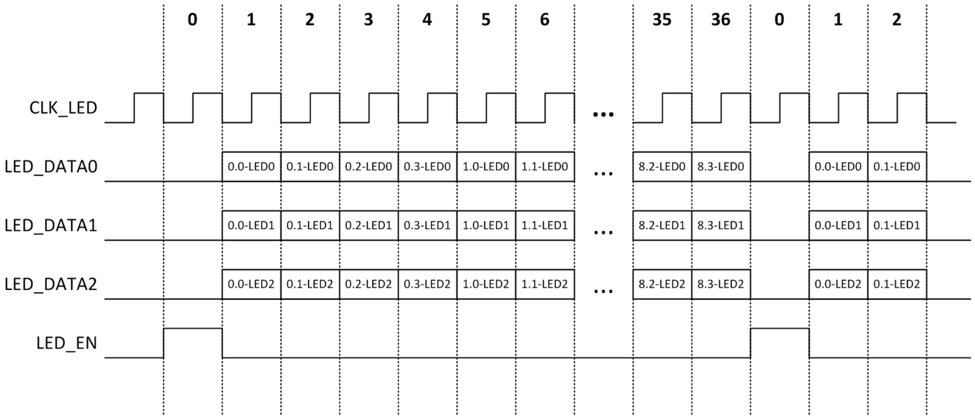


Figure 9-13 LED Status Timing Diagram

The LED_EN is asserted for one clock cycle to mark the beginning of a 37 clock cycle. Each successive clock cycle carry the 3 bits per MAC (9x4 = 36). The 3-bit LED state encoding in shown in Table 9-9.

Table 9-9 LED Controller State Decoding

DATA[2]	DATA[1]	DATA[0]	State
0	0	0	Port is in reset.
1	1	0	Port has detected local fault.
0	1	0	Port has detected remote fault.
0	0	1	Link is up and no fault. No packets transmitted or received since last sample.
1	0	1	Link is up and no fault. Packets transmitted since last sample.
0	1	1	Link is up and no fault. Packets received since last sample.
1	1	1	Link is up and no fault. Packets received and transmitted since last sample.
1	0	0	RESERVED.

The port status is derived from the EPL_LED_STATUS using the following priority decoding:

```
if ( EPL_LED_STATUS.Port?Reset )
   LED_State = 0;
 else if ( EPL_LED_STATUS.Port?LocalFault )
   LED_State = 6;
 else if ( EPL_LED_STATUS.Port?RemoteFault )
   LED_State = 2;
 else if ( EPL_LED_STATUS.Port?LinkUp )
   LED_State = 1 + (EPL_LED_STATUS.Port?Receiving)*2 + (EPL_LED_STATUS.Port?Transmitting)*4;
 else
   LED_State = 4;
```

9.9 System Time and IEEE 1588

The system time controller implements a time distribution through the device for each end point to use: Ethernet ports, PCIe interfaces, Tunnel Engines, or an external follower. The main purpose is to provide a synchronized time reference to time tag packets through the switch. The controller has the following features:

- Derive time from one of two sources:
  - — The dedicated IEEE 1588 reference clock (50 to 256 MHz).
  - — The PCIe reference clock (100 MHz).
- Time defined as multiple of reference clock.
  - — Multiple is 2..16 x reference clock (STEP).
- Internal adjustment available to compensate for reference clock drift.
  - — Adjustment down to N/(2^{48}) per reference clock cycle.
- Internal adjustment available to compensate for timing signal propagation delay.
- General period pulses for external reference.
- Capture transition time on up to 4 GPIOs.

Figure 9-14 shows the components of the system time distribution.

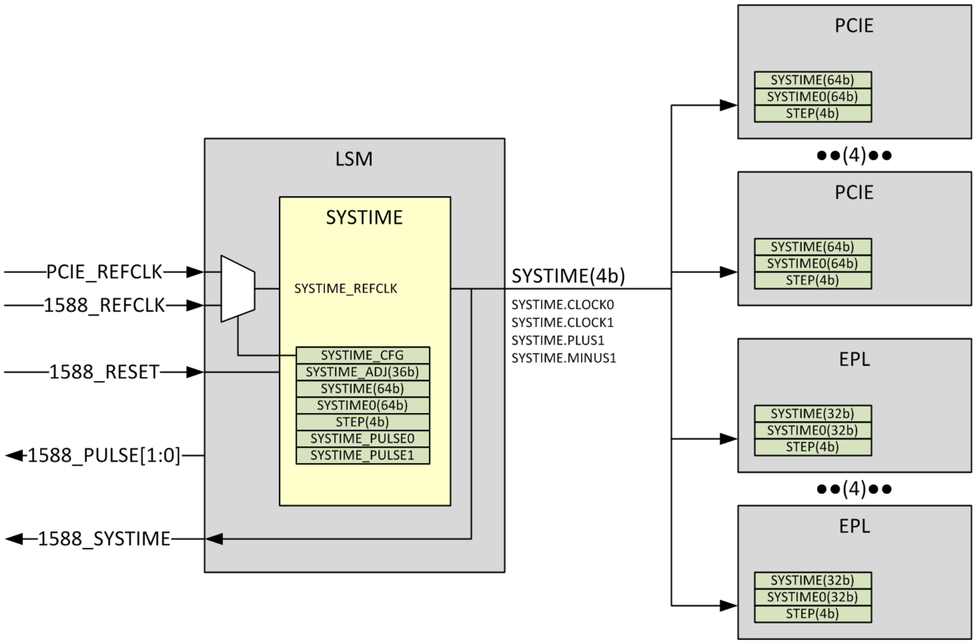


Figure 9-14 System Time Distribution

The external input pins are:

- Reference Clock (one of two choices):
  - IEEE1588_REFCLK
  - PCIE_REFCLK
- Time reset:
  - IEEE1588_RESET_N
- Optional capture of transition times:
  - GPIO[0..3]

External output pins are:

- 4-signal control for external clock follower:
  - IEEE1588_SYSTIME
- Configurable-frequency pulses:
  - IEEE1588_PULSE

The following values are configured in the system time controller in LSM:

- DEVICE_CFG.*SystimeClockSource* (1 bit) — Defines which clock source to use as the system time reference clock. This should be programmed before COLD_RESET is de-asserted.
- SYSTIME_CFG.*Adjust* (38 bits) — Defines the 1588_RESET adjustment fraction.
- SYSTIME_CFG.*Dir* (1 bit) — Indicates if the adjustment is de-asserted positive (1) or negative (0).

The following values are configured both in the system time controller and in each systime end point:

- SYSTIME_CFG.*STEP* (4 bits) — Defines the internal time delta added to SYSTIME at each rising edge of the systime refclock. The same STEP is programmed in each end point of the system as well.
- SYSTIME0 (64 bits) — Indicates the initial time T0 at systime reset. SYSTIME0 is also programmed in each end point, with optional adjustment from the controller SYSTIME0 to account for signal propagation latency.

The function of the controller is to maintain a local system time (SYSTIME, 64 bits), as well as generating a set of four signals which are distributed to Ethernet, PCIe, and TE end points to maintain a local copy of SYSTIME and replicated on the IEEE1588_SYSTIME output pins for an external follower.

When the system time reset IEEE1588_RESET_N pin is found, the controller resets SYSTIME to SYSTIME0 and maintains this value as long as the reset is asserted. When reset is not asserted, SYSTIME is incremented by STEP at each posedge of reference clock.

The Adjust value is used to add a small deviation to the rate at which SYSTIME advances. Adjust defines how many times an extra +1 (or -1) is added to SYSTIME per 2^{48} reference clock cycles, distributed at roughly even time intervals. The total advance in SYSTIME is thus (STEP * 2^{48}) + (Adjust * (Dir?+1:-1)) in 2^{48} refclock cycles. The maximum Adjust value is 0x3F_FFFFFFF, and, in this case, SYSTIME is incremented by an extra +1 (or -1) once every refclock 1024 cycles. This is enough to allow compensation from 1 ns per day to 100 ppm (assuming a 100 MHz reference clock or faster and a step of 10, SYSTIME representing nanoseconds).

The Adjust value can be computed from the deviation required. As an example, if the intended reference clock frequency is at 250 MHz and STEP is 4 (so SYSTIME is representing nanoseconds), but the actual reference clock is at 249.995 MHz, the Adjust is:

```
SYSTIME_CFG.Adjust = STEP * (250.000-249.995)/249.995 * 2^^48
                   = 22,518,448,506
```

An external monitor may track the drift in SYSTIME relative to a reference, and modify the Adjust configuration during operation as necessary. The deviation from linearity of the SYSTIME counter is bounded by STEP + ceil(log2(Adjust))/2 + 1, or 35 SYSTIME units in the worst-case configuration.

The four signals distributed by the controller to all targets are:

- SYSTIME.*CLOCK0/1* — Two divide-by-four of reference clock, 90 degree phase to each other. Each edge indicates to add STEP to SYSTIME in the target.
- SYSTIME.*PLUS1* — A positive transition indicates add 1 to SYSTIME. Do one increment +1 per posedge regardless of how long PLUS1 is asserted.
- SYSTIME.*MINUS1* — A positive transition indicates to subtract 1 from SYSTIME. Do one decrement -1 per posedge regardless of how long MINUS1 is asserted.

These four signals are reflected on the IEEE1588_SYSTIME output pins. The PLUS1 and MINUS1 are replicating the +1 and -1 events on the LSM controller, allowing the targets to follow. The maximum rate of PLUS1 and MINUS1 is 1/1024th of the reference clock.

A systime reset is indicated to the targets by simultaneously asserting PLUS1 and MINUS1. When systime is taken out of reset, the controller restarts its counters such that the next PLUS1 and MINUS1 transitions are away from the reset event, to avoid glitches on the PLUS1 and MINUS1 signals.

The local SYSTIME in targets (EPLs, PCIe and TE) continues to be maintained even if those targets are in warm reset (i.e., when SOFT_RESET.*EPLReset* is asserted for EPLs, or PCIE_RESET_N[i] or SOFT_RESET.*PCIeReset*[i] is asserted for PCIe[i]), or when the Ethernet switch itself is in reset. However, the LSM SYSTIME update and signal distribution is stopped when the chip is reset (via

RESET_N pin or via watchdog), and synchronization is lost in this case and must be re-established by reloading SYSTIME0 and re-asserting 1588_RESET_N.

The timing diagram is shown in Figure 9-15.

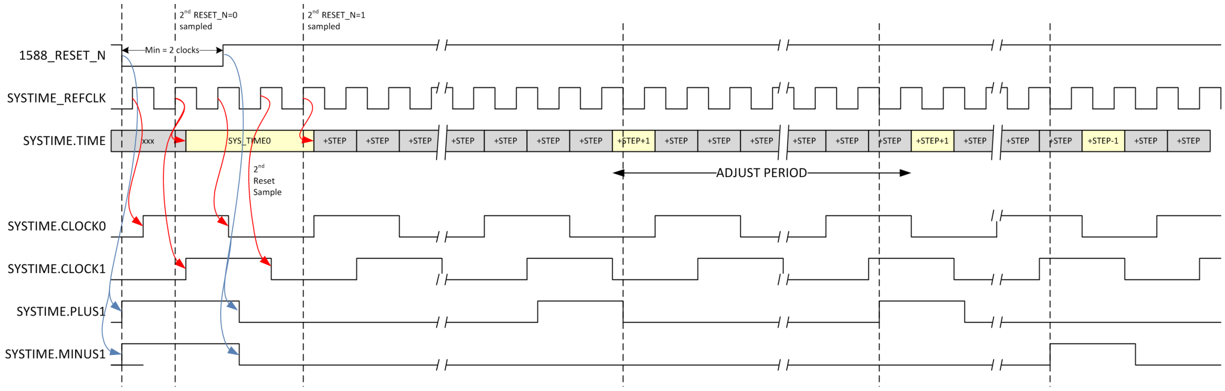


Figure 9-15 System Time Distribution Timing

The controller also generates two pulse outputs for external reference, on output pins IEEE1588_PULSE[0..1]. The time periods of the pulses are configured in SYSTIME_PULSE[0..1], measured in SYSTIME units. For instance, if SYSTIME is configured to represent nanoseconds, setting SYSTIME_PULSE[0] = 125000 causes IEEE1588_PULSE[0] to produce pulses at 8 KHz = 1/125000 ns. The width of the pulses is 8 refclock cycles. The pulse timers are restarted on assertion of IEEE1588_RESET_N.

Finally, the controller can capture the current system time on a transition of pins GPIO[0..3]. Either edge can be captured, or both. The edges to capture are defined in SYSTIME_CFG.*CaptureHigh*[0..3]/ *CaptureLow*[0..3].

9.9.1 Initialization of System Time

The IEEE 1588 reference clock may come from either PCIE_REFCLK, or the dedicated IEEE1588_REFCLK pin.This choice must be configured in DEVICE_CFG.*SystimeClockSource* before de-asserting COLD_RESET.

To initialize the SYSTIME counters (after COLD_RESET has been de-asserted):

1. To be programmed. all endpoints must be out of reset, as determined by the SOFT_RESET register. Temporarily set SOFT_RESET.*PCIeActive* = DEVICE_CFG.*PCIeEnable*, to force all enabled PEPs out of reset. (Do not write to non-enabled PEPs.)
2. Configure SYSTIME_CFG, and set XXX_SYSTIME_CFG.*Step* equal to SYSTIME_CFG.*STEP*.
3. Program the master SYSTIME0 in all enabled endpoints, to a time sufficiently far in the future. Due to skew and delay in the distribution of the 1588 systime protocol, configure all XXX_SYSTIME0 registers to 0.
4. Once PCIE_SYSTIMEXXX has been programmed in all enabled PEPs, SOFT_RESET.*PCIeActive* may be cleared.
5. De-assert the external IEEE1588_RESET_N pin (toggle from low to high) to initialize all SYSTIME registers to SYSTIME0 and begin counting time.

9.10 In-Band Management

In-band management provides a means to reliably configure and control multiple fabric devices in a system, reducing the need for CPU resources connected to each switch.

9.10.1 Overview

FTAG In-Band Management (FIBM) is the management of one or more FM10000 series, FM5000/FM6000 series, or FM4000 series chips through management commands encapsulated within Ethernet frames. This allows switches to be entirely managed remotely. There is no fixed limit to the number of switch chips that one CPU can manage through FIBM. The maximum number is determined by the bandwidth of the CPU interface, memory size and processing speed, and depends on the application. The usage of FIBM is inherently more demanding in terms of CPU processing speed than a directly attached switch, as all register accesses have to be done through Ethernet packets.

The management operations that can be performed through FIBM are: Register Read, Register Write, Register Read/Write, Delay, NOP, and Interrupts. Data is included with the write requests and read responses. A delay operation specifies the number of clock cycles before the next FIBM operation is executed. A NOP does not do anything and is only used for padding frames to 64 bytes. These operations are discussed in more detail below.

The features supported are:

- Arbitrary number of read/write/delay operations per manager request up to the maximum size of the frame (2 KB).
- Flexible protocol design for reliability and security.
- Automatic interrupt posting.
- Autoboot for fully CPU-less system.

Figure 9-16 shows the FIBM module integration in the switch.

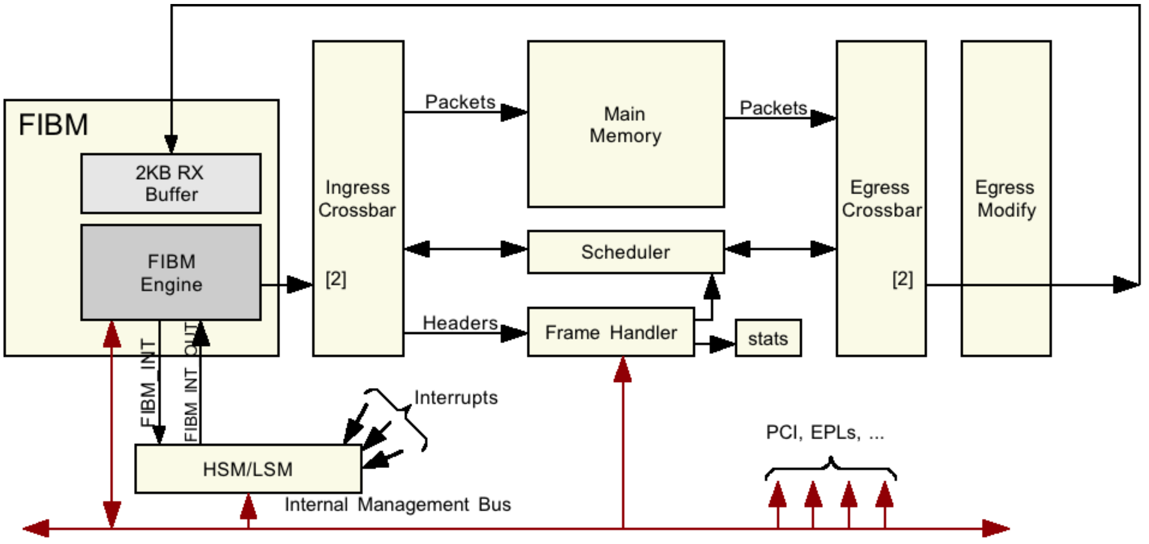


Figure 9-16 FIBM Module Integration

The FIBM is connected to of the fabric and to the management bus. The Frame Processing Pipeline is programmed to dispatch FIBM frames received from the network to the FIBM port. The FIBM module receives the frame into a temporary buffer to ensure the frame is fully valid before starting processing. Then, the FIBM engine unrolls the packet and executes any commands stored in the packet (read/write) and constructs a response, which is pushed to the fabric. The packet is sent to the fabric when the response has been fully constructed.

If an interrupt is received from the interrupt controller and FIBM is configured to forward interrupts, the FIBM generates an interrupt frame and sends it to the fabric. The interrupt frames can only be generated while FIBM is idle (i.e. not processing a request and not sending a response).

9.10.2 FIBM Frame Format

There are 3 types of FIBM frames.

- **Request** — Generated by host.
- **Response** — Generated by switch.
- **Interrupt** — Generated by switch.

The FIBM format is shown in Figure 9-17:

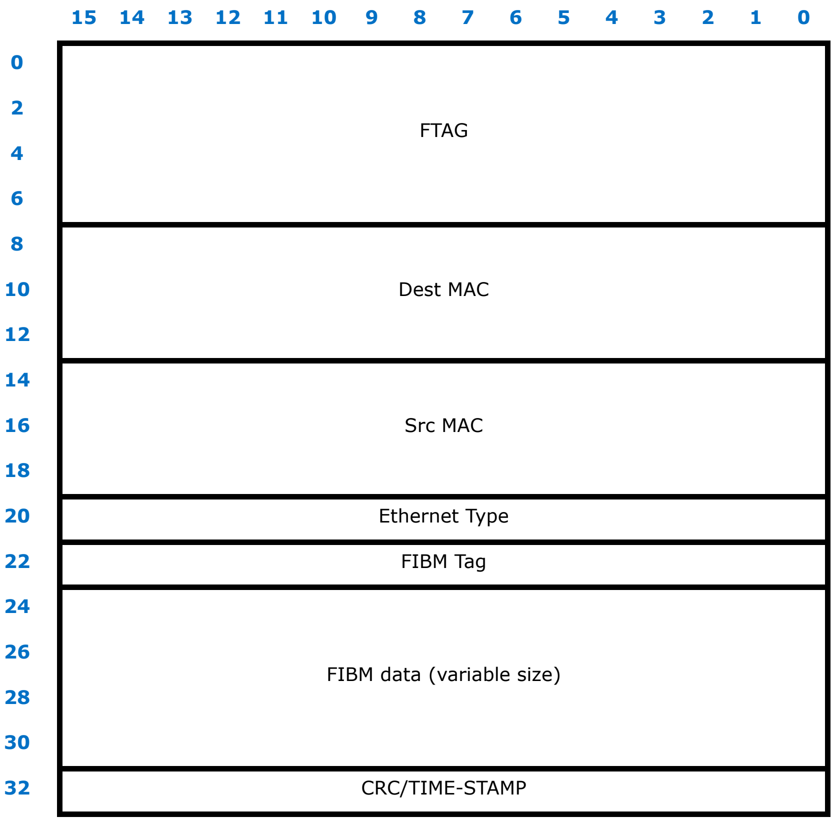


Figure 9-17 FIBM Frame Format 2

The CRC/TIME-STAMP presence is optional. The FIBM_CFG.*ignoreLast4Bytes* register field defines if this field is present or not. If this bit is set, the CRC/Time-Stamp is present and ignored.

Similarly, when a frame is generated, the FIBM_CFG.*append4bytes* indicates if the frame is padded with an extra four zero bytes while the field FIBM_CFG.*padIbmResponse* indicates if the frame is padded to 64 bytes as a minimum frame size.

The disposition of each field is detailed in Table 9-10.

Table 9-10 FIBM Frame Disposition

Field	Request	Response	Interrupt
DMAC	Don't care.	SMAC of the request frame.	FIBM_INT_FRAME_DMAC
SMAC	Don't care.	DMAC of the request frame.	FIBM_INT_FRAME_SMAC
FTAG.FTYPE	Don’t care.	FIBM_CFG.*responseType*	FIBM_CFG.*interruptType*
FTAG.DGLORT	Port 2 GLORT of the destination switch. This field is ignored by the FIBM module.	Src GLORT of the request frame. Defined in FIBM_SGLORT.	Set to FIBM_INT.*interruptGlort*.
FTAG.SGLORT	Destination GLORT for the response frame.	If FIBM_CFG.*ibmGlortEn* is set, set to FIBM_SGLORT. Otherwise, Dst GLORT of the request frame.	Set to FIBM_SGLORT.
FTAG.SWPRI	Priority.	Same as the request frame.	FIBM_INT_FRAME.*islSysPri*
FTAG.USER	Don’t care.	Set to 0b.	Set to 0b.
FTAG.{VPRI/VID}	Ignored	Set to 0b.	Set to 0b.
Ethertype	Ignored	FIBM_INT_FRAME.*etherType*	FIBM_INT_FRAME.*etherType*
FIBM Tag	Host supplied value.	Same as the request frame.	Set to 0b.
FIBM Ops & Data	See below.	See below.	See below.
CRC/TIME-STAMP	Optionally present. Ignore if present.	Optional set to zero.	Optional set to zero.

9.10.2.1 Processing Incoming Requests

The frame received is stored in a receive buffer until the end of the frame is seen. If the frame has reached 2 KB in length and the TE has not been seen yet, the frame is dropped as an error and the receiver flushes the rest of the frame. If the frame length is valid, the frame is dropped if the TE code indicates there is a framing error or a data check error or the type is not a request. Otherwise, the frame content is processed. While processing the content, if an invalid op-code is detected, the invalid op-code is copied into the response and processing of op-codes is terminated.

The response is padded if requested.

9.10.2.2 Reliable FIBM Communication

Ethernet does not provide a reliable communication link; i.e., frames can be dropped during transport through a network of switches. However, FIBM provides the means for reliable in-band management.

9.10.2.2.1 Sequence Numbers

Reliable communication over an Ethernet link basically requires two things: 1) acknowledgments, and 2) sequence numbers. Since an FIBM response frame is returned for every request frame, the response frame is the acknowledgment for the request frame. The FIBM Tag data in a request frame is placed in the Tag field of the corresponding response frame. Therefore, the CPU can easily assign each FIBM request frame a 16-bit sequence number in the Tag field.

9.10.2.2.2 Scratch Registers

There are 16 scratch registers in the FIBM block (FIBM_SCRATCH_[0..15]). Since they are RW, they can be accessed with an FIBM Read, Write, or Read/Write operation. With a Read or Write, they are accessed by their address, just like any other register. With a Read/Write operation they are accessed by a 4-bit relative address contained in the Read/Write operation.

9.10.2.2.3 Reading and Writing Idempotent Registers

An operation is idempotent if it can be repeated and the effect is the same whether it is executed once or twice. An effort has been made to make most register operations idempotent. For idempotent register operations, a remote CPU can:

- Put sequence numbers in the tag field.
- Issue multiple FIBM request frames with unique sequence numbers.
- If an FIBM response frame with a particular sequence number is not received after some period of time, the CPU can just re-send the request frame for that sequence number.

9.10.2.2.4 Writing Non-Idempotent Registers

If an acknowledgment is not received for a Write request to a non-idempotent register, it is necessary to be able to find out whether the Write operation was executed or not (whether the FIBM request frame got dropped or the FIBM response frame got dropped). A sequence number can be written to a scratch register by the same request frame that writes data to a non-idempotent register. If an FIBM response frame is not received, the CPU can read the scratch register and determine whether the FIBM request frame operations were carried out. The scratch register is idempotent. Therefore, the CPU can read it as many times as necessary.

9.10.2.2.5 Reading Non-Idempotent Registers

If an acknowledgment is not received for a Read request to a non-idempotent register, it is necessary to both 1) be able to find out whether the Read operation was executed or not (whether the FIBM request frame got lost or the FIBM response frame got lost), and 2) if the Read operation has already been executed, be able to find out what the original data was. Sequence numbers can be written to scratch registers as described above. In addition, FIBM Read/Write operations should be used to read non-idempotent registers. The Read/Write operation has a 22-bit address for the non-idempotent register and a 4-bit address for the scratch register. The non-idempotent register is read in the same manner as a normal register and, in addition, the register data is written to a scratch register. If no FIBM response frame is received by the CPU, the scratch registers can be read as many times as necessary to determine whether the initial Read/Write operation was carried out (from the sequence number); and, if it was carried out, what the original data was (stored in a scratch register by the Read/Write operation).

9.10.2.2.6 Interrupts

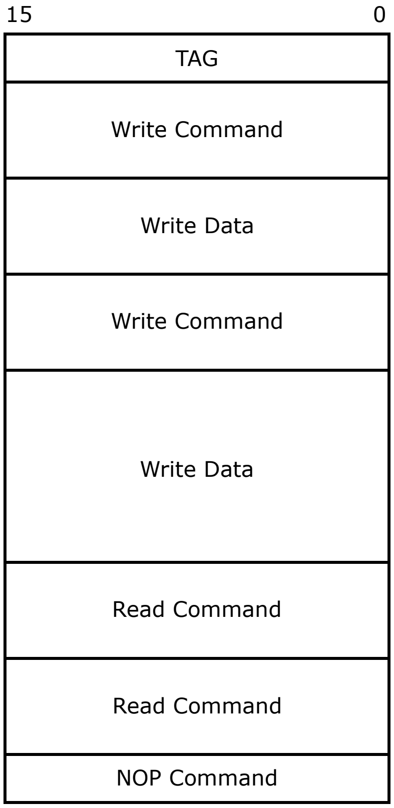
Since there is no guarantee that a single Interrupt frame is received by the CPU, interrupt frames are sent repeatedly until the interrupt is cleared. Refer to Section 9.10.3.

9.10.2.2.7 Timing

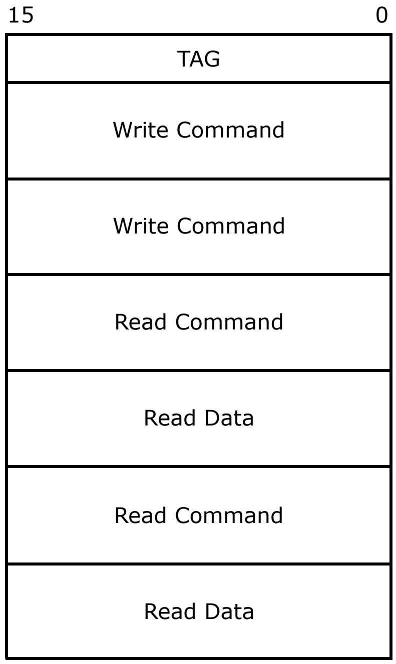
Since the time for an FIBM request frame to transit from a remote CPU to the chip being managed is indeterminate (especially since there is no guarantee that it arrives at all), FIBM operations cannot be executed at a precise absolute times. If a switch is being managed through FIBM, it must be managed in a way that does not require precise absolute timing. However, FIBM does provide a Delay operation so that FIBM operations can be executed with precise relative timing. A Delay operation specifies the number of FIBM clock cycles before the next FIBM operation is executed.

9.10.2.3 FIBM Payload Format

An example FIBM request frame payload is shown below. Each FIBM operation (Read Request, Write Request, etc.) is specified by an FIBM header word (HW), and if it is a Write Request, it is followed by the appropriate number of data words. Multiple FIBM operations are allowed in a single frame payload. The maximum length of a Read or Write is generally 16 words but see the note below for certain Write restrictions. Length is the "real length" modulo 16 (Length=0b0000 means 16 words). If it is desired to do a Read or Write Request of more than 16 words, multiple header words must be used.



A response is similar and shown below. Each request header word is repeated in the response frame. Read request header words are followed by the data. The data for a Write request is not included in the response frame.



9.10.2.3.1 Minimum FIBM Frame Payload Length

An FIBM frame header, tag, and tail is 28 bytes, thus a single FIBM Read Request could as short as 32 bytes which is smaller than the minimum Ethernet frame size. The frame is padded with NOPs (0s) to complete the frame. This can be done automatically for the request from CPU if sent via PCIe or MSB. For responses, the FIBM module includes an FIBM_CFG.*padIbmResponse* option to automatically pad the frame.

9.10.2.3.2 Maximum FIBM Frame Payload Length

The maximum length for FIBM request operations is 2044 bytes including header and trailer (CRC). The FIBM module has a protection to ignore any packet exceeded this size. Also, the maximum response length is 15860. The FIBM module does not include any protection to limit the size of the response. Therefore, software is responsible to ensure that no request or response exceeds these limits.

9.10.2.4 FIBM Header Words

Each FIBM operation is defined by a 32 bit header word as shown in Table 9-11.

Table 9-11 FIBM Header Words

OP	31	30	29	28	27 26 25 24	23 ... 0
NOP	0	0	0	x	nwords	x
Read	0	0	1	x	nwords	address
Write	0	1	0	x	nwords	address
Read/Write	0	1	1	x	scratch reg.	address
Delay	1	0	0	x	x	delay
Error	1	0	1	x	x	x
Interrupt	1	1	1	x	x	data

- NOP's are the only operations in an FIBM request frame that are not included in the FIBM response frame. If an FIBM response frame is padded to 64 bytes, it is padded with NOP's.
- The maximum Read and Write Length is 16 words (Length=0b0000 means 16 words). If a longer Read or Write request is desired, the CPU must break it up into multiple requests.
- A single Read or Write request is not allowed to cross a register address boundary.
- A Read/Write operation reads one register (specified by Address) and returns the data just as a normal Read operation would. In addition, it writes the data to a scratch register (specified by ScrAddr).
- The data field in a Delay operation specifies the number of FIBM clock cycles before the next FIBM operation is executed.
- The FIBM is processed one operation at a time, the processing continues until all operations have been executed or until an error is encountered. If an error is encountered, an error code is inserted in the response and the rest of the payload of the packet is copied to the response. The error could be of various type; invalid op-code, end of frame encountered before finishing operation, invalid address. The processing is the same regardless of the error, and it is the responsibility of the sender to recover what the error is from the packet content.

9.10.3 Interrupts and RESET

9.10.3.1 FIBM Unit Reset

The FIBM unit is connected on the MASTER_RESET and thus reset at the same time as the management infrastructure. It is not possible to reset the FIBM unit on its own.

9.10.3.2 Switch Reset

If FIBM is enabled and a switch reset is needed, the local switch manager must disable FIBM (FIBM_CFG.*disableIbm*=1b) and disable interrupt generation (FIBM_CFG.*interruptGlortEn*=0b) prior to resetting the switch. The unit terminates after the current in-flight frame completes (writing commands are immediately flushed) and ignores any further packets from the switch until enabled again. The unit should be enabled again only when the switch is fully up and running. Note that the time to terminate an in-flight packet depends on the size of packets and possible commands inside the packet.

9.10.3.3 Forwarding Interrupts

The top level GLOBAL_INTERRUPT_DETECT register is in the interrupt controller unit. An interrupt mask in that module, INTERRUPT_MASK_FIBM, allows it to select which of the global interrupts is posted to FIBM. If an interrupt is present and the FIBM mask is cleared for that interrupt, the FIBM interrupt input is asserted. If FIBM_CFG.*interruptGlortEn* is set, the FIBM block repeatedly sends FIBM Interrupt frames out its network to FIBM_IBM_INT.*interruptGlort* until the interrupt signal from the LSM is not asserted (until the interrupt is cleared). The interval between sending interrupts is determined by FIBM_IBM_INT.*interruptInterval*. This is a 16-bit register, and the time unit is 8192 FIBM clock cycles (8192/375 = 21.8 μs). Counting starts at 0; i.e., for interruptInterval = N, the actual interval is (N+1)*21.8 μs. This gives a range of ~21.8 μs to ~1.43 sec.

The FIBM Interrupt frame sent by the FIBM is shown in Table 9-12.

Table 9-12 FIBM Interrupt Frame

Interrupt Frame		
Dest MAC		FIBM_INT_FRAME_DMAC
Src MAC		FIBM_INT_FRAME_SMAC
ISL Tag	FTYPE	FIBM_CFG.*interruptType*[3:2]
	MTYPE	FIBM_CFG.*interruptType*[1:0]
	SYSPRI	FIBM_INT_FRAME.*islSysPri* (4 bits)
	USER	FIBM_INT_FRAME.*islUserBits* (8 bits)
	VLANPRI	FIBM_INT_FRAME.*vlanPRI*
	VLAN ID	FIBM_INT_FRAME.*vlanID*
	Src GLORT	FIBM_IBM_GLORT.*ibmGlort*
	Dst GLORT	FIBM_IBM_INT.*interruptGlort*
Ethertype		FIBM_INT_FRAME.*etherType*
FIBM Tag		FIBM_INTR_CTR_1.*interruptSeqCtr*
FIBM Interrupt Header Word		0xE0000000
8 NOP Words		0x00000000
CRC		CRC

FIBM_INTR_CTR_1.*interruptSeqCtr* is the number of FIBM Interrupt frames sent since the current interrupt signal from the HSM was first asserted.

The source GLORT in the ISL tag is FIBM_IBM_GLORT.*ibmGlort* whether FIBM_CFG.*ibmGlortEn* is set or not. If this register is not configured properly, the CPU does not know where the interrupt frames are coming from.

9.10.3.4 Disabling Interrupts

No FIBM interrupt frames are sent if FIBM_CFG.*interruptGlortEn* is not set.

9.10.3.5 RESET Initiated by a Remote CPU

A remote CPU can reset the chip by writing into the FATAL_ERROR register.

9.10.3.6 Boot After RESET

An FM10000 without an attached CPU boots from SPI. The boot sequence should include at least the following:

- Configure switch.
- Configure FIBM interrupt.
- Configure interrupt controller to post SW_IP interrupt to FIBM.
- Use an FIBM special source GLORT or Ethernet type to indicate a restart.
- Write a non-fatal, non-masked interrupt to the SW_IP register in the LSM.

When the reset occurs:

- Reset registers to their default value.
  - This include setting FIBM_CFG.*disableIBM* to 1b, which prevents any future FIBM write/read access to be executed.
- Restart loading the boot sequence.
  - This reprograms the source GLORT or FIBM Ethernet Type to a default value.

When the boot sequence completes:

- The SW_IP causes an interrupt to be posted to FIBM.
- The FIBM periodically sends out interrupt messages to FIBM_IBM_INT.*interruptGlort* until INTERRUPT_DETECT is cleared.
- The CPU:
  - Receives an interrupt frame.
  - Validates FIBM source GLORT or FIBM Ethernet Type to detect a self reset.
  - If a chip self reset has occurred:
    - Re-enables FIBM read/writes and clear interrupt generation (FIBM_CFG.*disableIBM* and FIBM_CFG.*interruptGlortEn* to 0b).
    - Fix esFIBM source GLORT or FIBM Ethernet Type for future interrupts.
    - Reads the LAST_FATAL_CODE and FATAL_COUNT registers in the Watchdog.
    - Clears SW_IP interrupt source.
    - Re-enables interrupts FIBM.interruptGlortEn to 1b.
  - If a chip reset has not occurred, process interrupts as normal.

NOTE: ***This page intentionally left blank.***

10.0 Reliability, Diagnostics and Testability

10.1 Reliability

10.1.1 Overview

In electronics and computing, an error is a signal or datum that is wrong. Errors can be classified as chip defects, resulting from either manufacturing or aging-related phenomena (such as electro-migration), or caused by high-energy particles that change the state of a single bit into a new unintended state. The latter category is called *soft error*, or *single event upset*, as the state change is transient and there is no permanent damage. The section describes the handling of soft errors.

Whether a circuit experiences a soft error depends on the energy of the incoming particle, the geometry of the impact, the location of the strike, and the design of the logic circuit. The FM10000 contains the following categories of circuits that are susceptible to soft errors:

- High-density static RAM (SRAM)
- Medium-density ternary CAM (TCAM)
- Low-density register files (REGISTERS)
- Logic (FLIP-FLOPS and GATES)

The nodes with the smallest capacitance, the SRAM circuits, are generally protected with ECC. The added ECC bits are updated when data is written to memory and are checked when data is read from memory. SRAM protected with ECC allows single-bit errors to be detected and corrected, and allows multiple-bit errors to be detected but not corrected. The primary exception is memory in the ingress and egress crossbars, which are parity-protected only (small memories), and in this case, soft errors are reported but not corrected.

All 6T SRAM errors (correctable or not) can be reported to software, which can perform any action deemed necessary to repair the issue. The device also contains control registers to simulate an error on most SRAM blocks, allowing software and hardware development teams to test the feature.

The nodes with the second smallest capacitance, the TCAMs, have three possible operations: write, read and lookup. The write and read operations are occasional (when TCAM content is updated), while the lookup operations occur for each packet that is switched, and thus are very high frequency. Due to the nature of the TCAM (i.e., parallel read of all bits during a lookup) it is very difficult to provide a protection during lookup. For this case, the FM10000 includes a hardware sweeper configured to read the TCAM continuously in the background, compute a checksum (in form of a XOR) on the contents of the TCAM, and report any unexpected changes to software via interrupts.

The register and register files are small and typically used for tiny memory blocks. They have higher capacitance than TCAM and SRAM, and thus are less susceptible to soft errors. Most of the register files are also protected by ECC or parity.

The final elements are flip-flops and gates which have the highest capacitance. the FM10000 provides no ECC or parity protection for those elements. For gates and flip-flops, the software can check integrity of the device by testing functionality and monitoring sanity of the device indirectly.

10.2 Diagnostics

10.2.1 Packet Loopbacks and Packet Test Interface

The FM10000 offers the following types of loopbacks:

- Internal loopback ports.
- Parallel loopbacks (facing switch and facing physical ports).
- Serial loopbacks on Ethernet ports.

The FM10000 also offers a Packet Test Interface, which can be used to generate or receive packets with arbitrary length, content and conditions.

Figure 10-1 shows the location of the loopbacks and the Packet Test Interface.

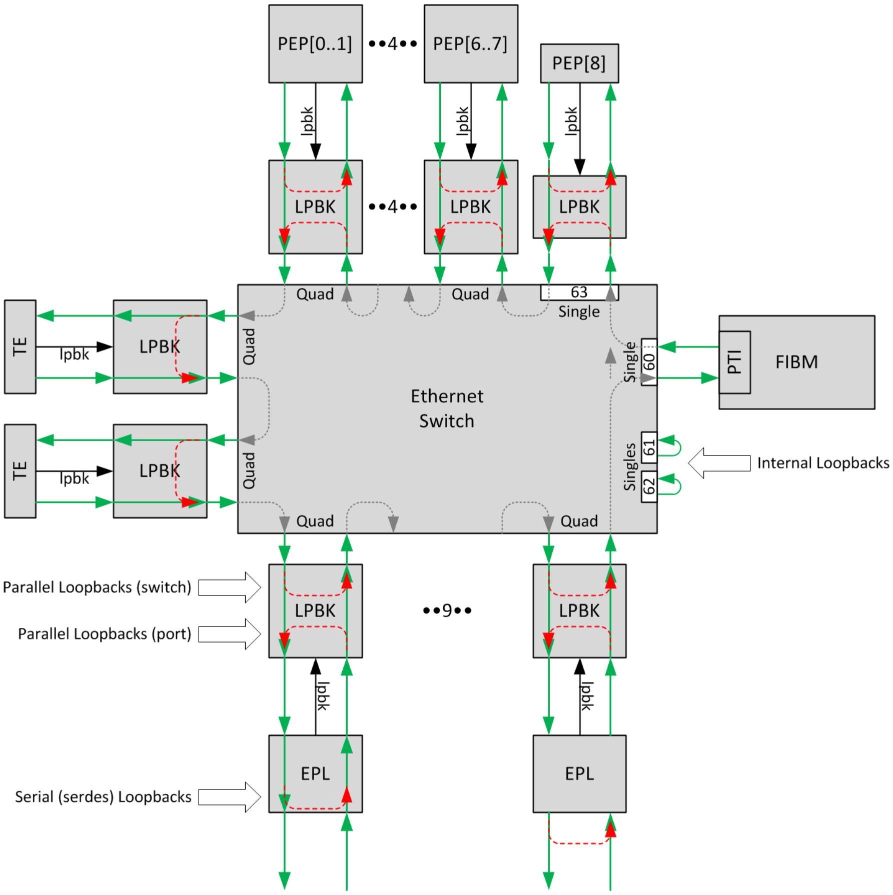


Figure 10-1 Loopbacks Locations and Types

10.2.1.1 Internal Loopback Port

The internal loopback ports, physical ports 61 & 62, are two ports of the Ethernet Switch that are permanent loopbacks. Any packet sent to those interfaces is looped back to the switch unmodified.

10.2.1.2 Parallel Loopbacks

The parallel loopbacks are available on physical ports 0 through 59 and port 63, They are controlled as follows:

- **EPLs** — See EPL_CFG_A[i].*Active[0..3]* (Section 11.3.3.5).
- **PCIE** — See PCIE_CTRL_EXT[i].*SwitchLoopback* (Section 11.27.2.2).
- **Tunneling Engine** — See TE_CFG[i].*SwitchLoopbackDisable* (Section 11.31.4.14).

If a physical port is in parallel loopback, packets from the Ethernet switch go back to the switch, and the packets from the ports go back to the ports.

For the PCI-Express interfaces 0 through 3 (PEPs 0 through 7), the loopbacks are controllable per Quad Port Channel, and are asserted if both PEPs of a host interface pair request it. As an example, if both PCIE_CTRL_EXT[0].*SwitchLoopback* and PCIE_CTRL_EXT[1].*SwitchLoopback* are enabled, the entire set of 4 channels of the quad facing the PEP #0/PEP #1 host interface are in loopback. For the last interface, if PCIE_CTRL_EXT[8].*SwitchLoopback* is enabled, the physical port 63 is in loopback.

For the tunneling engine, only the parallel switch loopback is available, as the TE cannot generate packets by itself. The TE also has one control for all four channels, and thus the entire Quad Port Channel is in loopback if the TE_CFG[i].*SwitchLoopbackDisable* is set to 0b (thus enabling the loopback).

For Ethernet, each channel of a quad can be placed in loopback individually. When used, please note that the port loopback (from network back to network) has very little buffering. So, if the clock of the link partner (which sends the packets) is higher than the reference clock for the current device, and the link partner sends packets at full line rate with minimum inter-frame gaps, there will be packets dropped due to systematic data accumulation. Therefore, when using the parallel port loopbacks, ensure the link partner is not transmitting at full wire speed or transmitting jumbo packets.

10.2.1.3 Serial Loopbacks

The serial loopbacks are controllable per SerDes and loop the data transmitted serially to the network back to the serial receiver, ignoring the data actually received from the network. This loopback can be used to test functionality of the interface without actually using an external loopback.

10.2.1.4 Packet Test Interface

The Packet Test Interface is a low-level interface in the FIBM unit which allows an external device to generate and receive packets using register peeks and pokes. The interface is slow as it requires the software to perform multiple register accesses for each 32-bits of packet data sent. The interface is flexible and allows any kind of packet to be generated or received.

Normally, the intended usage of this interface is to enable the software to send one or more packets to the switch with the switch pre-configured in such a way that the packets generated circulate through the loopbacks at very high speed, then reconfigure the switch at a later time to reroute the packets back to the PTI interface for checking.

The registers available are:

- TX (Toward switch)
  - PTI_TX_CTRL — Packet Test Interface Tx
  - PTI_TX_DATA0 — Packet Test Interface Data
  - PTI_TX_DATA1 — Packet Test Interface Data
  - PTI_TX_CNT — Packet Test Interface Statistics
- RX (From switch)
  - PTI_RX_CTRL — Packet Test Interface Rx
  - PTI_RX_DATA0 — Packet Test Interface Data
  - PTI_RX_DATA1 — Packet Test Interface Data
  - PTI_RX_CNT — Packet Test Interface Statistics

For sending a packet:

```
// For each 64b of data
foreach w64 (@words)
    PTI_TX_DATA0 = w64 & 0xFFFFFFFF;
    PTI_TX_DATA1 = w64>>32 & 0xFFFFFFFF;
    eof = 0;
    info = 0;
    n = 8;
    // If last word
    if ( lastWord )
    {
      n = frameLength & 7;
      if ( n == 0 ) n = 8;
      eof = 1;
      // use other codes 1,2,3 for generating bad packets
      info = 0;
    }
    // Write as one access
    PTI_TX_CTRL = {Len=>n,TxValid=>1,Mark=>eof,Info=>info};
    // Poll to check if data is sent
    while (PTI_TX_CTRL.TxEnable == 1);
```

For receiving one packet:

```
// For each 64b of data
eof = 0;
frameLength = 0;
PTI_RX_CTRL.ContinuousDrain = 0;
@words = ();
while (!eof)
{
    // Poll to check if data is received
    while ( PTI_RX_CTRL.RxValid == 0 );
    // Recover data
    w64 = PTI_RX_DATA1;
    w64 = (w64 << 32) + PTI_RX_DATA0;
    eof = PTI_RX_CTRL.Mark == 3;
    err = PTI_RX_CTRL.Info;
    frameLength += PTI_RX_CTRL.Len;
    push (@words,w64);
    // Clear word (self clear)
    PTI_TX_CTRL.RxEnable == 1;
}
// packets is in vector @words
// length is in frameLength
// err indicates if the packet is valid (0) or not.
```

10.3 Test Interfaces

The FM10000 supports three test interfaces:

- External Bus Interface (EBI) (70 pins)
- Scan test interface (73 pins)
- JTAG interface

The External Bus and Scan interfaces uses the same pins, and thus only one interface is active at a time. They are both sharing the same pins except for a single pin, TEST_MODE, which determines the mode of operation for that module.

Table 10-1 shows the pin mapping:

Table 10-1 Test Interface Pin Mapping

TEST_MODE=1	TEST_MODE=0
SCAN_OUT[0..31]	DATA[0..31]
SCAN_IN[0..23]	ADDR[2..25]
SCAN_IN[24..27]	n/a
SCAN_IN[28]	CS_N
SCAN_IN[29]	AS_N
SCAN_IN[30]	RW_N
SCAN_IN[31]	DTACK_N
SCAN_CLK[31]	EBI_CLK
SCAN_EN[0..3]	n/a
SCAN_MODE	n/a

10.3.1 External Bus Interface (EBI)

The EBI is available as a direct method to access internal registers of the device without requiring a complex interface.

Note: This interface is reserved for Intel, not intended for normal usage, and might be removed in future versions of this product.

The EBI supports the following signals:

- ADDR
- DATA
- AS_N
- CS_N
- RW_N
- DTACK_N
- EBI_CLK

The maximum EBI_CLK rate is 33 MHz.

10.3.1.1 EBI Timing Diagram

Figure 10-2 shows the EBI timing diagram.

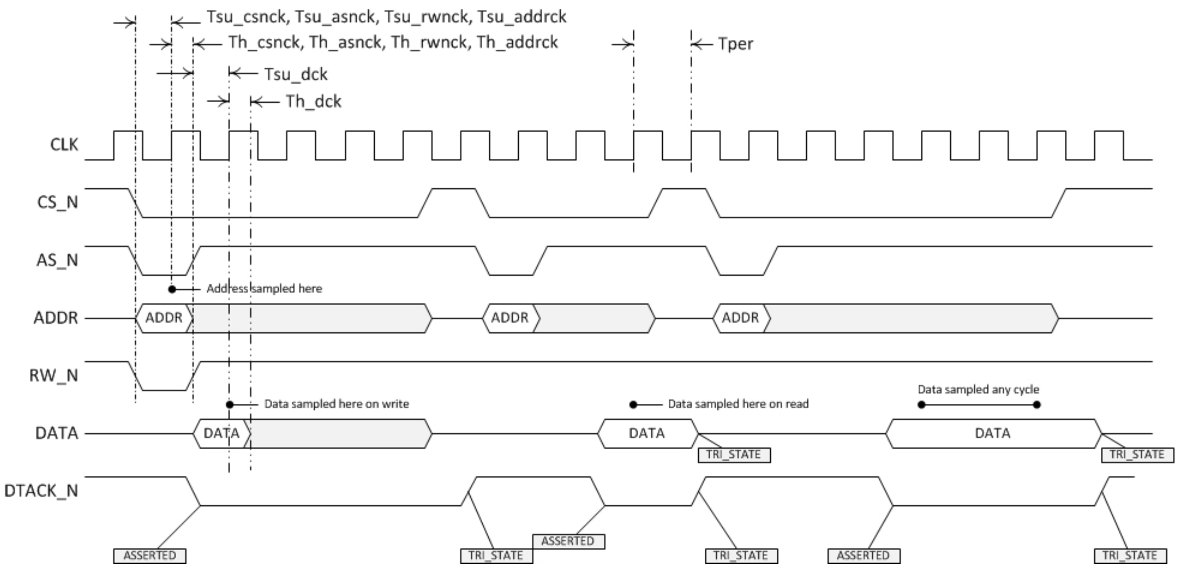


Figure 10-2 EBI Timing Diagram

10.3.1.2 EBI Timing Parameters

Table 10-2 EBI Timing Parameters

Parameter	Symbol	Min	Typical[1]	Max	Units
Setup CS_N to Clock	Tsu_csnck		7		ns
Hold CS_N to Clock	Th_csnck		7		ns
Setup AS_N to Clock	Tsu_asnck		7		ns
Hold AS_N to Clock	Th_asnck		7		ns
Setup RW_N to Clock	Tsu_rwnck		7		ns
Hold RW_N to Clock	Th_rwnck		7		ns
Setup ADDR[23:0] to Clock	Tsu_addrck		7		ns
Hold ADDR[23:0] to Clock	Th_addrck		7		ns
Setup DATA[31:0] to Clock	Tsu_dck		7		ns
Hold DATA[31:0] to Clock	Th_dck		7		ns
Hold DTACK_N to CS_N de-assertion	Th_dtack		0		ns
Clock Period[2]	Tper	30			ns

1. The EBI is intended as a test interface only. The timings given have been simulated and will work with ample margin. They are not optimized, and there is not a plan to do so. Thus, only "Typical" numbers are given.
2. Because the EBI is a test interface only, performance of the EBI is not guaranteed at clock periods less than 30 ns (33 MHz).

10.3.1.3 EBI Protocol

The external bus interface follows a simple protocol:

- Management access starts when CS_N and AS_N are both asserted (low) at rising edge of the clock. The ADDR and RW_N signals are also sampled on that same clock edge. The AS_N, ADDR and RW_N are ignored after the first cycle.
  - RW_N=1 is READ and RW_N=0 is WRITE
- Management access terminates when CS_N is detected de-asserted (high) at rising edge of clock.
  - Note that the host does not de-assert CS_N until a DTACK_N is asserted.
- Successive management accesses must be separate by at least one CS_N high cycle.
- For a write cycle, the data is always sampled on the next cycle after CS_N & AS_N are detected asserted. Furthermore, the switch has a small write FIFO to store a few transactions. The device asserts a DTACK_N signal on the next cycle if the write FIFO is not full. Otherwise the DTACK_N is delayed until this FIFO has room to store the data. Minimum write time is thus 2 clock cycles (excluding idle cycle). The DTACK_N remains actively asserted as long as CS_N is low and tri-stated when CS_N is detected de-asserted.
- For a read cycle, the switch delays asserting DATA and DTACK_N until data is available, and drives both signals on the same clock edge. Minimum read time is 3 clock cycles (excluding idle cycle). The DATA and DTACK_N remain actively asserted as long as CS_N is low. DTACK and DATA are tri-stated when CS_N is detected de-asserted.

  The access time from cycle start to DTACK_N asserted is variable depending on the register accessed and the traffic load in the switch. The worst-case is estimated to be 5 μs.

10.3.2 Scan Interface

The scan interface is not detailed in this document.

10.3.3 JTAG Interface

The JTAG controller is compliant with the IEEE 1149.1-2001 specification. The JTAG provides basic external chip debug features:

- Access to an identification register.
- Access to the boundary scan.
- Access to the internal scan chains.
- Ability to Clamp and HighZ all outputs (except SerDes).

The maximum frequency of operation is 40 MHz.

The Supported operations of these registers are:

- **Load IR** (instruction register)
- **Capture** — Initializes/captures/freezes value of register
- **Shift** — Serially shifts in/out value into/out of register.

- **Update** — Validates the contents of the register. (i.e., logic can now use the new value for its internal operation).

The JTAG reset domain is separate and independent from the chip reset domain.

10.3.3.1 Tap Controller

The tap controller is a finite state machine of 16 states controlled by the 5-pin JTAG interface. It is defined by IEEE 1149.1-2001. Instruction Register.

Supported JTAG instructions are shown in Table 10-3.

Table 10-3 Tap Controller JTAG Instructions

Instruction	Code (6b)	Description
IDCODE	x01	Selects the identification register.
SAMPLE/PRELOAD	x02	Select the boundary scan register. Sample input pins to input boundary scan register, pre-load the output boundary scan register.
EXTEST	x03	Select the boundary scan register. Output boundary scan register cells drive the covered output pins. Input boundary cell registers sample the input pins.
HIGHZ	x06	Selects the bypass register and sets all covered output pins to high impedance.
CLAMP	x06	Forces a known value on the outputs, but uses the bypass register to shorten scan length.
BYPASS	x3F	Bypass

The boundary scan register is a 537-bit deep shift register. Refer to the BSDL description file for pin assignment.

11.0 Register Definitions

11.1 Definitions

Registers types are the following:

RW	Read/Write
SRW	Read/Write (Service reset)
RO	Read-Only (This register cannot be programmed by software. Typically reports a status.)
CW	Clear-on-Write (Any value written clears the register.)
CW1	Clear-on-Write-1 (Writing 1b to any bit clears that bit. Writing 0b has no effect.)
SW1	Set-on-Write-1 (Writing 1b to any bit sets that bit. Writing 0b has no effect.)
CR	Clear-on-Read (Reading the register clears the register.)
RSV	Reserved (For upward compatibility. Always write as zero, ignore on read.)
WO	Write-Only (The value written cannot be read back.)

A "v" prefix indicates volatile (changed by hardware).

11.1.1 Terminology

IETF Geneve (GENEVE) is one of the three supported tunnels in the FM10000 (along with NVGRE and VXLAN). GENEVE is also known as NGE. GENEVE and NGE are used throughout this section.

11.2 Register Set Summary

Table 11-1 Register Set Summary

Block	Base Address	Base Size	Brief Description	Link
EPL	0x0E0000	2^15=0x8000	Ethernet Port Logic registers. (reserved 24 x 1 K => 24 KW)	427
PARSER	0xCF0000	2^16=0x10000	Parser register set	499
FFU	0xC00000	2^19=0x80000	FFU registers (reserved 512 KW)	505
FFU_MAP	0xDA0000	2^16=0x10000	FFU_MAP registers (reserved 8 KW)	512
EACL	0xDB0000	2^16=0x10000	Egress ACL registers (reserved 4 KW)	516
ARP	0xCC0000	2^17=0x20000	ARP control and table (128 KW)	518
POLICER_APPLY	0xD60000	2^16=0x10000	Policer registers (reserved 64 KW)	520
POLICER_USAGE	0xE40000	2^17=0x20000	Policer registers (reserved 64KW)	522
L2LOOKUP	0xC80000	2^18=0x40000	Layer 2 Lookup registers (reserved 256KW)	525
L2LOOKUP_TCN	0xE70000	2^15=0x8000	Layer 2 Lookup registers (reserved 64KW)	530
HANDLER	0xD50000	2^16=0x10000	Frame Handler generic registers (reserved 4KW)	533
HANDLER_TAIL	0xE30000	2^16=0x10000	Frame Handler generic registers (reserved 4KW)	544
GLORT	0xCE0000	2^16=0x10000	GLORT control registers and GLORT table (reserved 16KW)	550
LAG	0xD90000	2^16=0x10000	Link Aggregation registers	553
TRIG_APPLY	0xD70000	2^16=0x10000	Triggers registers (reserved 4KW)	554
TRIG_USAGE	0xE78000	2^15=0x8000	Triggers Usage registers	564
CM_APPLY	0xD40000	2^16=0x10000	Congestion Management registers (reserved 64KW)	565
CM_USAGE	0xE60000	2^16=0x10000	Congestion Management registers (reserved 64KW)	567
MOD	0xE80000	2^19=0x80000	Egress Modification registers	578
RX_STATS	0xE00000	2^17=0x20000	Ingress statistics counters (reserved 256KW)	591
SCHED	0xF00000	2^20=0x100000	Scheduler registers	594
FIBM	0x008000	2^15=0x8000	FIBM registers (reserved 2x4KW)	608
MGMT	0x000000	2^15=0x8000	Management registers (reserved 16KW)	619
PORTS_MGMT	0x0E8000	2^15=0x8000	Ports Management registers (reserved 16KW)	659
PCIE_PF	0x100000	2^20=0x100000	PCI Express registers - Physical Function	667
PCIE_VF	0x200000	2^20=0x100000	PCI Express registers - Virtual Function	717
PCIE_CFG	0x120000	2^17=0x20000	PCI Express Configuration registers - Physical Function	735
PCIE_CFG_VF	0x130000	2^16=0x10000	PCI Express Configuration registers - Virtual Function	775
TE	0xA00000	2^21=0x200000	Tunneling Engine register set	798

11.3 EPL Registers Description

11.3.1 EPL Map

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
EPL_IP[0..8]	Base + 0x400*i + 0x300	32	EPL interrupt pending register.
EPL_ERROR_IP[0..8]	Base + 0x400*i + 0x301	32	
EPL_ERROR_IM[0..8]	Base + 0x400*i + 0x302	32	
EPL_BIST_STATUS[0..8]	Base + 0x400*i + 0x303	32	
EPL_CFG_A[0..8]	Base + 0x400*i + 0x304	32	EPL unit configuration.
EPL_CFG_B[0..8]	Base + 0x400*i + 0x305	32	Port to SerDes mapping.
EPL_LED_STATUS[0..8]	Base + 0x400*i + 0x306	32	Port status information intended to be read by the LED handler.
EPL_FIFO_ERROR_STATUS[0..8]	Base + 0x400*i + 0x307	32	
EPL_TX_FIFO_RD_STATUS[0..8]	Base + 0x400*i + 0x308	32	Status for lane transmit FIFOs.
EPL_TX_FIFO_WR_STATUS[0..8]	Base + 0x400*i + 0x309	32	Status for lane transmit FIFOs.
EPL_TX_FIFO_A_STATUS[0..8]	Base + 0x400*i + 0x30A	32	Status for lane transmit FIFOs.
EPL_TX_FIFO_B_STATUS[0..8]	Base + 0x400*i + 0x30B	32	Status for lane transmit FIFOs.
EPL_TX_FIFO_C_STATUS[0..8]	Base + 0x400*i + 0x30C	32	Status for lane transmit FIFOs.
EPL_TX_FIFO_D_STATUS[0..8]	Base + 0x400*i + 0x30D	32	Status for lane transmit FIFOs.
EPL_RX_FIFO_RD_STATUS[0..8]	Base + 0x400*i + 0x30E	32	Status for lane receive FIFOs.
EPL_RX_FIFO_WR_STATUS[0..8]	Base + 0x400*i + 0x30F	32	Status for lane receive FIFOs.
EPL_RX_FIFO_A_STATUS[0..8]	Base + 0x400*i + 0x310	32	Status for lane receive FIFOs.
EPL_RX_FIFO_B_STATUS[0..8]	Base + 0x400*i + 0x311	32	Status for lane receive FIFOs.
EPL_RX_FIFO_C_STATUS[0..8]	Base + 0x400*i + 0x312	32	Status for lane receive FIFOs.
EPL_RX_FIFO_D_STATUS[0..8]	Base + 0x400*i + 0x313	32	Status for lane receive FIFOs.
PCS_ML_BASER_CFG[0..8]	Base + 0x400*i + 0x314	32	

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map [Continued]

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
PCS_ML_BASER_RX_STATUS[0..8]	Base + 0x400*i + 0x315	32	PCS40GBASER and PCS100GBASER status.
PCS_100GBASER_BIP_0[0..8]	Base + 0x400*i + 0x317	32	
PCS_100GBASER_BIP_1[0..8]	Base + 0x400*i + 0x318	32	
PCS_100GBASER_BIP_2[0..8]	Base + 0x400*i + 0x319	32	
PCS_100GBASER_BIP_3[0..8]	Base + 0x400*i + 0x31A	32	
PCS_100GBASER_BIP_4[0..8]	Base + 0x400*i + 0x31B	32	
PCS_100GBASER_BIP_5[0..8]	Base + 0x400*i + 0x31C	32	
PCS_100GBASER_BIP_6[0..8]	Base + 0x400*i + 0x31D	32	
PCS_100GBASER_BIP_7[0..8]	Base + 0x400*i + 0x31E	32	
PCS_100GBASER_BIP_8[0..8]	Base + 0x400*i + 0x31F	32	
PCS_100GBASER_BIP_9[0..8]	Base + 0x400*i + 0x320	32	
PCS_100GBASER_BIP_10[0..8]	Base + 0x400*i + 0x321	32	
PCS_100GBASER_BIP_11[0..8]	Base + 0x400*i + 0x322	32	
PCS_100GBASER_BIP_12[0..8]	Base + 0x400*i + 0x323	32	
PCS_100GBASER_BIP_13[0..8]	Base + 0x400*i + 0x324	32	
PCS_100GBASER_BIP_14[0..8]	Base + 0x400*i + 0x325	32	
PCS_100GBASER_BIP_15[0..8]	Base + 0x400*i + 0x326	32	
PCS_100GBASER_BIP_16[0..8]	Base + 0x400*i + 0x327	32	
PCS_100GBASER_BIP_17[0..8]	Base + 0x400*i + 0x328	32	
PCS_100GBASER_BIP_18[0..8]	Base + 0x400*i + 0x329	32	
PCS_100GBASER_BIP_19[0..8]	Base + 0x400*i + 0x32A	32	
PCS_100GBASER_BLOCK_LOCK[0..8]	Base + 0x400*i + 0x32B	32	
PCS_100GBASER_AMPS_LOCK[0..8]	Base + 0x400*i + 0x32C	32	
RS_FEC_UNCORRECTED[0..8]	Base + 0x400*i + 0x33D	32	

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map [Continued]

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
RS_FEC_CFG[0..8]	Base + 0x400*i + 0x32E	32	
RS_FEC_STATUS[0..8]	Base + 0x400*i + 0x330	32	
EPL_SYSTIME[0..8]	Base + 0x400*i + 0x331	32	System Time.
EPL_SYSTIME0[0..8]	Base + 0x400*i + 0x332	32	System Time Initial Value.
EPL_SYSTIME_CFG[0..8]	Base + 0x400*i + 0x333	32	System Time Configuration.
PORT_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x0	32	Port status information.
AN_IM[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x1	32	Auto-Negotiation Interrupt Mask.
LINK_IM[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2	32	Link Interrupt Mask.
AN_IP[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x3	32	Auto-Negotiation Interrupt Pending.
LINK_IP[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4	32	Link Interrupt Pending.
MP_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x6	32	Multi purpose register used for Auto-Negotiation pages to transmit.
AN_37_PAGE_RX[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x8	32	Multi purpose register used for Clause 37 Auto-Negotiation pages received.
AN_73_PAGE_RX[0..8][0..3]	Base + 0x400*j + 0x80*i + 0xA	32	Multi purpose register used for Clause 73 Auto-Negotiation pages received.
LINK_RULES[0..8][0..3]	Base + 0x400*j + 0x80*i + 0xC	32	
MAC_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x10	32	MAC and Reconciliation Sublayer configuration.
TX_SEQUENCE[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x1A	32	Programmable Sequence or FSIG column.
RX_SEQUENCE[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x1C	32	Last received Sequence or FSIG column.
MAC_1588_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x1E	32	IEEE 1588 EGRESS Time Tagging Status.
WAKE_ERROR_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x20	32	
MAC_OVERSIZE_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x21	32	Oversize Counter. Counts well formed frames that are greater than max length.
MAC_JABBER_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x22	32	Jabber Counter. Counts badly formed frames that are greater than max length.
MAC_UNDERSIZE_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x23	32	Undersize Counter. Counts well formed frames that are less than min length.

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map [Continued]

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
MAC_RUNT_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x24	32	Runt Counter. Counts badly formed frames that are less than min length.
MAC_OVERRUN_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x25	32	Overrun Event Counter. Counts frames lost due to switch congestion.
MAC_UNDERRUN_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x26	32	Underrun Event Counter. Counts the number of frames lost because data was not available from the switch.
MAC_CODE_ERROR_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x27	32	Code Error Counter. Counts the number of /E/ decoded bytes are received.
EPL_TX_FRAME_ERROR_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x28	32	Frame error counter. Counts the number of frames received from the switch marked as error.
MAC_LINK_COUNTER[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x29	32	Count link state change events.
PCS_1000BASEX_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2A	32	PCS 1000BASE-X configuration.
PCS_1000BASEX_RX_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2B	32	PCS 1000BASE-X receive status.
PCS_1000BASEX_TX_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2C	32	PCS 1000BASE-X transmit status.
PCS_10GBASER_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2D	32	PCS 10GBASE-R (Clause 49) configuration.
PCS_10GBASER_RX_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2E	32	PCS 10GBASE-R (Clause 49) receive status.
PCS_10GBASER_TX_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x2F	32	PCS 10GBASE-R (Clause 49) transmit status.
AN_37_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x30	32	Auto-Negotiation (Clause 37/SGMII) configuration.
AN_37_TIMER_CFG[0..8][0..3] *(alias of AN_73_TIMER_CFG)*	Base + 0x400*j + 0x80*i + 0x34	32	Auto-Negotiation (Clause 37) timer configuration.
AN_37_BASE_PAGE_TX[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	Auto-Negotiation (Clause 37) transmit base page configuration.
AN_37_BASE_PAGE_RX[0..8][0..3] *(alias of AN_37_PAGE_RX)*	Base + 0x400*j + 0x80*i + 0x8	32	Auto-Negotiation (Clause 37) receive base page status.
AN_37_NEXT_PAGE_TX[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	Auto-Negotiation (Clause 37) transmit next page status.
AN_37_NEXT_PAGE_RX[0..8][0..3] *(alias of AN_37_PAGE_RX)*	Base + 0x400*j + 0x80*i + 0x8	32	Auto-Negotiation (Clause 37) receive next page status.
SGMII_AN_TIMER_CFG[0..8][0..3] *(alias of AN_73_TIMER_CFG)*	Base + 0x400*j + 0x80*i + 0x34	32	SGMII Auto-Negotiation timer configuration.
SGMII_AN_TX_CONFIG[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	SGMII Auto-Negotiation transmit code word configuration.
SGMII_AN_TX_CONFIG_LOOPBACK[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	SGMII Auto-Negotiation transmit code word configuration, switch using PHY encodings for loopback test.
SGMII_AN_RX_CONFIG[0..8][0..3] *(alias of AN_37_PAGE_RX)*	Base + 0x400*j + 0x80*i + 0x8	32	SGMII Auto-Negotiation receive code word status.

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map [Continued]

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
AN_37_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x32	32	Auto-Negotiation (Clause 37/SGMII) status.
AN_73_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x33	32	Auto-Negotiation (Clause 73) configuration.
AN_73_TIMER_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x34	32	Auto-Negotiation (Clause 73) timer configuration.
AN_73_BASE_PAGE_TX[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	Auto-Negotiation (Clause 73) base page transmit configuration.
AN_73_BASE_PAGE_RX[0..8][0..3] *(alias of AN_73_PAGE_RX)*	Base + 0x400*j + 0x80*i + 0xA	32	Auto-Negotiation (Clause 73) base page receive status.
AN_73_NEXT_PAGE_TX[0..8][0..3] *(alias of MP_CFG)*	Base + 0x400*j + 0x80*i + 0x6	32	Auto-Negotiation (Clause 73) next page transmit configuration.
AN_73_NEXT_PAGE_RX[0..8][0..3] *(alias of AN_73_PAGE_RX)*	Base + 0x400*j + 0x80*i + 0xA	32	Auto-Negotiation (Clause 73) next page receive status.
AN_73_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x36	32	Auto-Negotiation (Clause 73) receive status.
AN_73_TX_LCW[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x38	32	Auto-Negotiation (Clause 73) transmit link code 32 word. Intended for DEBUG only.
AN_73_RX_LCW[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x3A	32	Auto-Negotiation (Clause 73) receive ability match word. Intended for DEBUG only.
PCSL_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x3C	32	PCS lane configuration.
MP_EEE_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x40	32	Multi-Purpose Energy Efficient Ethernet configuration.
PCS_1000BASEX_EEE_CFG[0..8][0..3] *(alias of MP_EEE_CFG)*	Base + 0x400*j + 0x80*i + 0x40	32	
PCS_10GBASER_EEE_CFG[0..8][0..3] *(alias of MP_EEE_CFG)*	Base + 0x400*j + 0x80*i + 0x40	32	
MP_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x44	32	Multi-purpose status.
PCS_40GBASER_RX_BIP_STATUS[0..8][0..3] *(alias of MP_STATUS)*	Base + 0x400*j + 0x80*i + 0x44	32	PCS40GBASER Bit Interleaved Parity Error Status.
PCS_10GBASER_RX_BER_STATUS[0..8][0..3] *(alias of MP_STATUS)*	Base + 0x400*j + 0x80*i + 0x44	32	PCS10GBASER Bit Error Status.
DISPARITY_ERROR_8B10B[0..8][0..3] *(alias of MP_STATUS)*	Base + 0x400*j + 0x80*i + 0x44	32	PCS1000BASEX Disparity Error Status.
LANE_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x45	32	Lane configuration.
LANE_SERDES_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x46	32	SerDes configuration.
LANE_ENERGY_DETECT_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x47	32	EEE energy detect configuration.
LANE_ACTIVITY_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x48	32	Lane activity monitor configuration.
LANE_SIGNAL_DETECT_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x49	32	Signal detect configuration.

Table 11-2 Ethernet Port Logic Registers (reserved 24 x 1 K => 24 KW) Map [Continued]

Name	Address (Base = 0x0E0000)	Atomicity	Brief Description
LANE_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4A	32	Lane status.
LANE_SERDES_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4B	32	SerDes status.
SERDES_IM[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4C	32	SerDes Interrupt Mask.
SERDES_IP[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4D	32	SerDes Interrupt Pending.
LANE_ANALOG_IM[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4E	32	Interrupt masks for LANE_ANALOG_IP.
LANE_ANALOG_IP[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x4F	32	SerDes analog_to_core signals exposed as interrupts.
LANE_SAI_CFG[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x50	32	Lane SerDes Access Interface Configuration.
LANE_SAI_STATUS[0..8][0..3]	Base + 0x400*j + 0x80*i + 0x52	32	Lane SerDes Access Interface Status.

11.3.2 EPL Enumerated Data Types

Table 11-3 EPL Enumerated Data Types

Name	Width	Description
RefClockSource	1	Reference clock sources are daisy chained through the EPL lanes. Within each lane the reference clock can either be passed through unchanged or the local RefClockSource 1 divided recovered clock signal can be selected dependent on *RefClockSource* as follows: 0b = PASS — Select divided clock from the previous stage in chain 1b = LOCAL — Select local divided recovered receive clock If all EPL lanes are in pass-through, no reference clock is provided.
Result	2	Indicate result of last exchange. The result are: 00b = RUNNING 01b = PASS 10b = FAIL 11b = TIMEOUT

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
QplMode	3	EPL has four PORT connections to the switch fabric and four LANE connections to the physical media. In single-lane mode, a PORT is connected to the corresponding LANE. In multi-lane mode, all four LANES connect to all four PORTS to provide the capacity for a high-speed channel. In this case, a single-lane PCS type or AN73 auto-negotiation uses the LANE associated with the single PORT.Key: • XX no lanes allocated to the associated port • L1 one lane allocated to the associated port • L4 four lane allocated to the associated port Management configuration of the MAC for 40GBASE-R and 100GBASE-R modes are configured by the selected channel resources based on QplMode. • L4_XX_XX_XX configured with registers at stride offset 0x080 x 0 • XX_L4_XX_XX configured with registers at stride offset 0x080 x 1 • XX_XX_L4_XX configured with registers at stride offset 0x080 x 2 • XX_XX_XX_L4 configured with registers at stride offset 0x080 x 3 The enumeration below provides the association with PORT0_PORT1_PORT2_PORT3 respectively: 000b = XX_XX_XX_XX — No lanes allocated. 001b = L1_L1_L1_L1 — One lane allocated to each port. 010b = XX_XX_XX_L4 — Four lanes; single lane uses PORT3 and Lane 3. 011b = XX_XX_L4_XX — Four lanes; single lane uses PORT2 and Lane 2. 100b = XX_L4_XX_XX — Four lanes; single lane uses PORT1 and Lane 1. 101b = L4_XX_XX_XX — Four lanes; single lane uses PORT0 and Lane 0. All other values are reserved.
TxOset	3	000b = C 001b = D 010b = I 011b = R 100b = S 101b = T 110b = V 111b = LI
TxOsetState	4	0000b = TX_TEST_XMIT 0001b = CONFIGURATION 0010b = IDLE 0011b = XMIT_DATA 0100b = START_OF_PACKET 0101b = ALIGN_ERR_START 0110b = START_ERROR 0111b = TX_DATA_ERROR 1000b = TX_DATA 1001b = END_OF_PACKET_EXT 1010b = CARRIER_EXTEND 1011b = EXTEND_BY_1 1100b = END_OF_PACKET_NOEXT 1101b = EPD2_NOEXT 1110b = EPD3 1111b = XMIT_LPIDLE

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
TxCgState	4	0000b = CONFIGURATION_C1A 0001b = CONFIGURATION_C1B 0010b = CONFIGURATION_C1C 0011b = CONFIGURATION_C1D 0100b = CONFIGURATION_C2A 0101b = CONFIGURATION_C2B 0110b = CONFIGURATION_C2C 0111b = CONFIGURATION_C2D 1000b = SPECIAL_GO 1001b = Reserved. 1010b = DATA_GO 1011b = IDLE_DISPARITY_WRONG 1100b = IDLE_I1B 1101b = IDLE_DISPARITY_OK 1110b = IDLE_I2B 1111b = Reserved.
CheckEnd	3	000b = NOT_END 001b = K28_5__D21_5_OR_D2_2__D0_0 010b = K28_5__D__K28_5 011b = R__R__K28_5 100b = R__R__R 101b = R__R__S 110b = T__R__K28_5 111b = T__R__R
Pcs1000basexRxState	5	00000b = EARLY_END 00001b = EARLY_END_EXT 00010b = EXTEND_ERR 00011b = FALSE_CARRIER 00100b = IDLE_D 00101b = LINK_FAILED 00110b = PACKET_BURST_RRS 00111b = RX_CB 01000b = RX_CC 01001b = RX_CD 01010b = RX_DATA 01011b = RX_DATA_ERROR 01100b = RX_INVALID 01101b = RX_K 01110b = START_OF_PACKET 01111b = TRI_OR_RRI 10000b = TRR_OR_EXTEND 10001b = WAIT_FOR_K 10010b = LP_IDLE_D 10011b = LPI_K 10100b = RX_NOISE 10101b = RX_QUIET 10110b = RX_WAKE 10111b = RX_WTF 11000b = RX_LINK_FAIL All other values are reserved.
Rudi	2	00b = IGNORE 01b = IDLE 10b = CONFIGURATION 11b = INVALID

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
An37State	4	0000b = ENABLE 0001b = RESTART 0010b = DISABLE_LINK_OK 0011b = ABILITY_DETECT 0100b = ACKNOWLEDGE_DETECT 0101b = COMPLETE_ACKNOWLEDGE 0110b = IDLE_DETECT 0111b = LINK_OK 1000b = NEXT_PAGE_WAIT All other values are reserved.
PcsSel	4	0000b = DISABLE 0001b = AN_73 0010b = SGMII_10 0011b = SGMII_100 0100b = SGMII_1000 0101b = 1000BASEX 0110b = 10GBASER 0111b = 40GBASER 1000b = 100GBASER All other values are reserved.
LinkFaultSignal	3	Stop sending traffic. 000b = OK — No Fault, the link is up. 001b = LOCAL_FAULT — Fault detected by PHY (could be an attached PHY) link is down. 010b = REMOTE_FAULT — Fault detected by the remote RS, link is down. Not applicable for 1000BASE-X modes. 011b = LINK_INTERRUPTION — Sequence detected by the PHY. 100b = RESERVED — A code for the set of reserved sequences. All other values are reserved.
LinkFault	3	000b = OK 001b = LOCAL_FAULT 010b = REMOTE_FAULT 011b = LINK_INTERRUPTION 100b = NOT_SEQUENCE 101b = USER_VAL 110b = Reserved. 111b = Reserved.
TxMacDrainMode	2	00b = DRAIN_DRAIN — Drain packets without forwarding to link partner if Link is NOT_OK or OK. 01b = DRAIN_NORMAL — Drain if Link is Not OK, forward traffic to link partner if link is OK. 10b = HOLD_NORMAL — Hold if Link is Not OK, forward traffic to link partner if link is OK. 11b = HOLD_HOLD — Hold if Link is NOT OK or OK.
TxMacFaultMode	3	000b = NORMAL — Fault controlled by Figure 46-9: Link Fault Signaling State Diagram. 001b = FORCE_IDLE — Force Idles on transmit stream. 010b = FORCE_LOCAL_FAULT — Force Local Fault on transmit stream. 011b = FORCE_REMOTE_FAULT — Force Remote Fault on transmit stream. 100b = FORCE_LINK_INTERRUPTION — Force Link Interruption. 101b = FORCE_OK — Force Local Fault state to OK for transmit stream. 110b = FORCE_ERROR — Force Errors on transmit stream. 111b = FORCE_USER_VAL — Force User Column on transmit stream.

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
TxMacFcsMode	3	000b = PASSTHRU 001b = PASSTHRU_CHECK 010b = REPLACE_GOOD 011b = REPLACE_BAD 100b = REPLACE_NORMAL All other values are reserved.
TPgFirstWordFillType	1	0b = FRAME_COUNT 1b = FIXED
LfSignalingState	2	00b = INIT 01b = COUNT 10b = FAULT 11b = Reserved.
TgbrLpiRxMode	1	0b = DATA 1b = QUIET
TxPcs10gbaserState	3	000b = INIT 001b = C 010b = D 011b = T 100b = E 101b = LI All other values are reserved.
Pcs10gbaserTxMode	2	00b = DATA 01b = QUIET 10b = ALERT 11b = Reserved.
PreambleMode	1	0b = STRIP — On ingress preamble stripped, and is not forwarded to the switch fabric. On egress preamble is inserted at the start of the frame. 1b = PASSTHRU — On ingress preamble is forwarded to the switch fabric, /START/ replaced by the data byte MAC_CFG.*StartCharD*. On egress, preamble is forwarded from the switch, and the first data byte is replaced with /START/ control.
TickTimeScale	4	0000b = ON 0001b = 50 ns 0010b = 1 µs 0011b = 10 µs 0100b = 100 µs 0101b= 1 ms 0110b = 10 ms 0111b = 100 ms 1000b = 1 s 1001b = OFF All other values are reserved.
EeeTimeScale	2	00b = 50 ns 01b = 1 µs 10b = 10 µs 11b = 100 µs
Override	2	00b = NORMAL 01b = FORCE_GOOD 10b = FORCE_BAD 11b = Reserved.

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
ControlLane	3	000b = DISABLE 001b = AN_73 010b = 1000BASEX 011b = 10GBASER 100b = 40GBASER 101b = 100GBASER All other values are reserved.
An73LinkControl	2	00b = DISABLE 01b = SCAN_FOR_CARRIER 10b = ENABLE 11b = Reserved.
An73State	3	000b = ENABLE 001b = TRANSMIT_DISABLE 010b = ABILITY_DETECT 011b = ACKNOWLEDGE_DETECT 100b = COMPLETE_ACKNOWLEDGE 101b = NEXT_PAGE_WAIT 110b = GOOD_CHECK 111b = GOOD
FecSyncState	3	000b = LOCK_INIT 001b = GET_BLOCK_1 010b = GET_BLOCK_2 011b = FIND_1ST 100b = COUNT_NEXT_COMP_2ND 101b = TWO_GOOD 110b = SLIP 111b = Reserved
FecAlignState	3	000b = LOSS_OF_ALIGNMENT 001b = DESKEW 010b = DESKEW_FAIL 011b = 3_BAD 100b = ALIGNED All other values are reserved.
Cl82AmSm	3	000b = LOCK_INIT 001b = RESET_CNT 010b = FIND_1ST 011b = TIMER_1 100b = Reserved 101b = Reserved 110b = TIMER_2 111b = SLIP
PopState	3	000b = DISABLED 001b = FREE_RUN 010b = DRAINING 011b = COUNTING 100b = WAITING 101b = JUST_ALIGNED 110b = ALIGNED 111b = Reserved
Cl36TxLpiState	2	00b = TX_AVTIVE 01b = TX_SLEEP 10b = TX_QUIET 11b = TX_REFRESH

Table 11-3 EPL Enumerated Data Types [Continued]

Name	Width	Description
Cl49TxLpiState	3	000b = TX_AVTIVE 001b = TX_SLEEP 010b = TX_QUIET 011b = TX_ALERT 100b = TX_WAKE All other values are reserved.
Cl49RxLpiState	3	000b = RX_AVTIVE 001b = RX_SLEEP 010b = NOISE 011b = RX_QUIET 100b = RX_WAKE 101b = RX_WTF 110b = RX_LINK_FAIL 111b = Reserved

11.3.3 EPL Registers

11.3.3.1 EPL_IP[0..8]

Address: 0x0E0000 + 0x400*i + 0x300
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AnPortInterrupt	3:0	vRO	0x0	Indicates an interrupt is pending or not for Auto-Negotiation status change from port 0 through 3. 0b = No interrupt pending. 1b = Interrupt pending. Check AN_IP register (Section 11.3.3.54) for the specific interrupt.
LinkPortInterrupt	7:4	vRO	0x0	Indicates an interrupt is pending or not for Link status change from port 0 through 3. 0b = No interrupt pending. 1b = Interrupt pending. Check LINK_IP register (Section 11.3.3.55) for the specific interrupt.
SerdesInterrupt	11:8	vRO	0x0	Indicates an interrupt is pending or not for SerDes interrupt A through D. 0b = No interrupt pending. 1b = Interrupt pending. Check SERDES_IP register (Section 11.3.3.116) for the specific interrupt.
ErrorInterrupt	12	vRO	0b	Indicates an interrupt is pending or not for register file correctable or uncorrectable errors. 0b = No interrupt pending. 1b = Interrupt pending. Check EPL_ERROR_IP register (Section 11.3.3.2) for the specific interrupt.
Reserved	31:13	RSV	0x0	Reserved.

11.3.3.2 EPL_ERROR_IP[0..8]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x301** **32** **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
JitterFifoUErrP0	0	CW1	0b	Jitter FIFO multi-bit error detected.
JitterFifoCErrP0	1	CW1	0b	Jitter FIFO error detected.
JitterFifoUErrP1	2	CW1	0b	Jitter FIFO multi-bit error detected.
JitterFifoCErrP1	3	CW1	0b	Jitter FIFO error detected.
JitterFifoUErrP2	4	CW1	0b	Jitter FIFO multi-bit error detected.
JitterFifoCErrP2	5	CW1	0b	Jitter FIFO error detected.
JitterFifoUErrP3	6	CW1	0b	Jitter FIFO multi-bit error detected.
JitterFifoCErrP3	7	CW1	0b	Jitter FIFO error detected.
RsSafFifoUErr	8	CW1	0b	RS FEC SAF FIFO parity error detected.
Reserved	31:9	RSV	0x0	Reserved.

11.3.3.3 EPL_ERROR_IM[0..8]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x302** **32** **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
JitterFifoUErrP0	0	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoCErrP0	1	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoUErrP1	2	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoCErrP1	3	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoUErrP2	4	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoCErrP2	5	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoUErrP3	6	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
JitterFifoCErrP3	7	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
RsSafFifoUErr	8	RW	0b	0b = Enable interrupt. 1b = Mask interrupt.
Reserved	31:9	RSV	0x0	Reserved.

11.3.3.4 EPL_BIST_STATUS[0..8]

Address: **0x0E0000 + 0x400*i + 0x303**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
JitterFifoBistPass	3:0	vRO	0x0	Jitter FIFO [D:A] BIST PASS status.
JitterFifoBistFail	7:4	vRO	0x0	Jitter FIFO [D:A] BIST FAIL status.
RsSafFifoBistPass	8	vRO	0b	RS SAF FIFO BIST PASS status.
RsSafFifoBistFail	9	vRO	0b	RS SAF FIFO BIST FAIL status.
Reserved	31:10	RSV	0x0	Reserved.

11.3.3.5 EPL_CFG_A[0..8]

Address: **0x0E0000 + 0x400*i + 0x304**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
SpeedUp	0	RW	0b	Particular timers which use a *TimeScale* can be configured as: Normal case: TimeUnit1us = 20 * TimeUnit50ns TimerTickTock = TimeUnit1us * 10**(TimeScale-2) for *TickTimeScale* values 2..8 (labeled 1 µs to 1 s) Speed Up case: TimeUnit1us = 4 * TimeUnit50ns TimerTickTock = TimeUnit1us * 2**(TimeScale-2) for *TickTimeScale* values 2..8 (labeled 1 µs to 1 s) 0b = Use normal options. 1b = Use speed up options.
Timeout	6:1	RW	0x27	Configuration for 50 ns timer. clock_freq = (system clock frequency)/2 (Hz) BasePeriod = 1.e9/(clock_freq (Hz))) (ns) RealTimeUnit = (EPL_CFG_A.Timeout+1)*BasePeriod (ns) TimeUnit50ns = RealTimeUnit
Active[0..3]	10:7	RW	0b	(4 x 1-bit) 0b = Standby (Loopback)— EPL Port Channels loopback receive to transmit (ingress to egress). Switch Port Channels loopback transmit to receive (egress to ingress). 1b = Active (Normal operation) — EPL Port Channels connected to the switch fabric. This field may not be written while traffic is flowing.
SkewTolerance	16:11	RW	0x6	The amount of skew the system can tolerate, configured dependent on mode and core clock frequency. SkewTolerance * (core_clock_period * 4) maximum skew tolerance (ns) For KR4 the specified maximum skew is 180 ns, while the maximum valid setting for the system is 190 ns.
Reserved	31:17	RSV	0x0	Reserved.

11.3.3.6 EPL_CFG_B[0..8]

May not be written while traffic is flowing.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x305** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Port0PcsSel	3:0	RW	0x0	Specifies the mode of operation for Port 0, for valid *QplMode* settings. Uses type *PcsSel* (see Table 11-3).
Port1PcsSel	7:4	RW	0x0	Specifies the mode of operation for Port 1, for valid *QplMode* settings. Uses type *PcsSel* (see Table 11-3).
Port2PcsSel	11:8	RW	0x0	Specifies the mode of operation for Port 2, for valid *QplMode* settings. Uses type *PcsSel* (see Table 11-3).
Port3PcsSel	15:12	RW	0x0	Specifies the mode of operation for Port 3, for valid *QplMode* settings. Uses type *PcsSel* (see Table 11-3).
QplMode	18:16	RW	000b	EPL has four PORT interfaces to the switch fabric and four LANE interfaces to the physical media. Single Lane PCS types use one PORT and ONE LANE. Multi-Lane PCS types use all four LANES and all four PORTS for a single stream. *QplMode* allocates LANES to PORTS. A PORT which has multiple LANES allocated can support PCS types with the same or fewer LANES. PCS LANE requirements: • 100GBASE-R — 4 lanes • 40GBASE-R — 4 lanes • 10GBASE-R — 1 lane • 1000BASE-X — 1 lane • SGMII — 1 lane • AN73 — 1 lane Uses type *QplMode* (see Table 11-3).
Reserved	31:19	RSV	0x0	Reserved.

11.3.3.7 EPL_LED_STATUS[0..8]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x306** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Port0Reset	0	vRO	0b	0b = MAC is not in reset. 1b = MAC is in reset.
Port0LinkUp	1	vRO	0b	0b = Link is down. 1b = Link is up.
Port0LocalFault	2	vRO	0b	0b = No event or no new event since last read. 1b = Local Fault event detected.
Port0RemoteFault	3	vRO	0b	0b = No event or no new event since last read. 1b = Remote Fault event detected.
Port0Transmitting	4	CR	0b	0b = No event or no new event since last read. 1b = Frame transmit event detected.
Port0Receiving	5	CR	0b	0b = No event or no new event since last read. 1b = Frame receive event detected.
Port1Reset	6	vRO	0b	0b = MAC is not in reset. 1b = MAC is in reset.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x306** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Port1LinkUp	7	vRO	0b	0b = Link is down. 1b = Link is up.
Port1LocalFault	8	vRO	0b	0b = No event or no new event since last read. 1b = Local Fault event detected.
Port1RemoteFault	9	vRO	0b	0b = No event or no new event since last read. 1b = Remote Fault event detected.
Port1Transmitting	10	vRO	0b	0b = No event or no new event since last read. 1b = Frame transmit event detected.
Port1Receiving	11	CR	0b	0b = No event or no new event since last read. 1b = Frame receive event detected.
Port2Reset	12	vRO	0b	0b = MAC is not in reset. 1b = MAC is in reset.
Port2LinkUp	13	vRO	0b	0b = Link is down. 1b = Link is up.
Port2LocalFault	14	vRO	0b	0b = No event or no new event since last read. 1b = Local Fault event detected.
Port2RemoteFault	15	vRO	0b	0b = No event or no new event since last read. 1b = Remote Fault event detected.
Port2Transmitting	16	CR	0b	0b = No event or no new event since last read. 1b = Frame transmit event detected.
Port2Receiving	17	CR	0b	0b = No event or no new event since last read. 1b = Frame receive event detected.
Port3Reset	18	vRO	0b	0b = MAC is not in reset. 1b = MAC is in reset.
Port3LinkUp	19	vRO	0b	0b = Link is down. 1b = Link is up.
Port3LocalFault	20	vRO	0b	0b = No event or no new event since last read. 1b = Local Fault event detected.
Port3RemoteFault	21	vRO	0b	0b = No event or no new event since last read. 1b = Remote Fault event detected.
Port3Transmitting	22	CR	0b	0b = No event or no new event since last read. 1b = Frame transmit event detected.
Port3Receiving	23	CR	0b	0b = No event or no new event since last read. 1b = Frame receive event detected.
Reserved	31:24	RSV	0x0	Reserved.

11.3.3.8 EPL_FIFO_ERROR_STATUS[0..8]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x307** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TxFifoErrorA	0	CW1	0b	Lane A Transmit FIFO pointer error status.
TxFifoErrorB	1	CW1	0b	Lane B Transmit FIFO pointer error status.
TxFifoErrorC	2	CW1	0b	Lane C Transmit FIFO pointer error status.
TxFifoErrorD	3	CW1	0b	Lane D Transmit FIFO pointer error status.

Address: 0x0E0000 + 0x400*i + 0x307
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
RxFifoErrorA	4	CW1	0b	Lane A Receive FIFO pointer error status.
RxFifoErrorB	5	CW1	0b	Lane B Receive FIFO pointer error status.
RxFifoErrorC	6	CW1	0b	Lane C Receive FIFO pointer error status.
RxFifoErrorD	7	CW1	0b	Lane D Receive FIFO pointer error status.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.9 EPL_TX_FIFO_RD_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x308
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrA	3:0	vRO	0x0	Lane A FIFO read pointer.
ReadPtrB	7:4	vRO	0x0	Lane B FIFO read pointer.
ReadPtrC	11:8	vRO	0x0	Lane C FIFO read pointer.
ReadPtrD	15:12	vRO	0x0	Lane D FIFO read pointer.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.10 EPL_TX_FIFO_WR_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x309
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
WritePtrA	3:0	vRO	0x0	Lane A FIFO write pointer.
WritePtrB	7:4	vRO	0x0	Lane B FIFO write pointer.
WritePtrC	11:8	vRO	0x0	Lane C FIFO write pointer.
WritePtrD	15:12	vRO	0x0	Lane D FIFO write pointer.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.11 EPL_TX_FIFO_A_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x30A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrA	3:0	vRO	0x0	Lane A FIFO read pointer.
WritePtrA	7:4	vRO	0x0	Lane A FIFO write pointer.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.12 EPL_TX_FIFO_B_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x30B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrB	3:0	vRO	0x0	Lane B FIFO read pointer.
WritePtrB	7:4	vRO	0x0	Lane B FIFO write pointer.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.13 EPL_TX_FIFO_C_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x30C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrC	3:0	vRO	0x0	Lane C FIFO read pointer.
WritePtrC	7:4	vRO	0x0	Lane C FIFO write pointer.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.14 EPL_TX_FIFO_D_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x30D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrD	3:0	vRO	0x0	Lane D FIFO read pointer.
WritePtrD	7:4	vRO	0x0	Lane D FIFO write pointer.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.15 EPL_RX_FIFO_RD_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x30E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrA	6:0	vRO	0x0	Lane A FIFO read pointer.
ReadPtrB	13:7	vRO	0x0	Lane B FIFO read pointer.
ReadPtrC	20:14	vRO	0x0	Lane C FIFO read pointer.
ReadPtrD	27:21	vRO	0x0	Lane D FIFO read pointer.
Reserved	31:28	RSV	0x0	Reserved.

11.3.3.16 EPL_RX_FIFO_WR_STATUS[0..8]

Address:	**0x0E0000 + 0x400*i + 0x30F**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
WritePtrA	6:0	vRO	0x0	Lane A FIFO write pointer.
WritePtrB	13:7	vRO	0x0	Lane B FIFO write pointer.
WritePtrC	20:14	vRO	0x0	Lane C FIFO write pointer.
WritePtrD	27:21	vRO	0x0	Lane D FIFO write pointer.
Reserved	31:28	RSV	0x0	Reserved.

11.3.3.17 EPL_RX_FIFO_A_STATUS[0..8]

Address:	**0x0E0000 + 0x400*i + 0x310**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ReadPtrA	6:0	vRO	0x0	Lane A FIFO read pointer.
WritePtrA	13:7	vRO	0x0	Lane A FIFO write pointer.
Reserved	31:14	RSV	0x0	Reserved.

11.3.3.18 EPL_RX_FIFO_B_STATUS[0..8]

Address:	**0x0E0000 + 0x400*i + 0x311**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ReadPtrB	6:0	vRO	0x0	Lane B FIFO read pointer.
WritePtrB	13:7	vRO	0x0	Lane B FIFO write pointer.
Reserved	31:14	RSV	0x0	Reserved.

11.3.3.19 EPL_RX_FIFO_C_STATUS[0..8]

Address:	**0x0E0000 + 0x400*i + 0x312**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ReadPtrC	6:0	vRO	0x0	Lane C FIFO read pointer.
WritePtrC	13:7	vRO	0x0	Lane C FIFO write pointer.
Reserved	31:14	RSV	0x0	Reserved.

11.3.3.20 EPL_RX_FIFO_D_STATUS[0..8]

Address: 0x0E0000 + 0x400*i + 0x313
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ReadPtrD	6:0	vRO	0x0	Lane D FIFO read pointer.
WritePtrD	13:7	vRO	0x0	Lane D FIFO write pointer.
Reserved	31:14	RSV	0x0	Reserved.

11.3.3.21 PCS_ML_BASER_CFG[0..8]

Address: 0x0E0000 + 0x400*i + 0x314
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AmTimeout	15:0	RW	0x3FFF	Alignment marker timeout. The number of 66-bit data blocks between alignment markers for one lane. The minimum setting for this field is 128.
AmSoonEn	16	RW	0b	When *AmSoonEn* is 1b, cfg_rx_pc_request is modified based on the temporal proximity of the next alignment marker. An AmSoon flag is generated as follows to indicate an alignment marker is soon. AmSoon is asserted when the number of bytes to the next alignment marker is less than `RxMaxFrameLength + OnGaurd` AmSoon is negated when the number of bytes past the marker is OffGaurd. 100GBASER: • OnGaurd is between 0 and 50 x 16 bytes (nominally 40) • OffGaurd is between 0 and 50 x 16-bytes (nominally 40) 40GBASER: • OffGaurd is between 0 and 20 x 16-bytes (nominally 10) • OffGaurd is between 0 and 20 x 16-bytes (nominally 10) AmSoon is defined this way for implementation convenience. The port channel MARK_REQUEST signal is generated based on cfg_rx_pc_request, which represents the number of 8-byte (sub-segments). The request should be asserted from the start of the segment, where cfg_rx_pc_request = 0 represents the first, 1 the second case (PCS TYPE). 100GBASER cfg_rx_pc_request = am_soon AND AmSoonEn ? MAC_CFG.*RxPcRequest* + 20 : MAC_CFG.*RxPcRequest*; 40GBASER cfg_rx_pc_request = am_soon AND AmSoonEn ? MAC_CFG.*RxPcRequest* + 4 : MAC_CFG.*RxPcRequest*;
Lane0Sel	18:17	RW	00b	PCS lane resolution for physical to logical lane mapping. 00b = SerDes Lane A = PCS Lane0 01b = SerDes Lane A = PCS Lane1 10b = SerDes Lane A = PCS Lane2 11b = SerDes Lane A = PCS Lane3
Lane1Sel	20:19	RW	00b	PCS lane resolution for physical to logical lane mapping. 00b = SerDes Lane B = PCS Lane0 01b = SerDes Lane B = PCS Lane1 10b = SerDes Lane B = PCS Lane2 11b = SerDes Lane B = PCS Lane3

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x314** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Lane2Sel	22:21	RW	00b	PCS lane resolution for physical to logical lane mapping. 00b = SerDes Lane C = PCS Lane0 01b = SerDes Lane C = PCS Lane1 10b = SerDes Lane C = PCS Lane2 11b = SerDes Lane C = PCS Lane3
Lane3Sel	24:23	RW	00b	PCS lane resolution for physical to logical lane mapping. 00b = SerDes Lane D = PCS Lane0 01b = SerDes Lane D = PCS Lane1 10b = SerDes Lane D = PCS Lane2 11b = SerDes Lane D = PCS Lane3
ManualLaneSel	25	RW	0b	Manual Lane Selection. 0b = Automatic lane selection. 1b = Manual lane selection configured by the field CL82_PCS_BASER_CFG.*LaneXSel*.
ScramblerBypass	26	RW	0b	Scrambler Bypass. 0b = Use scrambler. 1b = Bypass scrambler.
DescramblerBypass	27	RW	0b	Descrambler Bypass. 0b = Use descrambler. 1b = Bypass descrambler.
UseCl91Align	28	RW	0b	Align_status is a variable set by the PCS deskew process to reflect the status of the PCS lane-to-lane alignment. Set true when all lanes are synchronized and aligned. Set false when the deskew process is not complete. To speed up the alignment process in PCS 100GBASER mode, clause 91 alignment may be selected. 0b = PCS 100GBASER use clause 82 align status (standard mode). 1b = PCS 100GBASER use clause 91 align status.
LsMaskSeenI	29	RW	0b	Seen |I| Mask for Link Status. For KR and KR4 this is Seen |I| or |Sequence|. For KR4 this is an 8-byte column, for KR this is a 4-byte column 0b = State affects link status 1b = State ignored
LsMaskHiBer	30	RW	0b	HiBer Mask for Link Status. 0b = State affects link status 1b = State ignored
Reserved	31	RSV	0b	Reserved.

11.3.3.22 PCS_ML_BASER_RX_STATUS[0..8]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*i + 0x315** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MapLaneA	1:0	vRO	00b	Lane resolution for physical to logical lane mapping for 100GBASER. The possible mappings are also restricted to four Lanes per clause 91. 00b = SerDes Lane A = PCS 40GBASER/Clause 91 Lane0 01b = SerDes Lane A = PCS 40GBASER/Clause 91 Lane1 10b = SerDes Lane A = PCS 40GBASER/Clause 91 Lane2 11b = SerDes Lane A = PCS 40GBASER/Clause 91 Lane3

Address:	**0x0E0000 + 0x400*i + 0x315**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
MapLaneB	3:2	vRO	00b	Lane resolution for physical to logical lane mapping for 100GBASER. The possible mappings are also restricted to four Lanes per clause 91. 00b = SerDes Lane B = PCS 40GBASER/Clause 91 Lane0 01b = SerDes Lane B = PCS 40GBASER/Clause 91 Lane1 10b = SerDes Lane B = PCS 40GBASER/Clause 91 Lane2 11b = SerDes Lane B = PCS 40GBASER/Clause 91 Lane3
MapLaneC	5:4	vRO	00b	Lane resolution for physical to logical lane mapping for 100GBASER. The possible mappings are also restricted to four Lanes per clause 91. 00b = SerDes Lane C = PCS 40GBASER/Clause 91 Lane0 01b = SerDes Lane C = PCS 40GBASER/Clause 91 Lane1 10b = SerDes Lane C = PCS 40GBASER/Clause 91 Lane2 11b = SerDes Lane C = PCS 40GBASER/Clause 91 Lane3
MapLaneD	7:6	vRO	00b	Lane resolution for physical to logical lane mapping for 100GBASER. The possible mappings are also restricted to four Lanes per clause 91. 00b = SerDes Lane D = PCS 40GBASER/Clause 91 Lane0 01b = SerDes Lane D = PCS 40GBASER/Clause 91 Lane1 10b = SerDes Lane D = PCS 40GBASER/Clause 91 Lane2 11b = SerDes Lane D = PCS 40GBASER/Clause 91 Lane3
BlockLockLaneA	8	vRO	0b	PCS block lock on Lane A. Block lock indicates that the receiver acquires block delineation on a Lane. 0b = Not locked 1b = Locked For 100GBASER there are five virtual lanes mapped to this physical lane:. Locked indicates that all five virtual lanes have achieved 66-bit block lock.
BlockLockLaneB	9	vRO	0b	PCS block lock on Lane B. Block lock indicates that the receiver acquires block delineation on a Lane. 0b = Not locked 1b = Locked For 100GBASER there are five virtual lanes mapped to this physical lane:. Locked indicates that all five virtual lanes have achieved 66-bit block lock.
BlockLockLaneC	10	vRO	0b	PCS block lock on Lane C. Block lock indicates that the receiver acquires block delineation on a Lane. 0b = Not locked 1b = Locked For 100GBASER there are five virtual lanes mapped to this physical lane:. Locked indicates that all five virtual lanes have achieved 66-bit block lock.
BlockLockLaneD	11	vRO	0b	PCS block lock on Lane D. Block lock indicates that the receiver acquires block delineation on a Lane. 0b = Not locked 1b = Locked For 100GBASER there are five virtual lanes mapped to this physical lane:. Locked indicates that all five virtual lanes have achieved 66-bit block lock.
BerCnt	19:12	vRO	0x0	The current BER count for the HiBer flag.
HiBer	20	vRO	0b	Hi bit error rate as defined in Clause 82, Figure 82-13: BER Monitor State Diagram. 0b = BER acceptable 1b = BER too high
AlignStatus	21	vRO	0b	Resolved align status. Either Cl82AlignStatus or Cl91AlignStatus dependent on configuration ***Note:*** AlignStatus = UseCl91AlignStatus ? bitwise_and(CS_100GBASER_BLOCK_LOCK.*BlockLock*): bitwise_and(PCS_100GBASER_AMPS_LOCK.*AmpsLock*);

Address: **0x0E0000 + 0x400*i + 0x315**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
AmLockLaneA	22	vRO	0b	PCS40GBASER: Alignment marker lock as specified in IEEE Std. 802.3ba, Clause 82, Physical Coding Sublayer (PCS) type 40GBASE-R, Figure 82-11: PCS Alignment Marker Lock State Diagram Implementation. PCS100GBASER: Alignment marker lock on all five virtual lanes for this physical lane.
AmLockLaneB	23	vRO	0b	PCS40GBASER: Alignment marker lock as specified in IEEE Std. 802.3ba, Clause 82, Physical Coding Sublayer (PCS) type 40GBASE-R, Figure 82-11: PCS Alignment Marker Lock State Diagram Implementation. PCS100GBASER: Alignment marker lock on all five virtual lanes for this physical lane.
AmLockLaneC	24	vRO	0b	PCS40GBASER: Alignment marker lock as specified in IEEE Std. 802.3ba, Clause 82, Physical Coding Sublayer (PCS) type 40GBASE-R, Figure 82-11: PCS Alignment Marker Lock State Diagram Implementation. PCS100GBASER: Alignment marker lock on all five virtual lanes for this physical lane.
AmLockLaneD	25	vRO	0b	PCS40GBASER: Alignment marker lock as specified in IEEE Std. 802.3ba, Clause 82, Physical Coding Sublayer (PCS) type 40GBASE-R, Figure 82-11: PCS Alignment Marker Lock State Diagram Implementation. PCS100GBASER: Alignment marker lock on all five virtual lanes for this physical lane.
Reserved	31:26	RSV	0x0	Reserved.

11.3.3.23 PCS_100GBASER_BIP_0[0..8]

Address: **0x0E0000 + 0x400*i + 0x317**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.24 PCS_100GBASER_BIP_1[0..8]

Address: **0x0E0000 + 0x400*i + 0x318**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.25 PCS_100GBASER_BIP_2[0..8]

Address: 0x0E0000 + 0x400*i + 0x319
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.26 PCS_100GBASER_BIP_3[0..8]

Address: 0x0E0000 + 0x400*i + 0x31A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.27 PCS_100GBASER_BIP_4[0..8]

Address: 0x0E0000 + 0x400*i + 0x31B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.28 PCS_100GBASER_BIP_5[0..8]

Address: 0x0E0000 + 0x400*i + 0x31C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.29 PCS_100GBASER_BIP_6[0..8]

Address: 0x0E0000 + 0x400*i + 0x31D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.30 PCS_100GBASER_BIP_7[0..8]

Address:	**0x0E0000 + 0x400*i + 0x31E**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.31 PCS_100GBASER_BIP_8[0..8]

Address:	**0x0E0000 + 0x400*i + 0x31F**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.32 PCS_100GBASER_BIP_9[0..8]

Address:	**0x0E0000 + 0x400*i + 0x320**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.33 PCS_100GBASER_BIP_10[0..8]

Address:	**0x0E0000 + 0x400*i + 0x321**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.34 PCS_100GBASER_BIP_11[0..8]

Address:	**0x0E0000 + 0x400*i + 0x322**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.35 PCS_100GBASER_BIP_12[0..8]

Address: 0x0E0000 + 0x400*i + 0x323
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.36 PCS_100GBASER_BIP_13[0..8]

Address: 0x0E0000 + 0x400*i + 0x324
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.37 PCS_100GBASER_BIP_14[0..8]

Address: 0x0E0000 + 0x400*i + 0x325
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.38 PCS_100GBASER_BIP_15[0..8]

Address: 0x0E0000 + 0x400*i + 0x326
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.39 PCS_100GBASER_BIP_16[0..8]

Address: 0x0E0000 + 0x400*i + 0x327
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.40 PCS_100GBASER_BIP_17[0..8]

Address:	**0x0E0000 + 0x400*i + 0x328**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.41 PCS_100GBASER_BIP_18[0..8]

Address:	**0x0E0000 + 0x400*i + 0x329**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.42 PCS_100GBASER_BIP_19[0..8]

Address:	**0x0E0000 + 0x400*i + 0x32A**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BipErrorCount	15:0	CW	0x0	One of 20 lanes; bit interleaved parity count; saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.43 PCS_100GBASER_BLOCK_LOCK[0..8]

Address:	**0x0E0000 + 0x400*i + 0x32B**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BlockLock	19:0	vRO	0x0	Block lock state from each of the 20 virtual lanes. Bit 0 corresponds to lane 0
Reserved	31:20	RSV	0x0	Reserved.

11.3.3.44 PCS_100GBASER_AMPS_LOCK[0..8]

Address: 0x0E0000 + 0x400*i + 0x32C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AmpsLock	19:0	vRO	0x0	Alignment Marker lock from each of the 20 virtual lanes. Bit 0 corresponds to lane 0.
Reserved	31:20	RSV	0x0	Reserved.

11.3.3.45 RS_FEC_UNCORRECTED[0..8]

Address: 0x0E0000 + 0x400*i + 0x32D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
UncorrectedCwCount	31:0	CW	0x0	FEC uncorrected codeword error counter; wraps.

11.3.3.46 RS_FEC_CFG[0..8]

Address: 0x0E0000 + 0x400*i + 0x32E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	Register file err_write interface to force uncorrectable errors on the RS SAF FIFO (see EPL_ERROR_IP.*RsSafFifoUErr* - Section 11.3.3.2). ErrWrite is XORed with the 2 least significant bits of the parity-protected data word.

AmCounter	19:2	RW	0x20FFF	RS_M = 10
				RS_N = 528
				FEC_CODEWORD_BITS = (RS_M * RS_N)/RS_N_FEC_LANES
				FEC_CODEWORDS = 4096
				FEC_CODEWORDS_BITS = (FEC_CODEWORDS * FEC_CODEWORD_BITS)
				AM_COUNTER_DONE_VAL = (FEC_CODEWORDS_BITS/RS_W_SERDES)-1;
				The field's value must be set to a pre-determined multiple of the CL82_PCS_BASER_CFG.*AmTimeout* register field.
				AmCounter = (8.25 * (AmTimeout + 1)) - 1
				The term (AmTimeout + 1) should be a multiple of 4.
				Note: The 8.25 term is the result of:
				`RS_M * RS_N) / (4 * numSlices * dataWidth) numSlices = 4 dataWidth = 40`
				where the factor 4 is the result of the division of the AM separation distance in units of blocks (16384) by the AMP separation distance in units of Reed-Solomon codewords (4096)

HiSerK	31:20	RW	0x1A1	When the number of errors in an 8192 codeword window is greater than HiSerK, HiSer status is asserted. When ErrorAbility = 0b.
SerErrorInjectTime	39:32	RW	0xA6	When the number of symbol errors in a block of 8192 codewords exceeds HiSerK, the Reed-Solomon decoder causes 2-bit synchronization header of each subsequent 66-bit block that is delivered to the PCS to be assigned a value of 00 or 11 for a period of 60 ms to 75 ms. SerErrorInjectTime allows this time to be configurable: `SerErrorInjectTime * 8192 * 51.2 ns` where 51.2 ns is the code word period.

Address: **0x0E0000 + 0x400*i + 0x32E**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
EnableInbandSer	40	RW	0b	0b = Disable in band sync header corruption due to high codeword error rate. 1b = Enable in band sync header corruption due to high codeword error rate. This bit should be set if either *EnableSaf* = 0b or *ErrorAbility* = 0b
EnableSaf	41	RW	0b	0b = Disable store-and-forward. Do not indicate decoding errors to the PCS sublayer. Low latency configuration. 1b = Enable store-and-forward, FEC errored codewords are handled dependent on ErrorAbility. To be compliant, since there is no correction, *EnableSaf* should be configured 1b.
ErrorAbility	42	RW	0b	0b = Do not indicate decoding errors to the PCS sublayer. 1b = Indicate decoding errors to the PCS sublayer by intentionally corrupting 66-bit block synchronization headers as defined in Clause 91.5.3.3.
RestartLock	43	RW	0b	When 1b, forces the synchronization process into the LOCK_INIT state as shown in Clause 91, Figure 91-8: FEC Synchronization State Diagram.
MaskSmRestartLock	44	RW	0b	When 1, prevents Clause 91 Figure 91–9 FEC alignment state machine from restarting Clause 91 Figure 91–8 FEC synchronization state machine, allowing the link to stay up under heavy noise conditions. In this state software can use RestartLock to restart FEC synchronization state machine.
MaskAligned	45	RW	0b	Controls when code-word errors are counted in RS_FEC_UNCORRECTED.*UncorrectedCwCount*. 0b = Count code-word errors when aligned. 1b = Count code-word errors regardless of the aligned state.
Reserved	63:46	RSV	0x0	Reserved.

11.3.3.47 RS_FEC_STATUS[0..8]

Address: **0x0E0000 + 0x400*i + 0x330**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
HiSer	0	vRO	0b	Clause 91 high code word error rate status. Signals during the current period.
HiSerCorrupt	1	vRO	0b	Clause 91 high code word error rate in band corruption status. The persisted error status used to ensure decoded blocks are corrupted when passed to the clause 82 PCS layer.
AlignStatus	2	vRO	0b	Clause 91 Align Status.
FecAlignState	5:3	vRO	000b	Clause 91 FEC Align state diagram. Uses type *FecAlignState* (see Table 11-3).
Reserved	31:6	RSV	0x0	Reserved.

11.3.3.48 EPL_SYSTIME[0..8]

Time reference. The time unit is system implementation depended but should normally represent nanoseconds.

Address: 0x0E0000 + 0x400*i + 0x331
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
CurrentTime	31:0	vRO	0x0	System time.

11.3.3.49 EPL_SYSTIME0[0..8]

Address: 0x0E0000 + 0x400*i + 0x332
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
Time0	31:0	RW	0x0	The content of this register is transfered to PCIE_SYSTIME at time 0 reset from management.

11.3.3.50 EPL_SYSTIME_CFG[0..8]

Address: 0x0E0000 + 0x400*i + 0x333
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
Step	3:0	RW	0xA	The number of time unit per tick. At least 2. Values of 1 or 0 are not supported.
Reserved	31:4	RSV	0x0	Reserved.

11.3.3.51 PORT_STATUS[0..8][0..3]

The port status information register defines the current state of the port as a high level. The Tx*XXX* and Rx*XXX* fields reflect the actual operating mode negotiated with the link partner if this port is configured for auto-negotiation, and reflects actual EPL_CFG setting if this port is not in auto-negotiation mode.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
LinkFaultDebounced	2:0	vRO	000b	This field is intended to be used by software to determine the Link Status. The debounced link fault signal provides a filtered version of *RxLinkFault* intended for software use. Uses type *LinkFaultSignal* (see Table 11-3).
LinkFaultMac	5:3	vRO	000b	This field indicates the hardware state. *LinkFaultMac* depends on *LinkFaultRx* and *LinkFaultDebounced*, and is used by MAC hardware to select appropriate packet handling and fault signaling. Uses type *LinkFaultSignal* (see Table 11-3).

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x0** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LinkFaultRx	8:6	vRO	000b	Receive side link fault signal. 1000BASE-X PCS type, does not use Link Fault signaling state diagram, but uses the FT_OK and FT_LOCAL_FAULT signals to indicate the status of the PHY. 10GBASE-R PCS types, use Clause 46, Link Fault State Diagram. 40GBASE-R and 100GBASE-R types, use Clause 81, Link Fault State Diagram. Uses type *LinkFaultSignal* (see Table 11-3).
RxLinkUp	9	vRO	0b	Link status before debounce filtering and fault signaling. 0b = Link down. 1b = Link up.
HeartbeatOk	10	vRO	0b	Heartbeat OK. 0b = Flatline 1b = Beating
HiBer	11	vRO	0b	This bit error rate flag is only set in combination with BASER PCS types. 0b = Bit error rate is acceptable. 1b = Bit error rate is too high.
Transmitting	12	CR	0b	Transmitting. 0b = No event or no new event since clear. 1b = Frame transmit event detected.
Receiving	13	CR	0b	Receiving. 0b = No event or no new event since clear. 1b = Frame receive event detected.
SerXmit	14	vRO	0b	Port Lanes transmit ability. 0b = Lanes are not ready to transmit. 1b = Lanes are ready to transmit.
TxLpIdle	15	vRO	0b	Transmit low power idle indication v RO 0x0 0b = Normal mode 1b = Low power idle mode
RxLpIdle	16	vRO	0b	Receive low power idle indication valid when *RxLinkUp* is true. 0b = Normal mode. 1b = Low power idle mode.
EeeSlPcSilent	17	vRO	0b	Energy efficient Ethernet single lane switch port silent indicates the switch port has been silent for an extended period as configured with MAC_CFG.*TxPcActivityTimeout* and MAC_CFG.*TxPcActTimeScale*. This field is valid in single lane modes only, including 1000BASEX and 10GBASER. 0b = Activity on switch port. 1b = Silent.
Pcs	21:18	vRO	0x0	Indicates which PCS mode is active. Uses type *PcsSel* (see Table 11-3).
Reserved	31:22	RSV	0x0	Reserved.

11.3.3.52 AN_IM[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x1**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
An73TransmitDisable	0	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73AbilityDetect	1	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73AcknowledgeDetect	2	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73CompleteAcknowledge	3	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73NextPageWait	4	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73AnGoodCheck	5	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73AnGood	6	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73ReceiveIdle	7	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An73MrPageRx	8	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37AnEnable	9	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37AnRestart	10	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37AnDisableLinkOk	11	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37AbilityDetect	12	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37AcknowledgeDetect	13	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37CompleteAcknowledge	14	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37NextPageWait	15	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37IdleDetect	16	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37LinkOk	17	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
An37MrPageRx	18	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
Reserved	31:19	RSV	0x0	Reserved.

11.3.3.53 LINK_IM[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x2** **32** **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
LinkFaultDebounced	0	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
LinkFaultRx	1	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
RxLinkUp	2	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
Rx8b10bCommaDet	3	RW	1b	*Rx8b10bCommaDet* Interrupt Mask.
Rx8b10bDisparityErr	4	RW	1b	*Rx8b10bDisparityErr* Interrupt Mask.
Rx8b10bOutOfBandErr	5	RW	1b	*Rx8b10bOutOfBandErr* Interrupt Mask.
Pcs1000basexSyncStatus	6	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
Pcs10gbaserBlockLock	7	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
PcsBaserHiBer	8	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
PcsMlBaserBlockLockAllLanes	9	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
PcsMlBaserAlignStatus	10	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
MacRxFrame	11	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
MacRxFaultSequence	12	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorTxFcs	13	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxPreamble	14	RW	1b	0b = Enable interrupt. 1b = Mask interrupt. ***Note:*** This interrupt fires when minIfg is not met for PCS10GBASER modes. This occurs when the IFG is less than 4.
ErrorRxCode	15	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxFrame	16	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxFcs	17	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxOversize	18	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxJabber	19	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxUndersize	20	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxRunt	21	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
ErrorRxOverrun	22	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x2**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
ErrorTxUnderrun	23	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
EgressTimeStamp	24	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
LpiWakeError	25	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
LpIdleIndicate	26	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
EeePcSilent	27	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
Cl91HiSer	28	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
TxFrameError	29	RW	1b	0b = Enable interrupt. 1b = Mask interrupt.
Reserved	31:30	RSV	00b	Reserved.

11.3.3.54 AN_IP[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x3**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
An73TransmitDisable	0	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_TRANSMIT_DISABLE state. 0b = No interrupt pending. 1b = Interrupt pending.
An73AbilityDetect	1	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_ABILITY_DETECT state. 0b = No interrupt pending. 1b = Interrupt pending.
An73AcknowledgeDetect	2	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_ACKNOWLEDGE_DETECT state. 0b = No interrupt pending. 1b = Interrupt pending.
An73CompleteAcknowledge	3	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_COMPLETE_ACKNOWLEDGE state. 0b = No interrupt pending. 1b = Interrupt pending.
An73NextPageWait	4	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_NEXT_PAGE_WAIT state. 0b = No interrupt pending. 1b = Interrupt pending.
An73AnGoodCheck	5	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_GOOD_CHECK state. 0b = No interrupt pending. 1b = Interrupt pending.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x3**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			

Field Name	Bit(s)	Type	Default	Description
An73AnGood	6	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_GOOD state. 0b = No interrupt pending. 1b = Interrupt pending.
An73ReceiveIdle	7	CW1	0b	The variable an_receive_idle is asserted when the clause 73 receive state diagram is in the IDLE or DELIMITER DETECT state. This interrupt is asserted when an_receive_idle signal changes. 0b = No interrupt pending. 1b = Interrupt pending.
An73MrPageRx	8	CW1	0b	Asserted when a page is received. 0b = No interrupt pending. 1b = Interrupt pending.
An37AnEnable	9	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_ENABLE state. 0b = No interrupt pending. 1b = Interrupt pending.
An37AnRestart	10	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the AN_RESTART state. 0b = No interrupt pending. 1b = Interrupt pending.
An37AnDisableLinkOk	11	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the DISABLE_LINK_OK state. 0b = No interrupt pending. 1b = Interrupt pending.
An37AbilityDetect	12	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the ABILITY_DETECT state. 0b = No interrupt pending. 1b = Interrupt pending.
An37AcknowledgeDetect	13	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the ACKNOWLEDGE_DETECT state. 0b = No interrupt pending. 1b = Interrupt pending.
An37CompleteAcknowledge	14	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the COMPLETE_ACKNOWLEDGE state. 0b = No interrupt pending. 1b = Interrupt pending.
An37NextPageWait	15	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the NEXT_PAGE_WAIT state. 0b = No interrupt pending. 1b = Interrupt pending.
An37IdleDetect	16	CW1	0b	Asserted when idle is detected. 0b = No interrupt pending. 1b = Interrupt pending.
An37LinkOk	17	CW1	0b	Asserted when the Auto-Negotiation arbitration state machine transitions to the LINK_OK state. 0b = No interrupt pending. 1b = Interrupt pending.
An37MrPageRx	18	CW1	0b	Asserted when a page is received. 0b = No interrupt pending. 1b = Interrupt pending.
Reserved	31:19	RSV	0x0	Reserved.

11.3.3.55 LINK_IP[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x4**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
LinkFaultDebounced	0	CW1	0b	Debounced Link Fault change event. Used for all PCS modes. Remote Fault is not valid for 1000BASE-X. 0b = No interrupt pending. 1b = Interrupt pending.
LinkFaultRx	1	CW1	0b	Link Fault change event. Used for all PCS modes. Remote Fault is not valid for 1000BASE-X. 0b = No interrupt pending. 1b = Interrupt pending.
RxLinkUp	2	CW1	0b	Receive Link Up change event. Receive link status excluding heartbeat monitor. 0b = No interrupt pending. 1b = Interrupt pending.
Rx8b10bCommaDet	3	CW1	0b	Valid 8b10b comma character has been detected.
Rx8b10bDisparityErr	4	CW1	0b	8b10b disparity error detected.
Rx8b10bOutOfBandErr	5	CW1	0b	Invalid 8b10b detected.
Pcs1000basexSyncStatus	6	CW1	0b	PCS 1000BASEX Sync Status change event. Sync status indicates whether the PCS is synchronized to code-group boundaries. 0b = No interrupt pending. 1b = Interrupt pending.
Pcs10gbaserBlockLock	7	CW1	0b	PCS 10GBASER Block Lock change event. Block lock indicates that the receiver acquires block delineation. 0b = No interrupt pending. 1b = Interrupt pending.
PcsBaserHiBer	8	CW1	0b	PCS10GBASER and PCS40GBASER High Bit Error Rate Interrupt. HiBer indicates that the block sync field is received with an unacceptable error rate. 0b = No interrupt pending. 1b = Interrupt pending.
PcsMlBaserBlockLockAllLanes	9	CW1	0b	PCS 40GBASER Block Lock on All Lanes change event. Block lock indicates that the receiver acquires block delineation on a Lane. 0b = No interrupt pending. 1b = Interrupt pending.
PcsMlBaserAlignStatus	10	CW1	0b	PCS 40GBASER Align Status change event. Indicates that the alignment process is complete. 0b = No interrupt pending. 1b = Interrupt pending.
MacRxFrame	11	CW1	0b	Asserted when a frame is received. 0b = No interrupt pending. 1b = Interrupt pending.
MacRxSequenceOrFsig	12	CW1	0b	Asserted when a Sequence code or Fsig is detected. Value held in RX_SEQUENCE. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorTxFcs	13	CW1	0b	Asserted when an FCS error is detected on an otherwise good frame. 0b = No interrupt pending. 1b = Interrupt pending.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x4
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ErrorRxPreamble	14	CW1	0b	Asserted when a bad preamble is received. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxCode	15	CW1	0b	Asserted when an error code is received. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxFrame	16	CW1	0b	Asserted when a frame is terminated with a code error and the frame is not classed as runt or jabber. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxFcs	17	CW1	0b	Asserted when an FCS error is detected on an otherwise good frame. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxOversize	18	CW1	0b	Asserted when a received frame is too long and has a good FCS. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxJabber	19	CW1	0b	Asserted when a received frame is too long and has a bad FCS. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxUndersize	20	CW1	0b	Asserted when a received frame is too short and has a good FCS. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxRunt	21	CW1	0b	Asserted when a received frame is too short and has a bad FCS. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorRxOverrun	22	CW1	0b	Asserted when the receive overrun condition is detected, indicating that the switch fabric is congested. 0b = No interrupt pending. 1b = Interrupt pending.
ErrorTxUnderrun	23	CW1	0b	Asserted when the receive underrun condition is detected. Indicates that the switch fabric is congested. 0b = No interrupt pending. 1b = Interrupt pending.
EgressTimeStamp	24	CW1	0b	Asserted in IEEE1588 mode when the transmitted frame's interrupt request is asserted. Indicates that the IEEE1588_PACKET_EGRESS_TIME is valid. 0b = No interrupt pending. 1b = Interrupt pending.
LpiWakeError	25	CW1	0b	EEE wake time faults where the PHY fails to complete its normal wake sequence after a period of quiescence for transmit signals. 0b = No interrupt pending. 1b = Interrupt pending.
LpIdleIndicate	26	CW1	0b	Change of the EEE low power idle signal. Current state is PORT_STATUS.*RxLpIdle*. 0b = No interrupt pending. 1b = Interrupt pending.
EeePcSilent	27	CW1	0b	Port Channel Silent. Energy efficient Ethernet signal asserted when entering or exiting the port channel monitor silent state.
Cl91HiSer	28	CW1	0b	Clause 91 high coder word error rate.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x4** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TxFrameError	29	CW1	0b	For each egress frame the switch can signal, the frame is corrupted due to an internal error, such as an SRAM soft error. The switch error interrupt is asserted when the PortChannel signals a switch error encoded as: `port_channel.info==ERR(3) AND port_channel.mark=EOF`
Reserved	31:30	RSV	00b	Reserved.

11.3.3.56 MP_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x6** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
GeneralPurposeField	47:0	RW	0x0	A general purpose register.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.57 AN_37_PAGE_RX[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x8** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
GeneralPurposeField	15:0	vRO	0x0	A general purpose register.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.58 AN_73_PAGE_RX[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0xA** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
GeneralPurposeField	47:0	vRO	0x0	A general purpose register.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.59 LINK_RULES[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0xC
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
FaultTimeScaleUp	3:0	RW	0x0	Defines a period the link must be stable before raising an interrupt to software. Uses type *TickTimeScale* (see Table 11-3).
FaultTicksUp	8:4	RW	0x0	The number of LinkStatusTimeScaleUp ticks.
FaultTimeScaleDown	12:9	RW	0x0	Defines a period the link must be stable before raising an interrupt to software. Uses type *TickTimeScale* (see Table 11-3).
FaultTicksDown	17:13	RW	0x0	The number of LinkStatusTimeScaleDown ticks.
HeartbeatTimeScale	21:18	RW	0x0	Each link type has a heartbeat, a condition that is expected to repeat regularly. This field defines the time-frame in which the heartbeat must be recognized. Uses type *TickTimeScale* (see Table 11-3).
Reserved	31:22	RSV	0x0	Reserved.

11.3.3.60 MAC_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x10
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
TxAntiBubbleWatermark	5:0	RW	0x1	The Port Channel Anti-Bubble FIFO provides buffering to increase the word-to-word jitter tolerance. *TxAntiBubbleWatermark* specifies the number of words that are stored in the FIFO before a frame is scheduled for egress. The minimum setting is 1, and the maximum setting is 62. However, the recommended settings for standard data rates are as follows (other values might create errors on the interface): • 100 GbE (4-lanes) *TxAntiBubbleWatermark* = 4 • 40 GbE (4-lanes) *TxAntiBubbleWatermark* = 4 • 25 GbE (1-lanes) *TxAntiBubbleWatermark* = 6 • 10 GbE (1-lanes) *TxAntiBubbleWatermark* = 3 • 2.5 GbE (1-lanes) *TxAntiBubbleWatermark* = 4 • 1 GbE (1-lanes) *TxAntiBubbleWatermark* = 4 • AN73 (1-lanes) *TxAntiBubbleWatermark* = 3
TxRateFifoWatermark	9:6	RW	0x0	Specifies the water mark for the rate adaption FIFO. If the watermark is above this level, tokens are generated slowly. Otherwise, tokens are generated slowly. The recommended settings for standard data rates are as follows (other values might create errors on the interface): • 100 GbE (4-lanes) *TxRateFifoWatermark* = 3 • 40 GbE (4-lanes) *TxRateFifoWatermark* = 5 • 25 GbE (1-lanes) *TxRateFifoWatermark* = 3 • 10 GbE (1-lanes) *TxRateFifoWatermark* = 3 • 2.5 GbE (1-lanes) *TxRateFifoWatermark* = 3 • 1 GbE (1-lanes) *TxRateFifoWatermark* = 3 • AN73 (1-lanes) *TxRateFifoWatermark* = 3

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x10**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
TxRateFifoFastInc	17:10	RW	0x1	Consider that N bits are transmit at line rate each core clock period. Set *TxRateFifoFastInc* greater than N. The recommended settings for standard data rates are as follows (other values might create errors on the interface): • 100 GbE (4-lanes) *TxRateFifoFastInc* = (core_clk@800MHz:129)(core_clk@798.61MHz:130) • 40 GbE (4-lanes) *TxRateFifoFastInc* = 52 • 25 GbE (1-lane) *TxRateFifoFastInc* = 33 • 10 GbE (1-lane) *TxRateFifoFastInc* = 13 • 2.5 GbE (1-lanes *TxRateFifoFastInc* = 4 • 1 GbE (1-lane) *TxRateFifoFastInc* = 2 • AN73 (1-lanes) *TxRateFifoFastInc* = 4
TxRateFifoSlowInc	25:18	RW	0x2	Consider that N bits are transmit at line rate each core clock period. Set *TxRateFifoFastInc* less than N. The recommended settings for standard data rates are as follows (other values might create errors on the interface): • 100 GbE (4-lanes) *TxRateFifoFastInc* = (core_clk@800MHz:128)(core_clk@798.61MHz:129) • 40 GbE (4-lanes) *TxRateFifoFastInc* = 51 • 25 GbE (1-lane) *TxRateFifoFastInc* = 32 • 10 GbE (1-lane) *TxRateFifoFastInc* = 12 • 2.5 GbE (1-lane) *TxRateFifoFastInc* = 3 • 1 GbE (1-lane) *TxRateFifoFastInc* = 1 • AN73 (1-lanes) *TxRateFifoFastInc* = 3
TxIdleMinIfgBytes	31:26	RW	0xC	Minimum number of idle bytes to be transmit. If deficit idle count is enabled, this is the average value for the minimum inter-frame gap.
TxClockCompensationTimeout	47:32	RW	0x0	The periodic rate, in effective PCS Clock periods to schedule an idle byte deletion. • 100GBASER count unit is 1.28 ns • 40GBASER count unit is 3.2 ns • 25GBASER count unit is 1.28 ns • 10GBASER count unit is 3.2 ns • 1GBASE count unit is 8 ns • SGMII 1000 count unit is 8 ns • SGMII 100 count unit is 80 ns • SGMII 10 count unit is 800 ns
TxClockCompensationEnable	48	RW	0b	Controls periodic idle column deletion for clock compensation. For 100GBASE-R and 40GBASE-R columns are 8 bytes. For 25GBASER and 10GBASER columns are 4 bytes. For all others, columns are 1 byte. 0b = Disable clock compensation. 1b = Enable clock compensation.
TxFaultMode	51:49	RW	000b	Controls the transmitted fault sequences. Uses type *TxMacFaultMode* (see Table 11-3).
TxPcActTimeScale	55:52	RW	0x0	Select the tick period used for the activity timer. Uses type *TickTimeScale* (see Table 11-3).
TxPcActTimeout	63:56	RW	0x0	The switch port is declared silent after there has been no tokens available for: `TxPcActTimeScale * TxPcActTimeout` The measurement error is zero to one *TxPcActTimeScale*.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x10**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			

Field Name	Bit(s)	Type	Default	Description
TxDrainMode	65:64	RW	10b	Force the port to drain all packets regardless of link status. The drain is applied immediately and may cut short an in-flight frame. Force the port to hold any packets regardless of link status. The hold is applied cleanly at the beginning of a frame. Port drains all packets automatically when the link is down. Port holds transmission when the link is down. Uses type *TxMacDrainMode* (see Table 11-3).
TxMinColumns	71:66	RW	0x12	Defines the minimum egress frame length in units of 4-octet columns. This setting includes 8 octets, or 2 columns, for Preamble or FTAG. Shorter frames egressing the switch are padded. To configure a minimum packet length of 64 bytes, this register should be configured to the number of Packet columns plus the number of Preamble or FTAG columns: 64/4 + 2 = 18. Only *TxMinColumns* values from 12 to 31 are supported.
TxSegMinSpacing	83:72	RW	0x0	Minimum delay between start of transmitted segments, in units of 8BT. Value 0 disables segment rate throttle.
TxSegMaxCredit	95:84	RW	0x0	Maximum credits allowed for segment rate throttle, in units of 8BT.
TxSegSize	107:96	RW	0x18	Segment size used by segment rate throttle, in units of 8-bytes. Value 0b means 32 KB. Value 1b is not supported.
TxLpIdleRequest	108	RW	0b	Energy Efficient Ethernet Low Power Idle Request, generated by LPI client. To request transmit Low Power Idle generation, MAC_CFG.*TxLpIdleRequest* must transition from 0b to 1b. To disable transmit Low Power Idle generation configure MAC_CFG.*TxLpIdleRequest* to 0b. In automatic mode, port channel activity can also disable transmit Low Power Idle generation. In Low Power Idle mode, traffic is interrupted, and the LI signal is sent to the PCS. 0b = Normal operation. 1b = Request low power mode. ***Note:*** Not all PCS support EEE.
TxLpiAutomatic	109	RW	0b	0b = Manual Low power mode is exited when *TxLpIdleRequest*==0 1b = Automatic Low power mode is exited when *TxLpIdleRequest*==0 or a packet is available for transmission.
TxLpiTimeout	117:110	RW	0x0	After disabling the LPI mode the system waits for: `TxLpiTimescale * TxLpiTimeout` (with an error of +0/-*TxLpiTimescale*) before allowing traffic.
TxLpiTimescale	119:118	RW	00b	Tick timescale for the timer that expires after *TxLpiTimeout* ticks. Uses type *EeeTimeScale* (see Table 11-3).
TxLpiHoldTimeout	127:120	RW	0x0	After *TxLpiTimeout* is done, the *TxLpiHoldTimer* is started, and traffic is allowed to egress once this timer is done. This field configured the timer: `TxLpiHoldTimescale * TxLpiHoldTimeout` (with an error of +0/-TxLpiTimescale)
TxLpiHoldTimescale	129:128	RW	00b	Tick timescale for the timer that expires after *TxLpiHoldTimeout* ticks Uses type *EeeTimeScale* (see Table 11-3).
TxFcsMode	132:130	RW	100b	Select the FCS mode. Uses type *TxMacFcsMode* (see Table 11-3).

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x10
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
TxObeyLint	133	RW	0b	Transmit Obey Link Interrupt. 0b = When receiving Link Fault LINK_INTERRUPT transmit emits idles and frames per the Link Up *TxDrainMode* configuration 1b = When receiving Link Fault LINK_INTERRUPT transmit emits idles and frames per the Link Down *TxDrainMode* configuration. This occurs on a frame boundary, the frame is not halted and retransmit the frame later.
TxIdleEnableDic	134	RW	0b	On transmit, it is necessary to modify the length of the inter-frame in order to align the Start control character on lane 0. The method is selected: 0b = Always insert additional idle characters to align the start of preamble on a four byte boundary. 1b = Enable Deficit Idle Count — Maintain the effective data rate by sometimes inserting and sometimes deleting idle characters to align the Start control character.
CjpatEnable	135	RW	0b	Cjpat is not supported for 40GBASE-R and 100GBASE-R PCS modes. 0b = Disable (Output idle) 1b = Enable (Output CjPat)
RxMinFrameLength	143:136	RW	0x40	Specify the minimum frame length (in bytes) allowed for the receive stream.

RxMaxFrameLength	159:144	RW	0x3C00	Specify the maximum frame length (in bytes) allowed for the receive stream.
				The maximum value supported by the switch is 15 KB. The ingress frame is truncated. The rules for ingress frame truncation are as follows:
				Round the configured *RxMaxFrameLength* down to the nearest MII column boundary (4-byte columns for single lane PCS, and 8-byte columns for multi-lane PCS), add one, then convert back to bytes.
				Valid values of emaxLength are integer multiples of the miiWidth. The units of *emaxLength* are bytes.
				`emaxLength = (ROUNDDOWN(MaxFrameLength/miiWidth) + 1)* miiWidth`
				The units of truncated frame length are bytes. The truncated frame length is the minimum of *emaxFrameLength* or *ingressFrameLength*
				`TruncatedFrameLength = Math.min(emaxLength, ingressFrameLength)`

StartCharD	167:160	RW	0x0	Override data bits for the start character.
Ieee1588Enable	168	RW	0b	On ingress, the FCS is replaced by a time stamp and is forwarded to the switch fabric. On egress, if CAPTURE_TIME=1 and MAC_1588_STATUS is zero, ingress and egress time stamps are written to MAC_1588_STATUS and LINK_IP.*EgressTimeStamp* interrupt is raised. 0b = Disable IEEE1588 Mode. 1b = Enable IEEE1588 Mode.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x10**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FcsStart	169	RW	0b	FCS calculation can be configured to start at column 0 or 8. When MAC_CFG.*PreambleMode* = PM_PASSTHRU, byte 0 is the first byte of the preamble. When MAC_CFG.*PreambleMode* = PM_STRIP, byte 0 is the first byte of the frame (after preamble). Valid combinations are: • PM_PASSTHRU and FcsStart = 0 CRC includes modified preamble. • PM_PASSTHRU and FcsStart = 1 CRC excludes preamble. • PM_STRIP and FcsStart = 0 CRC starts the first byte after preamble. The following combination is invalid and not supported: • PM_STRIP and FcsStart = 1 CRC starts on the ninth byte of the frame after preamble. The modified preamble has the first byte replaced by MAC_CFG.*StartCharD*.
PreambleMode	170	RW	0b	Controls Preamble handling for 10 GbE and faster PCS modes. Setting to PM_PASSTHRU enables preamble forwarding. Setting to PM_STRIP enables preamble stripping. Uses type *PreambleMode* (see Table 11-3).
CounterWrap	171	RW	0b	Counter Wrap. 0b = Saturate 1b = Wrap
LinkFaultDisable	172	RW	0b	Disables the link fault state machine. 0b = Link fault signaling state machine enabled. 1b = Link fault signaling state machine disabled, link_fault is OK.
RxDrain	173	RW	0b	Receive Drain. 0b = Normal operation. 1b = Drain all packets — Do not forward packets to the switch.
RxFcsForceBad	174	RW	0b	Controls whether the extracted Frame Check Sequence is forced to be bad. The FCS in the data stream is not effected. The frame passed to the switch is ended with a Tail Error. 0b = Do not invert extracted FCS. 1b = Invert extracted FCS.
RxIgnoreCodeErrors	175	RW	0b	Ignore receive code errors. 0b = Normal operation. 1b = Ignore code errors.
RxIgnoreUndersizeErrors	176	RW	0b	Ignore receive undersize errors. 0b = Normal operation. 1b = Undersize frames are passed to the switch as good.
RxIgnoreOversizeErrors	177	RW	0b	Ignore receive oversize errors. 0b = Normal operation. 1b = Oversize frames are truncated and passed to the switch as good.
RxIgnoreFcsErrors	178	RW	0b	Ignore receive FCS errors. 0b = Normal operation. 1b = Ignore FCS errors.
RxIgnorePreambleErrors	179	RW	0b	Ignore receive preamble errors. 0b = Normal operation. 1b = Ignore preamble errors.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x10**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
RxIgnoreIfgErrors	180	RW	0b	Ignore inter-frame gap errors. If not set, checks that at least one idle column appears before the start of a frame. Applies for modes other than 40GBASE-R and 100GBASE-R, which do not have the check in the first place.
ErrWrite	182:181	RW	00b	Register file err_write interface to force uncorrectable errors on the Jitter FIFO (see EPL_ERROR_IP.JitterFifo* - Section 11.3.3.2). *ErrWrite* is XORed with the 2 least significant bits of the data word.
RxMinEventRate	186:183	RW	0x0	The purpose of this function is to provide a higher bound on the packet event rate. RxMinEventRate sets the minimum spacing, in units of 8-byte columns, from the start of one frame to the start of the next. Frames that start with less than this spacing are discarded. This field should be set to minimum frame size plus minimum IFG rounded down to the nearest 8-byte column boundary. Packets dropped because they do not meet the *RxMinFrameSapcing* rule are recorded as runt.
RxPcRequest	191:187	RW	0x18	For data entering the switch fabric over the wide-portchannel, REQUEST is an optional indication that the switch fabric may schedule the current 192-byte segment for transmission. Used to program the port channel request. Each segment is 192 bytes: 24 x 8-byte columns. Indicates in which wide-port-channel word the REQUEST flag should be sent relative to start of segment. Configuring *RxPxcRequest* to ZERO causes a request to be sent in the first column of each segment. Configuring *RxPxcRequest* to 23 causes REQUEST to be sent on the last word of a segment. When configured to any value greater than 23 a request is not sent.
RxPcSegSize	196:192	RW	0x18	EPL can generate an end-of-segment (EOS) flag in the EPL to Switch direction. Segment size is configurable as (8 * *RxPcSegSize*) bytes for *RxPcSegSize* from 1 to 24. For other settings no EOS flag is generated. Configuring a lower setting causes memory utilization to decrease, since 192 bytes is allocated for each segment, but also decreases latency. This is a debug feature.
Reserved	255:197	RSV	0x0	Reserved.

11.3.3.61 TX_SEQUENCE[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x1A**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
TxSequenceD	31:0	RW	0x0	Last sequence or FSIG column Data Field.
TxSequenceK	35:32	RW	0x0	Last sequence or FSIG column Control Field.
Reserved	63:36	RSV	0x0	Reserved.

11.3.3.62 RX_SEQUENCE[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x1C**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
RxSequenceD	31:0	vRO	0x0	Last sequence or FSIG column Data Field.
RxSequenceK	35:32	vRO	0x0	Last sequence or FSIG column Control Field.
Reserved	63:36	RSV	0x0	Reserved.

11.3.3.63 MAC_1588_STATUS[0..8][0..3]

When a frame egresses with a time stamp request (CAPTURE_TIME=1) on a port for which time stamping is enabled (MAC_CFG.*Ieee1588Enable*) and when all fields in this register are zero, this register is updated with ingress and egress time stamps associated with the frame. After reading this register, software must reset this register to zero to capture another frame's time stamp.

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x1E**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
EgressTimeStamp	31:0	CW	0x0	The egress time for the first byte of the frame's preamble.
IngressTimeStamp	63:32	CW	0x0	The ingress time for the first byte of the frame's preamble.

11.3.3.64 WAKE_ERROR_COUNTER[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x20**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
WakeError	15:0	CW	0x0	Used by PHY XS that support EEE to count wake time faults where the PHY XS fails to complete its normal wake sequence after a period of quiescence for the transmit signals. The fault event to be counted may occur during a refresh or a wake-up. This counter saturates.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.65 MAC_OVERSIZE_COUNTER[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x21**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
FrameErrCntOversize	31:0	CW	0x0	Oversize Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts well-formed frames that are greater than max length.

11.3.3.66 MAC_JABBER_COUNTER[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x22
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
FrameErrCntJabber	31:0	CW	0x0	Jabber Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts badly-formed frames that are greater than max length.

11.3.3.67 MAC_UNDERSIZE_COUNTER[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x23
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
FrameErrCntUndersize	31:0	CW	0x0	Undersize Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts well-formed frames that are less than min length.

11.3.3.68 MAC_RUNT_COUNTER[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x24
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
FrameErrCntRunt	31:0	CW	0x0	Runt Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts badly-formed frames that are less than min length. ***Note:*** This counter also counts number of frames dropped while the port is forced in receive drain mode.

11.3.3.69 MAC_OVERRUN_COUNTER[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x25
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
FrameErrCntOverrun	31:0	CW	0x0	Overrun Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts frames lost due to switch congestion.

11.3.3.70 MAC_UNDERRUN_COUNTER[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x26
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
UnderrunCnt	31:0	CW	0x0	Underrun Event Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts the number of frames lost because data was not available from the switch

11.3.3.71 MAC_CODE_ERROR_COUNTER[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x27**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
CodeErrorCnt	31:0	CW	0x0	Code Error Counter (rollover controlled by MAC_CFG.*CounterWrap*). Counts the number of /E/ decoded bytes are received. Disparity errors cause /E/ to be generated.

11.3.3.72 EPL_TX_FRAME_ERROR_COUNTER[0..8][0..3]

Address:	**0xE0000 + 0x400*j + 0x80*i + 0x28**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
TxFrameErrorCnt	31:0	CW	0x0	Counts the number of frames received from the switch marked as error. Errors can be due to soft errors in the switch memory, and in cut through mode, frame receive errors. In store and forward mode, this counter provides the same information as a soft-error-counter. This counter wraps.

11.3.3.73 MAC_LINK_COUNTER[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x29**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
NoFault	11:0	CW	0x0	Increment on transition to Link Up state (rollover controlled by MAC_CFG.*CounterWrap*).
LocalFault	21:12	CW	0x0	Increment on transition to Local Fault state (rollover controlled by MAC_CFG.*CounterWrap*).
RemoteFault	31:22	CW	0x0	Increment on transition to Remote Fault state (rollover controlled by MAC_CFG.*CounterWrap*).

11.3.3.74 PCS_1000BASEX_CFG[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x2A**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
LsMaskSeenI1orI2	0	RW	0b	Link Status Mask for Seen I1 or I2 or C1 or C2. 0b = Enable 1b = Mask
DisableCheckEnd	1	RW	0b	Disable Check End. 0b = IEEE behavior. 1b = Simplify check end rules.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x2A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
IgnoreDisparityError	2	RW	0b	When Rx_8b10bDecodeEn = 1b, the 8B10B decoder outside of the SerDes can be configured to ignore disparity errors 0b = Normal operation. 1b = Ignore disparity errors.
8b10bDisparityReset	3	RW	0b	0b = Normal operation. 1b = Reset encoder disparity to negative.
8b10bInjectDisparityErr	4	RW	0b	0b-to-1b transition inverts the encoder's disparity signal.
EnEee	5	RW	0b	0b = Disable Clause 36 Receive EEE support. 1b = Enable Clause 36 Receive EEE support.
Reserved	31:6	RSV	0x0	Reserved.

11.3.3.75 PCS_1000BASEX_RX_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x2B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
State	4:0	vRO	0x0	Clause 36, Figure 36-7: PCS Receive State Diagram Uses type *Pcs1000basexRxState* (see Table 11-3).
Rx8b10bCommaDet	5	vRO	1b	Current state of comma detect. See the interrupt to detect changes.
Rx8b10bDisparityErr	6	vRO	1b	Current state of disparity error. See the interrupt to detect changes.
Rx8b10bOutOfBandErr	7	vRO	1b	Current state of out of band error. See the interrupt to detect changes.
DecodeCarrierDetect	8	vRO	0b	For the latched code-group ([/x/]) carrier_detect function detects carrier when either: a) A two or more bit difference between [/x/] and both /K28.5/ encodings exists; or b) A two to nine bit difference between [/x/] and the expected /K28.5/ (based on current running disparity) exists.
CodeSyncStatus	9	vRO	0b	PCS Synchronization process flag. 0b = FAIL The receiver is not synchronized to code-group boundaries. 1b = OK The receiver is synchronized to code-group boundaries.
SyncStatus	10	vRO	0b	*CodeSyncStatus* OR *RxLpiActive* 0b = FAIL The receiver is not synchronized to code-group boundaries and is not in *RxLpiActive* state. 1b = OK The receiver is synchronized to code-group boundaries or is on *RxLpiActive* state.
RxLpiActive	11	vRO	0b	Indication that the receiver is in the low power idle mode.
RxQuiet	12	vRO	0b	Indication that the receiver is in the low power quiet state.
EnableCgalign	13	vRO	0b	Enable code-group alignment function. Valid for PCS1000BASEX.
LastBad10bCode	23:14	vRO	0x0	Valid in PCS1000BASEX mode. Once a basic symbol lock is established, the 10-bit code of any badly decoded symbol is captured.
Reserved	31:24	RSV	0x0	Reserved.

11.3.3.76 PCS_1000BASEX_TX_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x2C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
CodeGrpState	3:0	vRO	0x0	Clause 36, Figure 36-6: PCS Transmit Code Group State Variable Uses type *TxCgState* (see Table 11-3).
TxOsetState	7:4	vRO	0x0	Clause 36, Figure 36-5 PCS Transmit Ordered Set stPcs1000basexate Variable Uses type *TxOsetState* (see Table 11-3).
TxOset	10:8	vRO	000b	Command that indicates the type of ordered-set to transmit Uses type *TxOset* (see Table 11-3).
TxLpiState	12:11	vRO	00b	Clause 36m Figure 36–10 - LPI Transmit state diagram. Uses type *Cl36TxLpiState* (see Table 11-3).
Reserved	31:13	RSV	0x0	Reserved.

11.3.3.77 PCS_10GBASER_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x2D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ScramblerBypass	0	RW	0b	Control to bypass the scrambler function. 0b = Use scrambler. 1b = Bypass scrambler.
DescramblerBypass	1	RW	0b	Control to bypass the descrambler function. 0b = Use descrambler. 1b = Bypass descrambler.
LsMaskSeenI	2	RW	0b	Seen |I| Mask for Link Status. 0b = State affects link status. 1b = State ignored.
LsMaskHiBer	3	RW	0b	HiBer Mask for Link Status. 0b = State affects link status. 1b = State ignored.
EnEee	4	RW	0b	0b = Disable Clause 49 Receive EEE support. 1b = Enable Clause 49 Receive EEE support.
Reserved	31:5	RSV	0x0	Reserved.

11.3.3.78 PCS_10GBASER_RX_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x2E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
RxBlockLock	0	vRO	0b	Block delineation from block lock process. Used by low power receive process.
BlockLock	1	vRO	0b	Block delineation from low power receive process. Used for link.
HiBer	2	vRO	0b	Boolean variable asserted when the ber_cnt exceeds 16.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x2E** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BerCnt	10:3	vRO	0x0	The current BER count, for the HiBer flag.
RxMode	11	vRO	0b	Receive mode as controlled by the low power receive state. Uses type *TgbrLpiRxMode* (see Table 11-3).
RxLpiActive	12	vRO	0b	0b = Normal mode. 1b = Low power mode.
Reserved	31:13	RSV	0x0	Reserved.

11.3.3.79 PCS_10GBASER_TX_STATUS[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x2F** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TxState	2:0	vRO	000b	Clause 49, Figure 49-14: Transmit State Variable. Uses type *TxPcs10gbaserState* (see Table 11-3).
TxMode	4:3	vRO	00b	A variable set to QUIET when the transmitter is in the TX_QUIET state, set to ALERT when the transmitter is in the TX_ALERT state, and set to DATA otherwise. When set to QUIET, the PMD disables the transmitter. When set to ALERT, the PMD transmits a repeating pattern of eight ones and eight zeros. When set to DATA the PMD passes data as normal. Uses type *Pcs10gbaserTxMode* (see Table 11-3).
TxLpiState	7:5	vRO	000b	Clause 49: Transmit LPI State Diagram Uses type *Cl49TxLpiState* (see Table 11-3).
GbAlarm	8	vRO	0b	10GBASER transmit gearbox alarm. Indicates overrun and should not assert during normal operation if the system is configured correctly.
Reserved	31:9	RSV	0x0	Reserved.

11.3.3.80 AN_37_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x30** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MrAnEnable	0	RW	0b	Enable for Auto-Negotiation.
NpEnable	1	RW	0b	Signal to disable the optional NEXT_PAGE_WAIT implementation in Figure 37-6: Auto-Negotiation State Diagram. Next page is always disabled in SGMII mode. 0b = Disable NEXT_PAGE_WAIT implementation. 1b = Enable NEXT_PAGE_WAIT implementation.
ToggleNpLoaded	2	RW	0b	Management should transition this bit from 0b to 1b to indicate that a valid Next Page is available.
Reserved	31:3	RSV	0x0	Reserved.

11.3.3.81 AN_37_TIMER_CFG[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x34**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
TimeScale	3:0	RW	0x0	Select the tick period used for the timers specified below. Since the timescale counters are free running, when a timeout value, n, is specified the actual timeout is between (n-1 and n) tick periods. Uses type *TickTimeScale* (see Table 11-3).
LinkTimerTimeout	10:4	RW	0x0	
Reserved	31:11	RSV	0x0	Reserved.

11.3.3.82 AN_37_BASE_PAGE_TX[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x6**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
Rsrvd4to0	4:0	RW	0x0	Reserved. Should be set to 0b.
FullDuplex	5	RW	0b	Full Duplex ability. 0b = Not supported. 1b = Supported.
HalfDuplex	6	RW	0b	Half Duplex ability. 0b = Not supported. 1b = Supported.
Pause	7	RW	0b	*Pause* (0b) and *AsmDir* (0b) = No PAUSE. *Pause* (0b) and *AsmDir* (1b) = Asymmetric PAUSE toward link partner. *Pause* (1b) and *AsmDir* (0b) = Symmetric PAUSE. *Pause* (1b) and *AsmDir* (1b) = Both Symmetric PAUSE and Asymmetric PAUSE toward local device.
AsmDir	8	RW	0b	
Rsrvd11to9	11:9	RW	000b	Reserved. Should be set to 0b.
RemoteFault	13:12	RW	00b	Remote Fault. 00b = No error. Link OK 01b = Off line 10b = Link_Failure 11b = Auto-Negotiation Error
ACK	14	RW	0b	Used by hardware to communicate the receipt of next page. Should be set to 0b.
NP	15	RW	0b	Next Page. 0b = No more next page information to be sent. 1b = Request next page transmission.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.83 AN_37_BASE_PAGE_RX[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x8
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
Rsrvd4to0	4:0	vRO	0x0	Reserved. Should be set to 0b.
FullDuplex	5	vRO	0b	Full Duplex ability. 0b = Not supported 1b = Supported.
HalfDuplex	6	vRO	0b	Half Duplex ability. 0b = Not supported 1b = Supported.
Pause	7	vRO	0b	*Pause* (0b) and *AsmDir* (0b) = No PAUSE. *Pause* (0b) and *AsmDir* (1b) = Asymmetric PAUSE toward link partner. *Pause* (1b) and *AsmDir* (0b) = Symmetric PAUSE. *Pause* (1b) and *AsmDir* (1b) = Both Symmetric PAUSE and Asymmetric PAUSE toward local device.
AsmDir	8	vRO	0b	
Rsrvd11to9	11:9	vRO	000b	Reserved. Should be set to 0b.
RemoteFault	13:12	vRO	00b	Remote Fault. 00b = No error. Link OK 01b = Off line 10b = Link_Failure 11b = Auto-Negotiation Error
ACK	14	vRO	0b	Used by hardware to communicate the receipt of next page. Should be set to 0b.
NP	15	vRO	0b	Next Page. 0b = No more next page information to be sent. 1b = Request next page transmission.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.84 AN_37_NEXT_PAGE_TX[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
MU	10:0	RW	0x0	Two types of message are sent: Unformatted or Formatted. This field is the Message Code Field for Message Page encoding. This field is the Unformatted Code Field for Unformatted Message encoding.
T	11	RW	0b	Toggle. Used by hardware to ensure synchronization during next page exchange. Should be set to 0b.
ACK2	12	RW	0b	Used by the Next Page function to indicate that a device has the ability to comply with the message. 0b = Cannot comply with message. 1b = Can comply with message.
MP	13	RW	0b	Message Page flag. 0b = Unformatted Page. 1b = Message Page.
ACK	14	RW	0b	Used by hardware to communicate the receipt of next page. Should be set to 0b.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x6** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
NP	15	RW	0b	Next Page 0b = Last Page. 1b = Additional next page(s) to follow.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.85 AN_37_NEXT_PAGE_RX[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x8** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MU	10:0	vRO	0x0	Two types of message are sent: Unformatted or Formatted. This field is the Message Code Field for Message Page encoding. This field is the Unformatted Code Field for Unformatted Message encoding.
T	11	vRO	0b	Toggle. Used by hardware to ensure synchronization during next page exchange. Should be set to 0b.
ACK2	12	vRO	0b	Used by the Next Page function to indicate that a device has the ability to comply with the message. 0b = Cannot comply with message. 1b = Can comply with message.
MP	13	vRO	0b	Message Page flag. 0b = Unformatted Page. 1b = Message Page.
ACK	14	vRO	0b	Used by hardware to communicate the receipt of next page. Should be set to 0b.
NP	15	vRO	0b	Next Page 0b = Last Page. 1b = Additional next page(s) to follow.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.86 SGMII_AN_TIMER_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x34** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TimeScale	3:0	RW	0x0	Select the tick period used for the timers specified below. Since the timescale counters are free running, when a timeout value, n, is specified, the actual timeout is between (n-1 and n) tick periods. Uses type *TickTimeScale* (see Table 11-3).
LinkTimerTimeout	10:4	RW	0x0	
Reserved	31:11	RSV	0x0	Reserved.

11.3.3.87 SGMII_AN_TX_CONFIG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
B0	0	RW	0b	Should always be set to 1b.
Rsrvd9_1	9:1	RSV	0x0	Should always be set to 0b.
Rsrvd11to10	11:10	RW	00b	Should always be set to 0b.
Rsrvd12	12	RW	0b	Should always be set to 0b.
Rsrvd13	13	RW	0b	Should always be set to 0b.
OneB14	14	RW	0b	Should always be set to 1b.
Rsrvd15	15	RW	0b	Should always be set to 0b.
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.88 SGMII_AN_TX_CONFIG_LOOPBACK[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
B0	0	RW	0b	Expected to be 1b.
Rsrvd9to1	9:1	RW	0x0	Should always be set to 0b.
Speed	11:10	RW	00b	Speed selection. 00b = 10 MB/s 01b = 100 MB/s 10b = 1000 MB/s 11b = Reserved
Duplex	12	RW	0b	Duplex Mode. 0b = Half duplex 1b = Full duplex
Rsrvd13	13	RW	0b	Expected to be 0b.
Ack	14	RW	0b	Auto-Negotiation acknowledge.
Link	15	RW	0b	Link Status. 0b = Link down 1b = Link up
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.89 SGMII_AN_RX_CONFIG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x8
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
B0	0	vRO	0b	Expected to be 1b.
Rsrvd9to1	9:1	vRO	0x0	Should always be set to 0b.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x8**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
Speed	11:10	vRO	00b	Speed selection. 00b = 10 MB/s 01b = 100 MB/s 10b = 1000 MB/s 11b = Reserved
Duplex	12	vRO	0b	Duplex Mode. 0b = Half duplex 1b = Full duplex
Rsrvd13	13	vRO	0b	Expected to be 0b.
Ack	14	vRO	0b	Auto-Negotiation acknowledge.
Link	15	vRO	0b	Link Status. 0b = Link down 1b = Link up
Reserved	31:16	RSV	0x0	Reserved.

11.3.3.90 AN_37_STATUS[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x32**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
CurrentLcw	15:0	vRO	0x0	Current received link code word.
MrAnComplete	16	vRO	0b	Status indicating whether Auto-Negotiation is complete or not.
State	20:17	vRO	0x0	Clause 37, Figure 37-6. Uses type *An37State* (see Table 11-3).
AbilityMatch	21	vRO	0b	Flag indicates three consecutive Link Codewords match, ignoring the *Acknowledge* bit.
AcknowledgeMatch	22	vRO	0b	Flag indicates three consecutive Link Codewords match, with the *Acknowledge* bit set.
ConsistencyMatch	23	vRO	0b	Indicates that the ability_match_word is the same as the Link Codeword that caused acknowledge_match.
IdleMatch	24	vRO	0b	Indicates that three consecutive /I/ ordered_sets have been received.
LinkTimerDone	25	vRO	0b	Flag that indicates the Link Timer is done.
Rudi	27:26	vRO	00b	Status indicating the type of ordered_sets. Uses type *Rudi* (see Table 11-3).
AnSyncStatus	28	vRO	0b	Qualified version of sync_status for Auto-Negotiation to detect sync_status timeout condition.
Reserved	31:29	RSV	000b	Reserved.

11.3.3.91 AN_73_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x33** **32** **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
MrAutonegEnable	0	RW	0b	Control to enable Auto-Negotiation. 0b = Disable 1b = Enable
IgnoreNonceMatch	1	RW	0b	Ignore Nonce Match. 0b = Normal nonce match function. 1b = All patterns match, disabling the nonce match function.
IncompatibleLink	2	RW	0b	Incompatible Link. 0b = Resolution function indicates a compatible link exists. 1b = Resolution function indicates an incompatible link exists. Forces transition from AN GOOD CHECK to TRANSMIT DISABLE.
ClrNonceCnt	3	RW	0b	Control to clear NONCE_CNT. Clause 73.6.3: Transmitted Nonce Field. Transmitted Nonce field (T[4:0]) is a 5-bit wide field containing a random or pseudo-random number. A new value is generated for each entry to the Ability Detect state. The transmitted nonce should have a uniform distribution in the range from 0 to 31. For this purpose a 5-bit NONCE_CNT counter is implemented. The transmitted *T* field that will be `AN_73_BASE_PAGE_TX.T[4:0] ^ NONCE_CNT` NONCE_CNT is cleared when *ClrNonceCnt* is 1b, otherwise it increments each time Ability Detect state is entered.
TxNextPageLoaded	4	RW	0b	Flag indicating whether the a next page has been loaded.
Reserved	31:5	RSV	0x0	Reserved.

11.3.3.92 AN_73_TIMER_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x34** **32** **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
TimeScale	3:0	RW	0x0	Select the tick period used for the timers specified below. Since the timescale counters are free running, when a timeout value, n, is specified, the actual timeout is between (n-1 and n) tick periods. Uses type *TickTimeScale* (see Table 11-3).
BreakLinkTimeout	10:4	RW	0x0	Number of *TimeScale* ticks (60-75 ms).
LinkFailInhibitTimeout	19:11	RW	0x0	Number of *TimeScale* ticks. LinkFailInibitTimeoutKr (40-50 ms) LinkFailInibitTimeoutKr4 (500-510 ms)
Reserved	31:20	RSV	0x0	Reserved.

11.3.3.93 AN_73_BASE_PAGE_TX[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
S	4:0	RW	0x0	Selector Field. Clause 73.6.1: Selector Field.
E	9:5	RW	0x0	Echoed Nonce Field. Clause 73.6.2: Echoed Nonce Field.

C	12:10	RW	000b	Pause Ability.
				Clause 73.6.6: Pause Ability.
				C0 = The same as PAUSE as defined in Annex 28B.
				C1 = The same as ASM_DIR as defined in Annex 28B
				C2 = Reserved.

RF	13	RW	0b	Remote Fault. Clause 73.6.7: Remote Fault.
ACK	14	RW	0b	Acknowledge. Clause 73.6.8: Acknowledge. Used by the Auto-Negotiation function.
NP	15	RW	0b	Next Page. Clause 73.6.9: Next Page.
T	20:16	RW	0x0	Transmitted Nonce Field. Clause 73.6.3: Transmitted Nonce Field. Used as a bit-wise XOR with a 5-bit NONCE counter which is incremented on entry to the ABILITY DETECT state. The value transmitted is nonce_cnt[4:0] ^ T[4:0]. This allows 32 different patterns to be generated automatically by hardware. If a deadlock occurs, software may write a new value to this field to select a different sequence.

A	45:21	RW	0x0	Technology Ability Field.
				Clause 73.6.4: Technology Ability Field.
				A0 = 1000BASE-KX
				A1 = 10GBASE-KX4 (not supported)
				A2 = 10GBASE=KR

F	47:46	RW	00b	FEC Capability Field. Clause 73.6.5: FEC Capability Field.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.94 AN_73_BASE_PAGE_RX[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0xA
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
S	4:0	vRO	0x0	Selector Field. Clause 73.6.1: Selector Field.
E	9:5	vRO	0x0	Echoed Nonce Field. Clause 73.6.2: Echoed Nonce Field.

C	12:10	vRO	000b	Pause Ability.
				Clause 73.6.6: Pause Ability.
				C0 = The same as PAUSE as defined in Annex 28B.
				C1 = The same as ASM_DIR as defined in Annex 28B
				C2 = Reserved.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0xA**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
RF	13	vRO	0b	Remote Fault. Clause 73.6.7: Remote Fault.
ACK	14	vRO	0b	Acknowledge. Clause 73.6.8: Acknowledge. Used by the Auto-Negotiation function.
NP	15	vRO	0b	Next Page. Clause 73.6.9: Next Page.
T	20:16	vRO	0x0	Transmitted Nonce Field. Clause 73.6.3: Transmitted Nonce Field.

A	45:21	vRO	0x0	Technology Ability Field.
				Clause 73.6.4: Technology Ability Field.
				A0 = 1000BASE-KX
				A1 = 10GBASE-KX4 (not supported)
				A2 = 10GBASE=KR
				A3 = 40GBASE-KR4
				A4 = 40GBASE-CR4
				A5 = 100GBASE-CR10 not supported
				A6 = 100GBASE-KP4 not supported
				A7 = 100GBASE-KR4
				A8 = 100GBASE-CR4

F	47:46	vRO	00b	FEC Capability Field. Clause 73.6.5: FEC Capability Field.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.95 AN_73_NEXT_PAGE_TX[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x6**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
MU	10:0	RW	0x0	Message code field, or U[10:0] of the Unformatted Code Field. Clause 73, Figures 73-7 and 73-8.
T	11	RW	0b	Toggle to maintain synchronization between the local device and the link partner. Controlled by the Auto-Negotiation function.
ACK2	12	RW	0b	Used to indicate whether the receiver is able to act on the information. Clause 73.7.7: Next Page function.
MP	13	RW	0b	Message Page encoding. 0b = Unformatted Next Pages. 1b = Formatted Next Pages.
ACK	14	RW	0b	Used by the Auto-negotiation hardware to acknowledge receipt of information.
NP	15	RW	0b	Indicates whether more Next Pages are to be transmit/received.
U	47:16	RW	0x0	Unformatted Code Field.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.96 AN_73_NEXT_PAGE_RX[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0xA
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
MU	10:0	vRO	0x0	Message code field, or U[10:0] of the Unformatted Code Field. Clause 73, Figures 73-7 and 73-8.
T	11	vRO	0b	Toggle to maintain synchronization between the local device and the link partner. Controlled by the Auto-Negotiation function.
ACK2	12	vRO	0b	Used to indicate whether the receiver is able to act on the information. Clause 73.7.7: Next Page function.
MP	13	vRO	0b	Message Page encoding. 0b = Unformatted Next Pages. 1b = Formatted Next Pages.
ACK	14	vRO	0b	Used by the Auto-negotiation hardware to acknowledge receipt of information.
NP	15	vRO	0b	Indicates whether more Next Pages are to be transmit/received.
U	47:16	vRO	0x0	Unformatted Code Field.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.97 AN_73_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x36
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
RxPageCount	5:0	vRO	0x0	Counter increments for each page received, wraps at 0x3F, and is reset when an_receive_idle is asserted.
State	8:6	vRO	000b	Clause 73, Figure 73-11: Arbitration State Variable. Uses type *An73State* (see Table 11-3).
TxNonce	13:9	vRO	0x0	Transmitted nonce.
AbilityMatch	14	vRO	0b	Flag indicates three consecutive Link Codewords match, ignoring the *Acknowledge* bit Latched on transition to TRANSMIT_DISABLE to assist in determining the reason for AN restarting.
AckNonceMatch	15	vRO	0b	Indicates whether the echoed nonce received matches the transmitted nonce Latched on transition to TRANSMIT_DISABLE to assist in determining the reason for AN restarting.
AcknowledgeMatch	16	vRO	0b	Flag indicates three consecutive Link Codewords match, with the *Acknowledge* bit set Latched on transition to TRANSMIT_DISABLE to assist in determining the reason for AN restarting.
AnLinkGood	17	vRO	0b	Indicates that Auto-Negotiation has completed.
AnReceiveIdle	18	vRO	0b	Indicates that the receiver state diagram is in the IDLE or DELIMITER_DETECT state.
BasePage	19	vRO	0b	Indicates that the base page, initial Link Codeword, is currently being transmitted.
CompleteAck	20	vRO	0b	Controls the counting of transmitted Link Codewords that have their *Acknowledge* bit set.
ConsistencyMatch	21	vRO	0b	Indicates that the ability_match_word is the same as the Link Codeword that caused acknowledge_match Latched on transition to TRANSMIT_DISABLE to assist in determining the reason for AN restarting.
MrAutonegComplete	22	vRO	0b	Status indicating whether Auto-Negotiation is complete or not.
MrLpAutonegAble	23	vRO	0b	Status indicating whether the link partner supports Auto-Negotiation.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x36
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
NonceMatch	24	vRO	0b	Status indicating transmitted nonce matches the received nonce Latched on transition to TRANSMIT_DISABLE to assist in determining the reason for AN restarting.
NpRx	25	vRO	0b	Flag indicating the state of rx_link_code_word[NP] on entry to the COMPLETE_ACKNOWLEDGE state.
ToggleRx	26	vRO	0b	The state of the link partner's *Toggle* bit.
ToggleTx	27	vRO	0b	The state of the local device's *Toggle* bit.
TransmitAbility	28	vRO	0b	Controls the transmission of the Link Codeword.
TransmitAck	29	vRO	0b	Controls the setting of the *Acknowledge* bit in tx_link_code_word.
TransmitDisable	30	vRO	0b	Controls the transmission of tx_link_code_word.
Reserved	31	RSV	0b	Reserved.

11.3.3.98 AN_73_TX_LCW[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x38
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
TxLinkCodeWord	47:0	vRO	0x0	Auto-Negotiation (Clause 73) transmit link code word.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.99 AN_73_RX_LCW[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x3A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AbilityMatchWord	47:0	vRO	0x0	Auto-Negotiation (Clause 73) receive ability match word.
Reserved	63:48	RSV	0x0	Reserved.

11.3.3.100 PCSL_CFG[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x3C**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
SlipTime	7:0	RW	0x1E	Used in all BASER modes. Specifies the minimum time between bit-slips in units of the core clock period, nominally 1.25 ns
RxBitSlipEnable	8	RW	0b	When SerDes fast bit slip is enabled, this bit enables the PCS slip signal: `i_core_to_cntl[1] = slip` The SerDes can be configured to enable fast bit slip control. When enabled, the i_core_to_cntl[1] input can be used to perform a single bit slip. When using this mode, an inversion of i_core_to_cntl[1] results in a single bit slip. The following sequence should be used to enable this feature. Interrupts are issued using the SAI interface. 1. Issue interrupt 12 with only bit[7] set to '1' (interrupt_data = 0x0080): This requests the current state of bit slip. Valid return codes for this interrupt request are 0x00 and 0x01. 2. Set i_core_to_cntl[1] to current slip state (based on return code bit[0] from step 1). 3. Issue interrupt 12 with just bit[8] set to 1b (interrupt_data = 0x0100): enables direct override of slip in ESB. 4. Inverting i_core_to_cntl[1] causes a bit slip. 5. A delay of at least four o_rx_fifo_clk cycles between inversions/slips. 6. It takes 10 o_rx_fifo_clk cycles to flush a slip 7. Repeat steps 4-6 until framing of data is complete. 8. Issue interrupt 12 with all bits set to 0b (interrupt_data = 0x0000). Turns off direct control of bit slips. ESB maintains its current slip value.
RxBitSlipInitial	9	RW	0b	The initial value used for slip in the fast bit slip sequence. See *RxBitSlipEnable* (bit 8).
RxGbNarrow	10	RW	0b	Valid in PCS10GBASER and PCS40GBASER modes. The gearbox is configured to the SERDES width. For operation up to 10 GbE, *RxGbNarrow* may be configured to 1 for minimal latency. 0b = Operation up to 25 GbE — Configure SerDes parallel port to wide (40 bits). 1b = Operation up to 10 GbE — Configure SerDes parallel port to narrow (20 bits).
TxGbNarrow	11	RW	0b	The gearbox is configured to the SERDES width. For operation up to 10 GbE, *TxGbNarrow* may be configured to 1 for minimal latency. 0b = Operation up to 25 GbE — Configure SerDes parallel port to wide (40-bits). 1b = Operation up to 10 GbE — Configure SerDes parallel port to narrow (20-bits).
Reserved	31:12	RSV	0x0	Reserved.

11.3.3.101 MP_EEE_CFG[0..8][0..3]

Address: **0x0E0000 + 0x400*j + 0x80*i + 0x40**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
GeneralPurpose	105:0	RW	0x0	
Reserved	127:106	RSV	0x0	Reserved.

11.3.3.102 PCS_1000BASEX_EEE_CFG[0..8][0..3]

Address:	0x0E0000 + 0x400*j + 0x80*i + 0x40
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
TxTslTime	7:0	RW	0x0	Transmit LPI Local Sleep Time from entering the TX_SLEEP state to when tx_quiet is set to TRUE 19.9-20.1 µs.
TxTqlTime	15:8	RW	0x0	Transmit Lpi Local Quiet Time from when tx_quiet is set to TRUE to entry into the TX_REFRESH state 2.5-2.6 ms.
TxTulTime	23:16	RW	0x0	Local Refresh Time from entry into the TX_REFRESH state to entry into the TX_QUIET state 19.9-20.1 µs.
RxTqrTime	31:24	RW	0x0	The time the receiver waits for signal detect to be set to OK while in the LP_IDLE_D, LPI_K and RX_QUIET states before asserting a rx_fault 3-4 ms.
RxTwrTime	39:32	RW	0x0	Time the receiver waits in the RX_WAKE state before indicating a wake time fault (WTF) 11 µs.
RxTwtfTime	47:40	RW	0x0	Wake time fault recovery time 1 ms.
TxTslUnit	49:48	RW	00b	Timer tick unit selection. ***Note:*** 50 ns is divided to create a 100 ns tick. Uses type *EeeTimeScale* (see Table 11-3).
TxTqlUnit	51:50	RW	00b	Timer tick unit selection. ***Note:*** 50 ns is divided to create a 100 ns tick. Uses type *EeeTimeScale* (see Table 11-3).
TxTulUnit	53:52	RW	00b	Timer tick unit selection. ***Note:*** 50 ns is divided to create a 100 ns tick. Uses type *EeeTimeScale* (see Table 11-3).
RxTqrUnit	55:54	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
RxTwrUnit	57:56	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
RxTwtfUnit	59:58	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
FilterTrap	60	RW	0b	For Energy Efficient Ethernet there can be a short period when signal is being removed that the system may decode invalid sequences as good sequences, causing unexpected results. A feature is added to protect against this unwanted behavior, When lost_signal is asserted prior to energy_detect negate, the Receive state transitions from LP_IDLE_D or LPI_K to RX_NOISE state. *FilterTrap* controls how to exit the RX_NOISE state. 0b = Return to LPI_K when ~lp_signal_lost AND i_solid_sync_status AND is_k28_5 AND even_next0_q. 1b = Stay in RX_NOISE state awaiting energy_detect to be negated or the timeout to complete.

LostSignalMaskSignal	61	RW	0b	Lost signal mask for signal detect.
				Lost signal controls transitions to and from the RX_NOISE state:
				`sd = LostSignalMaskSignalDetect OR i_signal_detect`
				`lost_signal = er OR (~sd) OR (~sync);`

LostSignalMaskError	62	RW	0b	Lost signal mask for signal detect. Lost signal controls transitions to and from the RX_NOISE state: `er = (~LostSignalMaskError) AND i_code_sync_status AND decode_next2_d.err lost_signal = er OR (~sd) OR (~sync);`
LostSignalMaskSync	63	RW	0b	Lost signal mask for signal detect. Lost signal controls transitions to and from the RX_NOISE state: `sync = LostSignalMaskSync OR i_code_sync_status; lost_signal = er OR (~sd) OR (~sync);`

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x40** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Rsrvd105to64	105:64	RW	0x0	Reserved. Should be set to 0b.
Reserved	127:106	RSV	000b	Reserved.

11.3.3.103 PCS_10GBASER_EEE_CFG[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x40** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Pattern	31:0	RW	0xFF00FF00	Alert pattern is a repeating pattern in the EEE wake/refresh sequence. The default is the pattern for 10GBASE-KR PMD.
TxTslTime	39:32	RW	0x0	Local Sleep Time from entering the TX_SLEEP state to when tx_mode is set to QUIET 4.9-5.1 µs.
TxTqlTime	47:40	RW	0x0	Local Quiet Time from when tx_mode is set to QUIET to entry into the TX_ALERT state 1.7-1.8 ms.
TxTwlTime	55:48	RW	0x0	Time spent in the TX_WAKE state 10.9-11.1 µs.
TxT1uTime	63:56	RW	0x0	Time spent in the TX_ALERT and RW 0x0 TX_SCR_BYPASS states 1.1-1.3 µs.
RxTqrTime	71:64	RW	0x0	The time the receiver waits for energy_detect to be set to TRUE while in the RX_SLEEP and RX_QUIET states before asserting receive fault 2-3 ms.
RxTwrTime	79:72	RW	0x0	Time the receiver waits in the RX_WAKE state before indicating a wake time fault (when scr_bypass_enable = FALSE) 11.5 µs.
RxTwtfTime	87:80	RW	0x0	Wake time fault recovery time 10 ms.
TxTslUnit	89:88	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
TxTqlUnit	91:90	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
TxTwlUnit	93:92	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
TxT1uUnit	95:94	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
TxT1uUnit	97:96	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
RxTwrUnit	99:98	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
RxTwtfUnit	101:100	RW	00b	Timer tick unit selection. Uses type *EeeTimeScale* (see Table 11-3).
FilterTrap	102	RW	0b	For Energy Efficient Ethernet there can be a short period when signal is being removed that the system may decode invalid sequences as good sequences causing unexpected results. A feature is added to protect against this unwanted behavior. When lost_lock is asserted prior to energy_detect negated, the Receive LPI state transitions from RX_SLEEP to NOISE. *FilterTrap* controls how to exit the NOISE state. 0b = Return to RX_SLEEP when (~lost_lock) AND (rtype_is_li_q OR rtype_is_idle_q). 1b = Stay in NOISE state awaiting energy_detect to be negated or the timeout to complete.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x40
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
LostLockMaskEnergy	103	RW	0b	Lost lock mask for energy detect. Lost lock controls transitions to and from the NOISE state. When not masker lost lock is asserted when energy detect is negated: `ed = i_cfg_eee_lost_lock_mask_ed OR i_energy_detect; lost_lock = er OR (~ed) OR (~idle) OR (~i_rx_block_lock)`
LostLockMaskError	104	RW	0b	Lost lock mask for error, Lost lock controls transitions to and from the NOISE state. When not masked lost lock is asserted when an error block is decoded: `er = (~LostSignalMaskError) AND rtype_is_e_q; lost_lock = er OR (~ed) OR (~idle) OR (~i_rx_block_lock)` ***Note:*** Contrary to other mask errors, setting this bit to 0b masks the error, and setting to 1b un-masks the error. Recommended setting is 1b.
LostLockMaskIdle	105	RW	0b	Lost lock mask for idle. Lost lock controls transitions to and from the NOISE state. When not masked, lost lock is asserted if the input block is other than idle or low power idle: `idle = LostSignalMaskIdle OR rtype_is_li_q OR rtype_is_idle_q; lost_lock = er OR (~ed) OR (~idle) OR (~i_rx_block_lock)`
Reserved	127:106	RSV	0x0	Reserved.

11.3.3.104 MP_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ErrorCnt	31:0	CW	0x0	Decode error counter. Wraps.

11.3.3.105 PCS_40GBASER_RX_BIP_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BipCnt	31:0	CW	0x0	Bit-interleaved parity error counter. Wraps. ***Note:*** The error counter does count individual errors as long as they are separated in time by at least 4 valid symbol.

11.3.3.106 PCS_10GBASER_RX_BER_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
BerCnt	31:0	CW	0x0	Counts invalid sync headers. Wraps. ***Note:*** Does not count when block lock is false.

11.3.3.107 DISPARITY_ERROR_8B10B[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
ErrorCnt	31:0	CW	0x0	For 1000Base-X or SGMII. Counts the number of disparity/code errors. Wraps. ***Note:*** Does not count when sync status is false for 1000Base-X.

11.3.3.108 LANE_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x45
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
EeeHwRxQuiet	0	RW	0b	Allows hardware control of the EEE signals used to control SerDes. 0b = Software — core_to_cntl[5] = LANE_SERDES_CFG.*CoreToCntl*[5] 1b = Hardware — core_to_cntl[5] = hardware signal rx_quiet
EeeHwTxQuiet	1	RW	0b	Allows hardware control of the EEE signals used to control SerDes. 0b = Software — core_to_cntl[6] = LANE_SERDES_CFG.*CoreToCntl*[6] 1b = Hardware — core_to_cntl[6] = hardware signal rx_quiet
EeeHwLpiDisable	2	RW	0b	Allows hardware control of the EEE signals used to control SerDes. 0b = Software — core_to_cntl[7] = LANE_SERDES_CFG.*CoreToCntl*[7] 1b = Hardware — core_to_cntl[7] = hardware signal rx_quiet
RefClockADiv	3	RW	0b	Clock divider selection for local clock source. 0b = Divide local clock source by 2. 1b = Divide local clock source by 4.
RefClockASource	4	RW	0b	Set the reference clock "A" output source. The pass-through passes the clock "A" from previous EPL stage to the next stage unchanged. The EPLs are daisy-chained together from 0 through 23. If all EPLs are in pass-through, no reference clock is provided. Uses type *RefClockSource* (see Table 11-3).
RefClockBDiv	5	RW	0b	Clock divider selection for local clock source. 0b = Divide local clock source by 2. 1b = Divide local clock source by 4.
RefClockBSource	6	RW	0b	Set the reference clock "B" output source. The pass-through passes the clock "B" from previous EPL stage to the next stage unchanged. The EPLs are daisy-chained together from 0 through 23. If all EPLs are in pass-through, no reference clock is provided. Uses type *RefClockSource* (see Table 11-3).

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x45**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ErrWriteTx	8:7	RW	00b	Register file err_write interface to force uncorrectable errors on the transmit CDC FIFO (see SERDES_IP.*TxCdcFifoUErr* - Section 11.3.3.116). *ErrWriteTx* is XORed with the 2 least significant bits of the parity-protected data word.
ErrWriteRx	10:9	RW	00b	Register file err_write interface to force uncorrectable errors on the receive CDC FIFO (see SERDES_IP.*RxCdcFifoUErr* - Section 11.3.3.116). Goes through meta. Phase of the two bits is not guaranteed. *ErrWriteRx* is XORed with the 2 least significant bits of the parity-protected data word.
LoopbackRxToTx	11	RW	0b	This signal is ORed with the SERDES o_link_loopback_en. The loopback is implemented in the core clock domain. This signal is intended for internal debug. In general the loopback should be applied by configuring SERDES. 0b = Normal operation. 1b = Configure loopback from Receive to Transmit.
Reserved	31:12	RSV	0x0	Reserved.

11.3.3.109 LANE_SERDES_CFG[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x46**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**

CoreToCntl	15:0	RW	0x1126	SerDes input port i_core_to_cntl[15:0] Current bit assignment / i_core_to_cntl mapping:
				Bit[0] = PCS Fast bit slip (int 0x0c). Can invert every 80 UI (not used in auto slip mode).
				Bit[1] = Fast bit slip (int 0x0c). In auto slip mode, this can invert every 40 UI.
				Bit[2] = sel_pfd control (int 0x10).
				Bit[3] = tx_en / output_en (int 0x27).
				Bit[4] = rx_en (int 0x27).
				Bit[5] = rx_quiet (int 0x27).
				Bit[6] = tx_quiet (int 0x27).
				Bit[7] = lpi_disable (int 0x27).
				Bit[8] = Fast Tx polarity inversion (int 0x27).
				Bit[9:15] = Reserved.

Reserved	31:16	RSV	0x0	Reserved.

11.3.3.110 LANE_ENERGY_DETECT_CFG[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x47**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
EdOverride	1:0	RW	00b	Software is able to force the signal detect state using this field. Uses type *Override* (see Table 11-3).
EdMaskRxSignalOk	2	RW	0b	Mask SerDes Receive Signal Detect. 0b = Normal operation. 1b = Mask SerDes Receive Signal Detect (force OK).

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x47 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, EPL				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
EdMaskRxRdy	3	RW	0b	Mask SerDes rx_rdy signal. 0b = Normal operation. 1b = Mask (force OK).
EdMaskRxActivity	4	RW	0b	Mask Receive activity signal, which indicates there is activity on rx_data[9:0]. 0b = Normal operation. 1b = Mask (force OK).
EdMaskEnergyDetect	5	RW	0b	Mask SerDes Energy Detect signal, which indicates there is energy on the line, valid for EEE modes ***Note:*** The SerDes energy detect signal is used here not the forced value.
Reserved	31:6	RSV	0x0	Reserved.

11.3.3.111 LANE_ACTIVITY_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x48 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, EPL				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ActTime	5:0	RW	0xA	The SerDes rx_data[9:0] activity is recorded according to *ActMode*. If activity is recorded in (*ActTime* + 1 receive clock periods), the activity state progresses to AWAKE,. Otherwise, the activity state progresses to ASLEEP according to the activity state diagram.
ActMode	6	RW	0b	0b = Flatline — All zero are all one sequence activity = !(rx_data[9:0] == previous(rx_data[9:0])) and ((rx_data[9:0] == ALL_ZERO) or (rx_data[9:0] == ALL_ONE)) 1b = Static — rx_data[9:0] is not changing activity = !(rx_data[9:0] == previous(rx_data[9:0]) This mode is not suitable for 8B10B PCS types, e.g. 1000BASEX and SGMII
Reserved	31:7	RSV	0x0	Reserved.

11.3.3.112 LANE_SIGNAL_DETECT_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x49 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, EPL				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SdOverride	1:0	RW	00b	Software is able to force the signal detect state using this field. Uses type *Override* (see Table 11-3).
SdMaskRxSignalOk	2	RW	0b	Mask SerDes Receive Signal Detect. 0b = Normal operation. 1b = Mask SerDes Receive Signal Detect (force OK).
SdMaskRxRdy	3	RW	0b	Mask SerDes rx_rdy signal. 0b = Normal operation. 1b = Mask (force OK).
SdMaskRxActivity	4	RW	0b	Mask Receive activity signal, which indicates there is activity on rx_data[9:0]. 0b = Normal operation. 1b = Mask (force OK).

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x49
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
SdMaskEnergyDetect	5	RW	0b	Mask SerDes Energy Detect signal, which indicates there is energy on the line, valid for EEE modes ***Note:*** The SerDes energy detect signal is used here not the forced value.
Reserved	31:6	RSV	0x0	Reserved.

11.3.3.113 LANE_STATUS[0..8][0..3]

RxCdcFifo, *BistPass* and *BistFail* are only valid when *BistMode* is set to regular BIST, not pattern fill. Pattern Fill is not supported for *RxCdcFifo*.

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x4A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
RxSignalDetect	0	vRO	0b	Status of signal_detect.
RxEnergyDetect	1	vRO	0b	Status of energy_detect.
LoopbackRxToTx	2	vRO	0b	This path is selected when either v LANE_CFG.*LoopbackRxToTx* OR SERDES.*o_link_loopback_en.* 0b = Normal operation. 1b = Rx to Tx loopback path selected.
TxCdcFifoBistPass	3	vRO	0b	Transmit CDC FIFO BIST PASS status.
TxCdcFifoBistFail	4	vRO	0b	Transmit CDC FIFO BIST FAIL status.
RxCdcFifoBistPass	5	vRO	0b	Receive CDC FIFO BIST PASS status.
RxCdcFifoBistFail	6	vRO	0b	Receive CDC FIFO BIST FAIL status.
Reserved	31:7	RSV	0x0	Reserved.

11.3.3.114 LANE_SERDES_STATUS[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x4B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AnalogToCore	7:0	vRO	0x0	SerDes output port o_analog_to_core[7:0]. Miscellaneous ports.
CoreStatus	23:8	vRO	0x0	SerDes output port o_core_status[15:0]. Miscellaneous ports. *CoreStatus*[4]= signal_ok. Is optionally used for signal detect and/or energy detect.
RxRdy	24	vRO	0b	SerDes output port o_rx_rdy. When asserted (1b), the SerDes Rx PLL is calibrated, and the SerDes receiver is in a functional mode. Optionally used for signal detect and/or energy detect.
RxIdleDetect	25	vRO	0b	SerDes output port o_tx_rdy. When asserted (1b), the SerDes transmitter has calibrated, and is ready to send data presented on i_tx_in[39:0].
RxIdleDetect	26	vRO	0b	SerDes output port o_rx_idle_detect. When asserted (1b), the receiver has detected idle on SI_T/SI_C. This may be used for PCIe electrical idle, or 802.3az !energy_detect. This signal is valid in all power states. Optionally used for signal detect and/or energy detect.

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x4B**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RxActivity	27	vRO	0b	Indicates there is activity on SerDes rx_data[9:0]. Optionally used for signal detect and/or energy detect.
Reserved	31:28	RSV	0x0	Reserved.

11.3.3.115 SERDES_IM[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x4C**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CoreStatus0	0	RW	1b	Interrupt mask.
CoreStatus1	1	RW	1b	Interrupt mask.
CoreStatus2	2	RW	1b	Interrupt mask.
CoreStatus3	3	RW	1b	Interrupt mask.
RxSignalOk	4	RW	1b	Interrupt mask.
CoreStatus5	5	RW	1b	Interrupt mask.
CoreStatus6	6	RW	1b	Interrupt mask.
CoreStatus7	7	RW	1b	Interrupt mask.
CoreStatus8	8	RW	1b	Interrupt mask.
CoreStatus9	9	RW	1b	Interrupt mask.
CoreStatus10	10	RW	1b	Interrupt mask.
CoreStatus11	11	RW	1b	Interrupt mask.
CoreStatus12	12	RW	1b	Interrupt mask.
CoreStatus13	13	RW	1b	Interrupt mask.
CoreStatus14	14	RW	1b	Interrupt mask.
CoreStatus15	15	RW	1b	Interrupt mask.
TxRdy	16	RW	1b	Interrupt mask.
RxEnergyDetect	17	RW	1b	Interrupt mask.
RxSignalDetect	18	RW	1b	Interrupt mask.
RxRdy	19	RW	1b	Interrupt mask.
RxActivity	20	RW	1b	Interrupt mask.
RxIdleDetect	21	RW	1b	Interrupt mask.
SaiComplete	22	RW	1b	Interrupt mask.
SaiRequestError	23	RW	1b	Interrupt mask.
TxCdcFifoUErr	24	RW	1b	Interrupt mask.
TxCdcFifoError	25	RW	1b	Interrupt mask.
RxCdcFifoUErr	26	RW	1b	Interrupt mask.
RxCdcFifoError	27	RW	1b	Interrupt mask.
SlipRequest	28	RW	1b	Interrupt mask.
AnalogIp	29	RW	1b	Interrupt mask.
Reserved	31:30	RSV	00b	Reserved.

11.3.3.116 SERDES_IP[0..8][0..3]

Address:	**0x0E0000 + 0x400*j + 0x80*i + 0x4D**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, EPL**

Field Name	Bit(s)	Type	Default	Description
CoreStatus0	0	CW1	0b	General purpose SerDes interrupt.
CoreStatus1	1	CW1	0b	General purpose SerDes interrupt.
CoreStatus2	2	CW1	0b	General purpose SerDes interrupt.
CoreStatus3	3	CW1	0b	General purpose SerDes interrupt.
RxSignalOk	4	CW1	0b	SerDes signal OK. Mapped to *CoreStatus4*.
CoreStatus5	5	CW1	0b	General purpose SerDes interrupt.
CoreStatus6	6	CW1	0b	General purpose SerDes interrupt.
CoreStatus7	7	CW1	0b	General purpose SerDes interrupt.
CoreStatus8	8	CW1	0b	General purpose SerDes interrupt.
CoreStatus9	9	CW1	0b	General purpose SerDes interrupt.
CoreStatus10	10	CW1	0b	General purpose SerDes interrupt.
CoreStatus11	11	CW1	0b	General purpose SerDes interrupt.
CoreStatus12	12	CW1	0b	General purpose SerDes interrupt.
CoreStatus13	13	CW1	0b	General purpose SerDes interrupt.
CoreStatus14	14	CW1	0b	General purpose SerDes interrupt.
CoreStatus15	15	CW1	0b	General purpose SerDes interrupt.
TxRdy	16	CW1	0b	Transmit PLL ready.
RxEnergyDetect	17	CW1	0b	Filtered Energy Detect Interrupt pending.
RxSignalDetect	18	CW1	0b	Filtered Signal Detect Interrupt pending.
RxRdy	19	CW1	0b	Receive PLL ready.
RxActivity	20	CW1	0b	rx_data[9:0] activity.
RxIdleDetect	21	CW1	0b	Idle signal detection change on SerDes serial inputs (In EEE mode this is the inverse of energy_detect).
SaiComplete	22	CW1	0b	Completion of SerDes access. The interrupt is generated based on LANE_SAI_CFG.
SaiRequestError	23	CW1	0b	SerDes access error caused by requesting an interrupt while there is an interrupt in progress.
TxCdcFifoUErr	24	CW1	0b	Transmit CDC FIFO parity error detected.
TxCdcFifoError	25	CW1	0b	Receive FIFO overflow/underflow error.
RxCdcFifoUErr	26	CW1	0b	Receive CDC FIFO parity error detected.
RxCdcFifoError	27	CW1	0b	Receive FIFO overflow/underflow error.
SlipRequest	28	CW1	0b	Slip request detected.
AnalogIp	29	CW1	0b	Unmasked LANE_ANALOG_IP interrupt changed. This fires when an interrupt signal computed from LANE_ANALOG_IM and LANE_ANALOG_IP changes state. So this interrupt fires both when the interrupt is asserted and when it is negated. In this case it is recommended that the interrupt is serviced as follows: 1. Service and clear all LANE_ANALOG_IP interrupts. 2. Clear SERDES_IP.*AnalogIp*. 3. Verify that LANE_ANALOG_IP is zero, to avoid race condition.
Reserved	31:30	RSV	00b	Reserved.

11.3.3.117 LANE_ANALOG_IM[0..8][0..3]

Address: 0xE0000 + 0x400*j + 0x80*i + 0x4E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AnalogToCore0	0	RW	1b	Interrupt Mask.
AnalogToCore1	1	RW	1b	Interrupt Mask.
AnalogToCore2	2	RW	1b	Interrupt Mask.
AnalogToCore3	3	RW	1b	Interrupt Mask.
AnalogToCore4	4	RW	1b	Interrupt Mask.
AnalogToCore5	5	RW	1b	Interrupt Mask.
AnalogToCore6	6	RW	1b	Interrupt Mask.
AnalogToCore7	7	RW	1b	Interrupt Mask.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.118 LANE_ANALOG_IP[0..8][0..3]

Address: 0xE0000 + 0x400*j + 0x80*i + 0x4F
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
AnalogToCore0	0	CW1	0b	SerDes signal analog_to_core[0] event.
AnalogToCore1	1	CW1	0b	SerDes signal analog_to_core[1] event.
AnalogToCore2	2	CW1	0b	SerDes signal analog_to_core[2] event.
AnalogToCore3	3	CW1	0b	SerDes signal analog_to_core[3] event.
AnalogToCore4	4	CW1	0b	SerDes signal analog_to_core[4] event.
AnalogToCore5	5	CW1	0b	SerDes signal analog_to_core[5] event.
AnalogToCore6	6	CW1	0b	SerDes signal analog_to_core[6] event.
AnalogToCore7	7	CW1	0b	SerDes signal analog_to_core[7] event.
Reserved	31:8	RSV	0x0	Reserved.

11.3.3.119 LANE_SAI_CFG[0..8][0..3]

Address: 0x0E0000 + 0x400*j + 0x80*i + 0x50
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, EPL

Field Name	Bit(s)	Type	Default	Description
Code	15:0	RW	0x0	Interrupt Code. This field must not be programmed while LANE_SAI_STATUS.*InProgress* is set.
Data	31:16	RW	0x0	Interrupt Data. This field must not be programmed while LANE_SAI_STATUS.*InProgress* is set.
ResultPattern	47:32	RW	0x0	Pattern to compare with LANE_SAI_STATUS.*Result* for interrupt generation.

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x50** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ResultMode	49:48	RW	00b	Options for generating an interrupt: 00b = RESULT — Generate an interrupt when any LANE_SAI_STATUS.*Result* is returned. 01b = DISABLE — Do not generate an interrupt 10b = MATCH — Generate an interrupt when LANE_SAI_STATUS.*Result* == LANE_SAI_CFG.*ResultPattern*. 11b = NOT_MATCH — Generate an interrupt when LANE_SAI_STATUS.*Result* != LANE_SAI_CFG.*ResultPattern*.
Request	50	RW	0b	An access is triggered when this register bit changes from 0b to 1b, and LANE_SAI_STATUS.*InProgress* is 0b. LANE_SAI_STATUS.*RequestError* is set when this register bit changes from 0b to 1b, and LANE_SAI_STATUS.*InProgress* is 1b, in which case no access is performed.
Reserved	63:51	RSV	0x0	Reserved.

11.3.3.120 LANE_SAI_STATUS[0..8][0..3]

Address: **Atomicity:** **Reset Domains:**	**0x0E0000 + 0x400*j + 0x80*i + 0x52** **32** **DEVICE, MASTER, COLD, EPL**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Result	15:0	vRO	0x0	Result data.
Complete	16	vRO	0b	When set, indicates that the *Result* is valid.
AccessRequest	17	vRO	0b	SerDes access request status: 0b = Request is in progress. 1b = Service request is asserted.
InProgress	18	vRO	0b	Indicator of SerDes access status.
Busy	19	vRO	0b	Indicates there is either an access request pending or in progress.
RequestError	20	vRO	0b	This bit indicates that a requested interrupt has been ignored. This bit is set when LANE_SAI_CFG.*Request* bit changes from 0b to 1b and LANE_SAI_STATUS.*Busy* is 1b. There is no guarantee which interrupt is serviced, most likely the last one.
Reserved	31:21	RSV	0x0	Reserved.

11.4 PARSER Registers Description

11.4.1 PARSER Map

Table 11-4 Parser Register Set Map

Name	Address (Base = 0xCF0000)	Atomicity	Brief Description
PARSER_PORT_CFG_1[0..47]	Base + 0x2*i + 0x0	64	Parser options used by stage 1.
PARSER_PORT_CFG_2[0..47]	Base + 0x2*i + 0x80	64	Parser options used by stage 2 and deeper stages.
PARSER_PORT_CFG_3[0..47]	Base + 0x2*i + 0x100	64	Parser options used by stage 4 and deeper stages.
PORT_CFG_ISL[0..47]	Base + 0x1*i + 0x180	32	
PARSER_VLAN_TAG[0..3]	Base + 0x1*i + 0x1C0	32	Defines 4 VLAN Tag Protocol ID.
PARSER_CUSTOM_TAG[0..3]	Base + 0x1*i + 0x1C4	32	Defines four custom ETYPES.
PARSER_MPLS_TAG	Base + 0x1C8	32	Defines MPLS tag.
PARSER_DI_CFG[0..5]	Base + 0x2*i + 0x1D0	64	Defines DEEP inspection profile.
RX_VPRI_MAP[0..47]	Base + 0x2*i + 0x200	64	Defines mapping of received VLAN VPRI into an internal 4-bit VPRI code per port.
DSCP_PRI_MAP[0..63]	Base + 0x1*i + 0x280	32	Maps DSCP to switch priority.
VPRI_PRI_MAP[0..15]	Base + 0x1*i + 0x2C0	32	Maps VPRI to switch priority.

11.4.2 PARSER Registers

11.4.2.1 PARSER_PORT_CFG_1[0..47]

Parser options used by stage 1.

Address: 0xCF0000 + 0x2*i + 0x0
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
FTAG	0	RW	0b	Presence and format of F tags in frames sent or received over this port. 0b = NOT_TAGGED 1b = F56_TAG
Vlan1Tag	4:1	RW	0x0	Defines which VLAN TAGs are enabled for VLAN1. Vector with one bit for each entry of PARSER_VLAN_TAG.
Vlan2Tag	8:5	RW	0x0	Defines which VLAN TAGs are enabled for VLAN2. Vector with one bit for each entry of PARSER_VLAN_TAG.
Vlan2First	9	RW	0b	If two VLAN tags are present and identical, this bit defines which one is first. 0b = VLAN1 first, VLAN2 second. 1b = VLAN2 first, VLAN1 second.
defaultVID	21:10	RW	0x1	Default VLAN ID. Applied internally if *useDefaultVLAN*==1 or the frame is not VLAN tagged, or the VLAN tag has VID==0 (Priority tagged).

Address: Atomicity: Reset Domains:	0xCF0000 + 0x2*i + 0x0 64 DEVICE, MASTER, COLD, SWITCH			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
defaultVPRI	25:22	RW	0x0	Default VLAN user priority. Internal mapped 4-bit VPRI, which is regenerated to a TX VPRI on each egress port. Applied if *useDefaultVLAN*==1 or the frame is not VLAN tagged.
defaultVID2	37:26	RW	0x1	Default VLAN ID. Applied internally if the frame does not have a VLAN2 tag, or if the VLAN2 tag has VID==0 (Priority tagged).
defaultVPRI2	41:38	RW	0x0	Default VLAN user priority. Applied if the frame does not have a VLAN2 tag.
useDefaultVLAN	42	RW	0b	If 1b, overwrite the VID and VPRI with *defaultVID* and *defaultVPRI* regardless of whether the frame was VLAN tagged or not. This default is applied after SwitchPriority is chosen.
Reserved	63:43	RSV	0x0	Reserved.

11.4.2.2 PARSER_PORT_CFG_2[0..47]

Parser options used by stage 2 and deeper stages.

The various IPv4 and IPv6 options that can be flagged are all condensed to a single flag to the FFU, IPMISC.*TrapIPoptions*, that is set whenever a packet contains an IP option and the corresponding PARSER_PORT_CFG_2.*FlagIP** bit is set for that port. The PARSER does not distinguish between different options that have been selected to be flagged.

Address: Atomicity: Reset Domains:	0xCF0000 + 0x2*i + 0x80 64 SWITCH_MEM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CustomTag1	3:0	RW	0x0	Defines which custom TAGs are enabled.
CustomTag2	7:4	RW	0x0	Defines which custom TAGs are enabled.
ParseMPLS	8	RW	0b	Set if MPLS tags should be parsed.
StoreMPLS	11:9	RW	000b	Number of half-words in the outermost MPLS tags to save (up to 4), for use in the FFU. These words are stored L4D, L4C, L4B and L4A in that order.
ParseL3	12	RW	0b	Set if L3 (IPv4 and IPv6) frame headers should be parsed.
ParseL4	13	RW	0b	Set if L4 frame headers should be parsed.
FlagIPv4Options	14	RW	0b	Set if the presence of IPv4 options should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
FlagIPv6HopByHop	15	RW	0b	Set if the presence of IPv6 hop-by-hop option should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
FlagIPv6Routing	16	RW	0b	Set if the presence of IPv6 routing header should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
FlagIPv6Frag	17	RW	0b	Set if the presence of IPv6 fragment header should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
FlagIPv6Dest	18	RW	0b	Set if the presence of IPv6 destination options should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
FlagIPv6Auth	19	RW	0b	Set if the presence of IPv6 authentication header should be flagged in IPMISC.*TrapIPoptions*, for trap or FFU action.
defaultDSCP	25:20	RW	0x0	Default DSCP of this port. Applied internally when *useDefaultDSCP*==1 or the frame is not IP.

Address: 0xCF0000 + 0x2*i + 0x80
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
dropTagged	26	RW	0b	If 1b, drop tagged frames. That is, drop if the packet has been received with at least one VLAN tag and a VID different than 0. Not applicable if port is FTAGed; set to 0b in this case.
dropUntagged	27	RW	0b	If 1b, drop untagged frames. That is, drop if the packet has been received with no VLAN tag present or the VLAN tags present have their VIDs equal to 0. Not applicable if port is FTAGed; set to 0b in this case.
useDefaultDSCP	28	RW	0b	If 1b, overwrite incoming DSCP with defaultDSCP. This default is applied after SwitchPriority is chosen.
SwitchPriorityFromVLAN	29	RW	1b	If 1b and the frame is VLAN tagged, sets SwitchPriority from the mapped internal VPRI.
SwitchPriorityFromDSCP	30	RW	0b	If 1b and the frame is IP, map DSCP to SwitchPriority.
SwitchPriorityFromISL	31	RW	1b	If 1b and the frame is fabric tagged, use switch priority from Fabric tag as SwitchPriority regardless of *SwitchPriorityFromVLAN* or *SwitchPriorityFromDSCP*.
SwitchPriorityPrefersDSCP	32	RW	0b	If 1b, SwitchPriority prefers the DSCP mapped priority over VPRI.
Reserved	63:33	RSV	0x0	Reserved.

11.4.2.3 PARSER_PORT_CFG_3[0..47]

Parser options used by stage 4 and deeper stages.

Address: 0xCF0000 + 0x2*i + 0x100
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
PORT_DMAC	47:0	RW	0x0	Defines the port DMAC.
Reserved	63:48	RSV	0x0	Reserved.

11.4.2.4 PORT_CFG_ISL[0..47]

Address: 0xCF0000 + 0x1*i + 0x180
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
srcGlort	15:0	RW	0x0	Default source GLORT to assign to frames arriving on this port if the frames do not have a fabric tag.
USR	23:16	RW	0x0	Default USER bits to assign to frames arriving on this port if the frames do not have a fabric tag.
defaultPriority	27:24	RW	0x0	Default switch priority to assign to frames arriving on this port if the frames do not have a fabric tag.
Reserved	31:28	RSV	0x0	Reserved.

11.4.2.5 PARSER_VLAN_TAG[0..3]

Defines 4 VLAN Tag Protocol ID.

Address:	**0xCF0000 + 0x1*i + 0x1C0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Tag	15:0	RW	0x8100	L2 Ethernet type used to identify VLAN type.
Reserved	31:16	RSV	0x0	Reserved.

11.4.2.6 PARSER_CUSTOM_TAG[0..3]

Defines four custom ETYPES.

Address:	**0xCF0000 + 0x1*i + 0x1C4**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Tag	15:0	RW	0x0	L2 Ethernet type used to identify custom tags. Should be distinct from the standard Ethertypes 0x0800 (IPv4), 0x86dd (IPv6), 0x8808 (MAC Control), and the MPLS Ethertypes configured in PARSER_MPLS_TAG.
Capture	19:16	RW	0x0	Indicates which half-word to capture in the next four half-words following the tag. Setting to 0b means no capture. A maximum of two half-word could be captured.
CaptureSelect	20	RW	0b	Indicates if captures are in: 0b = L4A/B 1b = L4C/D
Length	25:21	RW	0x1	Length of the custom TAG in half-word including the Tag itself. Valid length is 1..9 (thus 1 to 18 bytes). Behavior undefined for any other value.
Reserved	31:26	RSV	0x0	Reserved.

11.4.2.7 PARSER_MPLS_TAG

Defines MPLS tag.

Address:	**0xCF0000 + 0x1C8**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Tag1	15:0	RW	0x8847	L2 Ethernet type used to identify MPLS tags.
Tag2	31:16	RW	0x8848	L2 Ethernet type used to identify MPLS tags.

11.4.2.8 PARSER_DI_CFG[0..5]

Defines DEEP inspection profile. The profile 0 is used when there is no match. If there are multiple match, the highest index is selected.

Address:	**0xCF0000 + 0x2*i + 0x1D0**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Prot	7:0	RW	0x0	Defines the protocol number to match.
L4Port	23:8	RW	0x0	Defines which port to capture.
WordOffset[0..7]	55:24	RW	0xF	(8 x 4-bits) Defines the word offset to capture for each DI field starting from the end of the L4 header. The L4 header length is defined in TCP HL field. It is 8 bytes for UDP, and is assumed to be 4 bytes for everything other than protocols. Setting the value to 15 is equivalent to not capturing this field. The maximum capturable fields are within the first 56 bytes of payload after the L4 header.
CaptureTCPFlags	56	RW	0b	When set, on TCP frames, the first deep inspection chunk (L4A) contains the TCP flags and header length, as copied from the TCP header.
Enable	57	RW	0b	Indicates if this filter is enabled.
L4Compare	58	RW	0b	Indicates if the L4 port is considered in the match or not.
Reserved	63:59	RSV	0x0	Reserved.

11.4.2.9 RX_VPRI_MAP[0..47]

This table defines mapping of received VLAN VPRI into an internal 4-bit VPRI code per port. The table is first indexed by the port number and then indexed by a 4-bit number, where bits [3:1] are set to the actual PCP received, and bit {0} is set to the actual DEI/CFI bit received. The table is not used if the packet is not VLAN tagged.

Address:	**0xCF0000 + 0x2*i + 0x200**
Atomicity:	**64**
Reset Domains:	**SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
pri0	3:0	RW	0x0	
pri1	7:4	RW	0x0	
pri2	11:8	RW	0x1	
pri3	15:12	RW	0x1	
pri4	19:16	RW	0x2	
pri5	23:20	RW	0x2	
pri6	27:24	RW	0x3	
pri7	31:28	RW	0x3	
pri8	35:32	RW	0x4	
pri9	39:36	RW	0x4	
pri10	43:40	RW	0x5	
pri11	47:44	RW	0x5	
pri12	51:48	RW	0x6	
pri13	55:52	RW	0x6	

Address: 0xCF0000 + 0x2*i + 0x200
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
pri14	59:56	RW	0x7	
pri15	63:60	RW	0x7	

11.4.2.10 DSCP_PRI_MAP[0..63]

Maps DSCP to switch priority.

Address: 0xCF0000 + 0x1*i + 0x280
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
pri	3:0	RW	0x0	SwitchPri to set for this DSCP.
Reserved	31:4	RSV	0x0	Reserved.

11.4.2.11 VPRI_PRI_MAP[0..15]

Maps VPRI to switch priority.

Address: 0xCF0000 + 0x1*i + 0x2C0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
pri	3:0	RW	0x0	Defines the switch priority associated with each VPRI.
Reserved	31:4	RSV	0x0	Reserved.

11.5 FFU Registers Description

11.5.1 FFU Map

Table 11-5 FFU Registers (reserved 512 KW) Map

Name	Address (Base = 0xC00000)	Atomicity	Brief Description
FFU_SLICE_TCAM[0..31][0..1023]	Base + 0x2000*j + 0x4*i + 0x0	128	Configures an entry in a slice's key TCAM.
FFU_SLICE_SRAM[0..31][0..1023]	Base + 0x2000*j + 0x2*i + 0x1000	64	Configures an entry in a slice's action SRAM.
FFU_SLICE_VALID[0..31]	Base + 0x2000*i + 0x1800	64	Configures which scenarios are valid in each slice.
FFU_SLICE_CASCADE_ACTION[0..31]	Base + 0x2000*i + 0x1804	64	Configures action cascade.
FFU_SLICE_CFG[0..31][0..31]	Base + 0x2000*j + 0x2*i + 0x1840	64	Configures slice.
FFU_MASTER_VALID	Base + 0x40000	64	Atomically set validity of all FFU slices.

11.5.2 FFU Enumerated Data Types

Table 11-6 FFU Enumerated Data Types

Name	Width	Description
FFU_MuxSelec	6	For IPv4 addresses, use Select_L3_DIP[31:0] or Select_L3_SIP[31:0]. 000000b = Select_MAP_DIP_MAP_SIP (8 bits) 000001b = Select_MAP_DMAC_MAP_SMAC (8 bits) 000010b = Select_MAP_PROT_MAP_LENGTH (8 bits) 000011b = Select_MAP_SRC_MAP_TYPE (8 bits) 000100b = Select_USER (8 bits) 000101b = Select_FTYPE_SWPRI (8 bits) 000110b = Select_IPMISC (8 bits) 000111b = Select_TOS (8 bits) 001000b = Select_PROT (8 bits) 001001b = Select_TTL (8 bits) 001010b = Select_SRC_PORT (8 bits) 001011b = Select_VPRI_VID_11_8 (8 bits) 001100b = Select_VID_7_0 (8 bits) 001101b = Select_RXTAG (8 bits) 001110b = Select_L2_DMAC[15:0] (16 bits) 001111b = Select_L2_DMAC[31:16] (16 bits) 010000b = Select_L2_DMAC[47:32] (16 bits) 010001b = Select_L2_SMAC[15:0] (16 bits) 010010b = Select_L2_SMAC[31:16] (16 bits) 010011b = Select_L2_SMAC[47:32] (16 bits) 010100b = Select_DGLORT (16 bits) 010101b = Select_SGLORT (16 bits) 010110b = Select_VPRI_VID (16 bits) 010111b = Select_VPRI2_VID2 (16 bits) 011000b = Select_L2_TYPE (16 bits) 011001b = Select_L4_DST (16 bits)

Table 11-6 FFU Enumerated Data Types [Continued]

Name	Width	Description
		011010b = Select_L4_SRC (16 bits) 011011b = Select_MAP_L4_DST (16 bits) 011100b = Select_MAP_L4_SRC (16 bits) 011101b = Select_L4A (16 bits) 011110b = Select_L4B (16 bits) 011111b = Select_L4C (16 bits) 100000b = Select_L4D (16 bits) 100001b = Select_MAP_VPRI1_VID1 (16 bits) 100010b = Select_L3_DIP[31:0] (32 bits) 100011b = Select_L3_DIP[63:32] (32 bits) 100100b = Select_L3_DIP[95:64] (32 bits) 100101b = Select_L3_DIP[127:96] (32 bits) 100110b = Select_L3_SIP[31:0] (32 bits) 100111b = Select_L3_SIP[63:32] (32 bits) 101000b = Select_L3_SIP[95:64] (32 bits) 101001b = Select_L3_SIP[127:96] (32 bits) All other values are reserved.

11.5.3 FFU Structures

11.5.3.1 FFU_SLICE_SRAM_ROUTE_ARP

Table 11-7 FFU_SLICE_SRAM_ROUTE_ARP Structure

Name	Bit(s)	Description
ArpIndex	15:0	Defines the base index in the ARP table for the first entry. Only lower 14 bits used.
ArpCount	19:16	Defines range of hashed ARP indices for ECMP or load balancing.
ArpType	20	When 0b, ARP range = *ArpCount*. When 1b, ARP range = 1<<*ArpCount* with a maximum range of 4096.
Reserved	31:21	Reserved.

11.5.3.2 FFU_SLICE_SRAM_ROUTE_GLORT

Table 11-8 FFU_SLICE_SRAM_ROUTE_GLORT Structure

Name	Bit(s)	Description
DGlort	15:0	Destination GLORT.
reserved	19:16	Reserved.
FloodSet	20	Ignored if DGlort is zero, in which case the destination GLORT results from the normal L2 lookup. If DGlort is non-zero, this bit operates as follows: 0b = Use DGlort as the destination GLORT, regardless of whether there is an L2 lookup hit or miss in the MA_TABLE. 1b = Use DGlort as the destination GLORT only if there is an L2 lookup miss in the MA_TABLE. If there is a hit, the destination GLORT is taken from the MA_TABLE.
Reserved	31:21	Reserved.

11.5.3.3 FFU_SLICE_SRAM_SET_BITS

This structure defines how the Bitset command is encoded for the command==BIT_SET case.

Table 11-9 FFU_SLICE_SRAM_SET_BITS Structure

Name	Bit(s)	Description
ByteMask	7:0	8-bit mask for FLAGS, TRIG, USR actions. FLAGS are: 00000001b = DROP 00000010b = TRAP 00000100b = LOG 00001000b = NO_ROUTE 00010000b = RX_MIRROR 00100000b = CAPTURE_TIME All other values are reserved.
ByteData	15:8	8-bit value for FLAGS, TRIG, USR actions.
SubCommand	20:16	Additional bits of control when Command==BitSet: 00001b = Change FLAG field. 00010b = Change TRIG field. 00100b = Change USR field. All other values are reserved.
Reserved	31:21	Reserved.

11.5.3.4 FFU_SLICE_SRAM_SET_VLAN

This structure defines how the SETVLAN command is encoded.

Table 11-10 FFU_SLICE_SRAM_SET_VLAN Structure

Name	Bit(s)	Description
VLAN	11:0	VLAN ID.
PRI	15:12	VLAN or software priority.
TxTag	17:16	Set transmit VLAN tag request. 00b = Use normal port tagging. 01b = Force adding VLAN tag. 10b = Force deleting VLAN tag if present. 11b = Force updating VLAN tag if present. Add if absent.
SetVPRI	18	Set VPRI field. If this field is set to 0b, VPRI not updated when this command is executed.
SetPRI	19	Set PRI field. If this field is set to 0b, VPRI not updated when this command is executed.
SubCommand	20	Must be set to 0b for SET_VLAN.
Reserved	31:21	Reserved.

11.5.3.5 FFU_SLICE_SRAM_SET_PRI

This structure defines how the SETDSCP command is encoded.

Table 11-11 FFU_SLICE_SRAM_SET_PRI Structure

Name	Bit(s)	Description
DSCP	5:0	DSCP Field.
Reserved0	11:6	Reserved. Set to 0b, not used.
PRI	15:12	VLAN or software priority.
Reserved1	16	Reserved. Set to 0b, not used.
SetDSCP	17	Set DSCP field. If this field is set to 0b, DSCP not updated when this command is executed.
SetVPRI	18	Set VPRI field. If this field is set to 0b, VPRI not updated when this command is executed.
SetPRI	19	Set PRI field. If this field is set to 0b, VPRI not updated when this command is executed.
SubCommand	20	Must be set to 1b for SET_DSCP.
Reserved	31:21	Reserved.

11.5.4 FFU Registers

11.5.4.1 FFU_SLICE_TCAM[0..31][0..1023]

Configures an entry in the TCAM. The TCAM entries are readable but the key returned is masked out according to the mask originally written along with the key.

Note: The case bit is ORed with !Valid before being written into TCAM

All TCAM entries start out as invalid. Each entry becomes valid on the first write to that entry, to either the *Key* or *KeyInvert* data.

A match occurs when the data searched is equal to the content, as defined in the table below. Only valid entries are included in the compare.

A key is loaded by writing the key into 'Key' and loading its 1s compliment in *KeyInvert*. Writing both '*Key*' and '*KeyInvert*' with 1b on same bit in both fields invalidates the entry while writing 0b on same bit in both field is equivalent of ignoring this bit in the search.

The table below shows the encoded data value stored (Content) based on the state of the data (*Key*) and mask (*KeyInvert*) values in the TCAM bitcell.

Mask Storage	Data Storage	Content
0	0	x (always match)
0	1	1
1	0	0
1	1	Always mismatch state

The highest index has precedence in case there is more than one match in the same slice.

Address: 0xC00000 + 0x2000*j + 0x4*i + 0x0
Atomicity: 128
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Key	31:0	RW	0x0	
KeyTop	39:32	RW	0x0	
rsvd0	63:40	RSV	0x0	Reserved.
KeyInvert	95:64	RW	0x0	
KeyTopInvert	103:96	RW	0x0	
rsvd1	127:104	RSV	0x0	Reserved.

11.5.4.2 FFU_SLICE_SRAM[0..31][0..1023]

Address: 0xC00000 + 0x2000*j + 0x2*i + 0x1000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Data	20:0	RW	0x0	Encoding depends on command. See structures: • FFU_SLICE_SRAM_ROUTE_ARP • FFU_SLICE_SRAM_ROUTE_GLORT • FFU_SLICE_SRAM_SET_BITS • FFU_SLICE_SRAM_SET_VLAN • FFU_SLICE_SRAM_SET_PRI
Command	22:21	RW	00b	Command encoding: 00b = ROUTE_ARP 01b = ROUTE_GLORT 10b = BIT_SET 11b = FIELD_SET
CounterIndex	34:23	RW	0x0	Counter or Policer index.
CounterBank	36:35	RW	00b	Counter or Policer bank.
Precedence	39:37	RW	000b	Precedence of actions.
Reserved	63:40	RSV	0x0	Reserved.

11.5.4.3 FFU_SLICE_VALID[0..31]

FFU_MASTER_VALID bit for this slice must also be true. If invalid for either reason, no rules in the slice can hit, and all comparisons or actions are cascaded through unmodified.

Address: 0xC00000 + 0x2000*i + 0x1800
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Valid	31:0	RW	0x0	Valid bit-mask for each of 32 scenarios.

11.5.4.4 FFU_SLICE_CASCADE_ACTION[0..31]

Address:	**0xC00000 + 0x2000*i + 0x1804**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CascadeAction	31:0	RW	0x0	*CascadeAction* bit-mask for each of 32 scenarios.

11.5.4.5 FFU_SLICE_CFG[0..31][0..31]

Configure each slice behavior for each scenario. This register is indexed by slice number and scenario (FFU_SLICE_CFG[slice][scenario]) and provides configuration for the cascading and mux selection.

The keys are defined in Section 11.5.4.1.

Address:	**0xC00000 + 0x2000*j + 0x2*i + 0x1840**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Select0	5:0	RW	0x0	Selects byte 0 of the lookup key. Uses type *FFU_MuxSelect* (see Table 11-6).
Select1	11:6	RW	0x0	Selects byte 1 of the lookup key. Uses type *FFU_MuxSelect* (see Table 11-6).
Select2	17:12	RW	0x0	Selects byte 2 of the lookup key. Uses type *FFU_MuxSelect* (see Table 11-6).
Select3	23:18	RW	0x0	Selects byte 3 of the lookup key. Uses type *FFU_MuxSelect* (see Table 11-6).
SelectTop	29:24	RW	0x0	Selects top 8 bits of the lookup key. Only KEY8s can be selected for this field. Uses type *FFU_MuxSelect* (see Table 11-6).
StartCompare	30	RW	0b	Starts a compare cascade.
StartAction	31	RW	0b	Starts a action cascade.
ValidLow	32	RW	0b	Specifies if the lower 512 rules of the CAM has any valid entry. Used to reduce power.
ValidHigh	33	RW	0b	Specifies if the upper 512 rules of the CAM has any valid entry. Used to reduce power.
Case_*XXX*	37:34	RW	0x0	Defines a 4-bit case for each scenario.
SetCaseLocation	39:38	RW	00b	Defines Case location in TOP 8 bits of key. 00b = NOT_MAPPED — Case not mapped. 01b = TOP_LOW_NIBBLE — Case mapped to KeyTop[3:0]. 10b = TOP_HIGH_NIBBLE — Case mapped to KeyTop[7:4]. 11b = Reserved.
Reserved	63:40	RSV	0x0	Reserved.

11.5.4.6 FFU_MASTER_VALID

Used to allow atomic and non-disruptive changes of ingress and egress ACL's. Writing this register atomically disables certain slices or chunks and enables other slices or chunks. Applies to all scenarios. Invalid slices pass all cascaded hit, index, raw hit, and actions through unmodified to the next slice. Invalid chunks do likewise.

Address: 0xC00000 + 0x40000 **Atomicity:** 64 **Reset Domains:** DEVICE, MASTER, COLD, SWITCH				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SliceValid	31:0	RW	0xFFFFFFFF	Bit mask indicating all valid FFU slices.
ChunkValid	63:32	RW	0xFFFFFFFF	Bit mask indicating all valid egress chunks.

11.6 FFU_MAP Registers Description

11.6.1 FFU_MAP Map

Table 11-12 FFU_MAP Registers (reserved 8 KW) Map

Name	Address (Base = 0xDA0000)	Atomicity	Brief Description
FFU_MAP_VLAN[0..4095]	Base + 0x1*i + 0x0	32	Configures 12-bit to 12-bit VLAN mapping.
FFU_MAP_SRC[0..47]	Base + 0x1*i + 0x1000	32	Configures source port to 4-bit mapping.
FFU_MAP_MAC[0..31]	Base + 0x2*i + 0x1040	64	Configures 48-bit to 4-bit MAC address mapping.
FFU_MAP_TYPE[0..15]	Base + 0x1*i + 0x1080	32	Configures 16-bit to 4-bit EtherType mapping.
FFU_MAP_LENGTH[0..15]	Base + 0x1*i + 0x1090	32	Configures 16-bit to 4-bit IP Length mapping.
FFU_MAP_IP_LO[0..15]	Base + 0x2*i + 0x10A0	64	Configures low 64-bit key of 128-bit to 4-bit IP address mapping.
FFU_MAP_IP_HI[0..15]	Base + 0x2*i + 0x10C0	64	Configures high 64-bit key of 128-bit to 4-bit IP address mapping.
FFU_MAP_IP_CFG[0..15]	Base + 0x1*i + 0x10E0	32	Configures 128-bit to 4-bit IP address mapping.
FFU_MAP_PROT[0..7]	Base + 0x1*i + 0x10F0	32	Configures 8-bit to 3-bit L4 Protocol mapping.
FFU_MAP_L4_SRC[0..63]	Base + 0x2*i + 0x1100	64	Configures 16-bit to 4-bit L4 source port mapping.
FFU_MAP_L4_DST[0..63]	Base + 0x2*i + 0x1180	64	Configures 16-bit to 4-bit L4 destination port mapping.

11.6.2 FFU_MAP Registers

11.6.2.1 FFU_MAP_VLAN[0..4095]

Address: **Atomicity:** **Reset Domains:**	**0xDA0000 + 0x1*i + 0x0** **32** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MAP_VLAN	11:0	RW	0x0	The mapped VLAN.
Routable	12	RW	0b	Enable IP routing on this VLAN.
Reserved	31:13	RSV	0x0	Reserved.

11.6.2.2 FFU_MAP_SRC[0..47]

Address: **Atomicity:** **Reset Domains:**	**0xDA0000 + 0x1*i + 0x1000** **32** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MAP_SRC	3:0	RW	0x0	Resulting *MAP_SRC* per source port.
Routable	4	RW	0b	Enable IP routing on this source port.
Reserved	31:5	RSV	0x0	Reserved.

11.6.2.3 FFU_MAP_MAC[0..31]

Address: **0xDA0000 + 0x2*i + 0x1040**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
MAC	47:0	RW	0x0	Ethernet MAC Key. Ignored bits must be set to 0b.
IgnoreLength	53:48	RW	0x0	Ignore this many LSB bits when matching.
validSMAC	54	RW	0b	Entry valid for MAP_SMAC.
validDMAC	55	RW	0b	Entry valid for MAP_DMAC.
MAP_MAC	59:56	RW	0x0	4-bit mapped MAC.
Router	60	RW	0b	Is this a virtual router destination MAC?
Reserved	63:61	RSV	000b	Reserved.

11.6.2.4 FFU_MAP_TYPE[0..15]

Address: **0xDA0000 + 0x1*i + 0x1080**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
TYPE_*XXX*	15:0	RW	0x0	EtherType Key.
MAP_TYPE	19:16	RW	0x0	4-bit mapped EtherType.
Reserved	31:20	RSV	0x0	Reserved.

11.6.2.5 FFU_MAP_LENGTH[0..15]

Address: **0xDA0000 + 0x1*i + 0x1090**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
LENGTH	15:0	RW	0x0	IP Packet Length Key.
MAP_LENGTH	19:16	RW	0x0	4-bit *MAP_LENGTH*.
Reserved	31:20	RSV	0x0	Reserved.

11.6.2.6 FFU_MAP_IP_LO[0..15]

Address: **0xDA0000 + 0x2*i + 0x10A0**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
IP_LO	63:0	RW	0x0	Low 64 bits of IP address prefix key. Ignored bits must be set to 0b.

11.6.2.7 FFU_MAP_IP_HI[0..15]

Address:	0xDA0000 + 0x2*i + 0x10C0
Atomicity:	64
Reset Domains:	DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
IP_HI	63:0	RW	0x0	High 64 bits of IP address prefix key. Ignored bits must be set to 0b.

11.6.2.8 FFU_MAP_IP_CFG[0..15]

Address:	0xDA0000 + 0x1*i + 0x10E0
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
IgnoreLength	7:0	RW	0x0	Ignore this many LSB bits when matching.
validSIP	8	RW	0b	Entry valid for MAP_SIP.
validDIP	9	RW	0b	Entry valid for MAP_DIP.
MAP_IP	13:10	RW	0x0	4-bit mapped IP address.
Reserved	31:14	RSV	0x0	Reserved.

11.6.2.9 FFU_MAP_PROT[0..7]

Address:	0xDA0000 + 0x1*i + 0x10F0
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
PROT	7:0	RW	0x0	Input IP protocol number.
MAP_PROT	10:8	RW	000b	3-bit mapped IP protocol number.
Reserved	31:11	RSV	0x0	Reserved.

11.6.2.10 FFU_MAP_L4_SRC[0..63]

Address:	0xDA0000 + 0x2*i + 0x1100
Atomicity:	64
Reset Domains:	DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
L4SRC	15:0	RW	0x0	L4 source port key.
MAP_PROT	18:16	RW	000b	3-bit *MAP_PROT*.
VALID	19	RW	0b	Valid mapping.
Reserved	31:20	RSV	0x0	Reserved
MAP_L4SRC	47:32	RW	0x0	Mapped result.
Reserved	63:48	RSV	0x0	Reserved.

11.6.2.11 FFU_MAP_L4_DST[0..63]

Address:	**0xDA0000 + 0x2*i + 0x1180**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
L4SRC	15:0	RW	0x0	L4 destination port key.
MAP_PROT	18:16	RW	000b	3-bit *MAP_PROT*.
VALID	19	RW	0b	Valid mapping.
Reserved	31:20	RSV	0x0	Reserved.
MAP_L4DST	47:32	RW	0x0	Mapped result.
Reserved	63:48	RSV	0x0	Reserved.

11.7 EACL Registers Description

11.7.1 EACL Map

Table 11-13 Egress ACL Registers (reserved 4 KW) Map

Name	Address (Base = 0xDB0000)	Atomicity	Brief Description
FFU_EGRESS_CHUNK_ACTIONS[0..31]	Base + 0x1*i + 0x0	32	Egress ACL actions of each chunk.
FFU_EGRESS_CHUNK_VALID[0..31]	Base + 0x1*i + 0x20	32	Configures valid scenarios for 16-rule chunks of egress ACLs.
FFU_EGRESS_CHUNK_CFG[0..31]	Base + 0x1*i + 0x40	32	Configures Egress ACL chunks.
FFU_EGRESS_PORT_CFG[0..47]	Base + 0x1*i + 0x80	32	Egress ACLs Port Configuration.
FFU_EGRESS_DROP_COUNT[0..47]	Base + 0x2*i + 0x100	64	Egress ACLs Drop Count.

11.7.2 EACL Registers

11.7.2.1 FFU_EGRESS_CHUNK_ACTIONS[0..31]

Defines actions for each rule of each chunk.

Address:	**0xDB0000 + 0x1*i + 0x0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Drop	31:0	RW	0x0	Drop packets on selected rules.

11.7.2.2 FFU_EGRESS_CHUNK_VALID[0..31]

Contains a *Valid* bit for each of 32 possible scenarios. The FFU_MASTER_VALID bit for this chunk must also be true. If invalid for either reason, no rules in the chunk can hit, and any cascaded hit ripples through unmodified.

Address:	**0xDB0000 + 0x1*i + 0x20**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Valid	31:0	RW	0xFFFFFFFF	Valid bit-mask for each of 32 scenarios.

11.7.2.3 FFU_EGRESS_CHUNK_CFG[0..31]

Address:	**0xDB0000 + 0x1*i + 0x40**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
StartCascade	0	RW	0b	Starts a priority hit cascade across chunks.
Reserved	31:1	RSV	0x0	Reserved.

11.7.2.4 FFU_EGRESS_PORT_CFG[0..47]

Address:	**0xDB0000 + 0x1*i + 0x80**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DropApply	31:0	RW	0x0	Indicates which chunk drop action applies per port.

11.7.2.5 FFU_EGRESS_DROP_COUNT[0..47]

Address:	**0xDB0000 + 0x2*i + 0x100**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Frames	47:0	RW	0x0	Number of frames dropped due to Egress ACL drop action.
Reserved	63:48	RSV	0x0	Reserved.

11.8 ARP Registers Description

11.8.1 ARP Map

Table 11-14 ARP Control and Table (128 KW) Map

Name	Address (Base = 0xCC0000)	Atomicity	Brief Description
ARP_TABLE[0..16383]	Base + 0x2*i + 0x0	64	Stores next-hop information for IP routing.
ARP_USED[0..511]	Base + 0x1*i + 0x8000	32	Tracks if hardware has used each ARP entry.

11.8.2 ARP Structures

11.8.2.1 ARP_ENTRY_DMAC

Table 11-15 ARP_ENTRY_DMAC Structure

Name	Bit(s)	Description
DMAC	47:0	Destination MAC address.
EVID	59:48	Egress VLAN ID. Set to zero when using the GLORT type for IP multicast.
RouterId	63:60	Setting to 15 means do not replace RouterID. Setting to 0b means to use the GLORT encoding.

11.8.2.2 ARP_ENTRY_GLORT

Table 11-16 ARP_ENTRY_GLORT Structure

Name	Bit(s)	Description
DGLORT	15:0	Destination GLORT.
MTU_Index	18:16	Defines the MTU size for this VLAN through a 3-bit index into MTU_TABLE. This supersede the MTU size index defined in the EGRESS_VID_TABLE.
markRouted	19	0b = None of these changes are made, the egress VLAN stays as the ingress VLAN, and the frame is switched to the assigned destination GLORT. 1b = The frame is routed to this GLORT and egress VLAN for use in IP multicast. This causes the frame to be marked as routed in the FTAG, checked against MTU, TTL, ARP redirect as well as modified for routing on transmit (DMAC, SMAC, and TTL decrement).
Reserved	42:20	Reserved.
IPv6Entry	43	Set to one to derive MAC entry from IPv6, and ignore DGLORT, MTU_Index, and *markRouted* for this entry (treat it like ARP_ENTRY_DMAC). Use the RouterIdGlort for the routerId.
RouterIdGlort	47:44	Setting to 15 means do not replace RouterID.
EVID	59:48	Egress VLAN ID. Set to zero when using the GLORT type for IP multicast.
RouterId	63:60	Set to zero to encode this mode.

11.8.3 ARP Registers

11.8.3.1 ARP_TABLE[0..16383]

Address:	**0xCC0000 + 0x2*i + 0x0**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
entry	47:0	RW	0x0	Encoding depends on command. See structures: • ARP_ENTRY_DMAC • ARP_ENTRY_GLORT
EVID	59:48	RW	0x0	Egress VLAN ID used in the route. Set to zero when using the GLORT type for IP multicast.
RouterId	63:60	RW	0x0	0000b = Use the GLORT encoding. 0001b–1110b = Use the DMAC encoding. 1111 = Use the DMAC encoding, but do not replace RouterID.

11.8.3.2 ARP_USED[0..511]

Each time ARP_TABLE[index] entry is used to forward a frame, the chip sets the ARP_USED[index{13:5}]{index{4:0}} bit to 1b. When software writes a word to this table, any bits that are 1b in the word clear the corresponding bit in ARP_USED to 0b. The software should read a word into x, record which entries have been used since the last sample, and then write x back to ARP_USED to clear the bits. This avoids any race between the hardware and software. The concept is similar to an interrupt register.

Address:	**0xCC0000 + 0x1*i + 0x8000**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
used	31:0	vCW1	0x0	32 consecutive bits marking usage of corresponding ARP_TABLE entries.

11.9 POLICER_APPLY Registers Description

11.9.1 POLICER_APPLY Map

Table 11-17 Policer Registers (reserved 64 KW) Map

Name	Address (Base = 0xD60000)	Atomicity	Brief Description
POLICER_APPLY_CFG_4K[0..1][0..4095]	Base + 0x1000*j + 0x1*i + 0x0	32	4 K x 2-bits. Configurable action per policer entry.
POLICER_APPLY_CFG_512[0..1][0..511]	Base + 0x200*j + 0x1*i + 0x2000	32	512 x 2-bits. Configurable action per policer entry.
POLICER_DSCP_DOWN_MAP[0..63]	Base + 0x1*i + 0x6400	32	
POLICER_SWPRI_DOWN_MAP[0..15]	Base + 0x1*i + 0x6440	32	

11.9.2 POLICER_APPLY Registers

11.9.2.1 POLICER_APPLY_CFG_4K[0..1][0..4095]

Address: 0xD60000 + 0x1000*j + 0x1*i + 0x0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
CommittedAction	0	RW	0b	Drop if egress color is yellow, one bit per policer entry.
ExcessAction	1	RW	0b	Drop if egress color is red, one bit per policer entry.
Reserved	31:2	RSV	0x0	Reserved.

11.9.2.2 POLICER_APPLY_CFG_512[0..1][0..511]

Address: 0xD60000 + 0x200*j + 0x1*i + 0x2000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
CommittedAction	0	RW	0b	Drop if egress color is yellow, one bit per policer entry.
ExcessAction	1	RW	0b	Drop if egress color is red, one bit per policer entry.
Reserved	31:2	RSV	0x0	Reserved.

11.9.2.3 POLICER_DSCP_DOWN_MAP[0..63]

Address: **0xD60000 + 0x1*i + 0x6400**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
NewDSCP	5:0	RW	0x0	Defines new downgrade DSCP. Also used to define coloring.
Reserved	31:6	RSV	0x0	Reserved.

11.9.2.4 POLICER_SWPRI_DOWN_MAP[0..15]

Address: **0xD60000 + 0x1*i + 0x6440**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
NewSwitchPriority	3:0	RW	0x0	Defines new downgrade Switch Priority. Also used to define coloring.
Reserved	31:4	RSV	0x0	Reserved.

11.10 POLICER_USAGE Registers Description

11.10.1 POLICER_USAGE Map

Table 11-18 Policer Registers (reserved 64 KW) Map

Name	Address (Base = 0xE40000)	Atomicity	Brief Description
POLICER_CFG_4K[0..1][0..4095]	Base + 0x2000*j + 0x2*i + 0x0	64	4 K x 64-bits. Policer bank configuration.
POLICER_CFG_512[0..1][0..511]	Base + 0x400*j + 0x2*i + 0x4000	64	512 x 64-bits. Policer bank configuration.
POLICER_STATE_4K[0..1][0..4095]	Base + 0x4000*j + 0x4*i + 0x8000	64	4 K x 64-bits. Policer bank state.
POLICER_STATE_512[0..1][0..511]	Base + 0x800*j + 0x4*i + 0x10000	64	512 x 64-bits. Policer bank state.
POLICER_CFG[0..3]	Base + 0x2*i + 0x11000	64	Policer banks configuration.
POLICER_SWEEPER_PERIOD_CFG	Base + 0x11008	64	Sweeper configuration.

11.10.2 POLICER_USAGE Structures

11.10.2.1 POLICER_CFG_ENTRY

Table 11-19 POLICER_CFG_ENTRY Structure

Name	Bit(s)	Description
RateMantissa	3:0	Policer rate is expressed in floating point form in units of bytes or frames per sweeper_period, depending on Unit. `Rate = Mantissa << Exponent` ***Note:*** Out-of-bounds Rate values (larger than 2^31-1) result in undefined behavior.
RateExponent	8:4	Policer rate exponent.
CapacityMantissa	12:9	Policer token bucket capacity mantissa, in units of bytes or frames depending on Unit. `Capacity = Mantissa << Exponent` Capacity must be set equal-or-larger to Rate * sweeper_period.
CapacityExponent	17:13	Policer token bucket capacity exponent.
Reserved	31:18	Reserved

11.10.3 POLICER_USAGE Registers

11.10.3.1 POLICER_CFG_4K[0..1][0..4095]

Address:	**0xE40000 + 0x2000*j + 0x2*i + 0x0**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Committed	17:0	RW	0x0	Rate configuration for committed token bucket (refer to Section 11.10.2.1).
Reserved	31:18	RSV	0x0	Reserved.
Excess	49:32	RW	0x0	Rate configuration for excess token bucket (refer to Section 11.10.2.1).
Reserved	63:50	RSV	0x0	Reserved.

11.10.3.2 POLICER_CFG_512[0..1][0..511]

Address:	**0xE40000 + 0x400*j + 0x2*i + 0x4000**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Committed	17:0	RW	0x0	Rate configuration for committed token bucket (refer to Section 11.10.2.1).
Reserved	31:18	RSV	0x0	Reserved.
Excess	49:32	RW	0x0	Rate configuration for excess token bucket (refer to Section 11.10.2.1).
Reserved	63:50	RSV	0x0	Reserved.

11.10.3.3 POLICER_STATE_4K[0..1][0..4095]

Address:	**0xE40000 + 0x4000*j + 0x4*i + 0x8000**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Count1	63:0	vRW	0x0	Count for entry 1 (in units of frames when counting). Represents an absolute unsigned rollover count (Mode==0), or a signed token count that is decremented as frames are credited to the entry and incremented by Rate every sweeper period.
Count2	127:64	vRW	0x0	Count for entry 2 (in units of bytes when counting).

11.10.3.4 POLICER_STATE_512[0..1][0..511]

Address: 0xE40000 + 0x800*j + 0x4*i + 0x10000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Count1	63:0	vRW	0x0	Count for entry 1 (in units of frames when counting). Represents an absolute unsigned rollover count (Mode==0), or a signed token count that is decremented as frames are credited to the entry and incremented by Rate every sweeper period.
Count2	127:64	vRW	0x0	Count for entry 2 (in units of bytes when counting).

11.10.3.5 POLICER_CFG[0..3]

Address: 0xE40000 + 0x2*i + 0x11000
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
IndexLastPolicer	11:0	RW	0x0	Defines the index of the last policer. Setting this value to 0b means there are no policers in this bank. Must be set to less than 512 for banks 2 and 3.
IngressColorSource	13:12	RW	00b	Defines what defines the color. The choices are: 00b = Reserved 01b = Reserved 10b = NoColor (assume green) 11b = Reserved.
MarkDSCP	14	RW	0b	Defines if DSCP field is changed when color is changed or not.
MarkSwitchPri	15	RW	0b	Defines if Switch Priority is changed when color is changed or not.
Reserved	31:16	RSV	0x0	Reserved.

11.10.3.6 POLICER_SWEEPER_PERIOD_CFG

Address: 0xE40000 + 0x11008
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Period	19:0	RW	0x11FD4	Number of idle cycles between sweeper operations.
SweeperEnable	20	RW	0b	Enable the policer sweeper. This must be set to 1b for normal policer operation.
Reserved	31:21	RSV	0x0	Reserved.

11.11 L2LOOKUP Registers Description

11.11.1 L2LOOKUP Map

Table 11-20 Layer 2 Lookup Registers (reserved 256 KW) Map

Name	Address (Base = 0xC80000)	Atomicity	Brief Description
MA_TABLE[0..1][0..16383]	Base + 0x10000*j + 0x4*i + 0x0	64	MAC address forwarding table.
INGRESS_VID_TABLE[0..4095]	Base + 0x4*i + 0x20000	64	Ingress VLAN table.
EGRESS_VID_TABLE[0..4095]	Base + 0x4*i + 0x24000	64	Egress VLAN table.
MA_USED_TABLE[0..1][0..511]	Base + 0x200*j + 0x1*i + 0x28000	32	MAC address used table.
INGRESS_MST_TABLE[0..1][0..255]	Base + 0x200*j + 0x2*i + 0x28400	64	Ingress spanning-tree forwarding state table.
EGRESS_MST_TABLE[0..255]	Base + 0x2*i + 0x28800	64	Ingress spanning-tree forwarding state table.
MA_TABLE_CFG_1	Base + 0x28A00	32	MA_TABLE Configuration.
MA_TABLE_CFG_2	Base + 0x28A02	64	GloRTs for flooding, broadcast, and multicast.
MTU_TABLE[0..7]	Base + 0x1*i + 0x28A08	32	MA Table frame watermark.
IEEE_RESERVED_MAC_ACTION	Base + 0x28A10	64	Reserved multicast MAC address action.
IEEE_RESERVED_MAC_TRAP_PRIORITY	Base + 0x28A14	64	Reserved multicast MAC addresses trap priority select.
IEEE_RESERVED_MAC_CFG	Base + 0x28A16	32	Reserved multicast MAC addresses configuration.

11.11.2 L2LOOKUP Registers

11.11.2.1 MA_TABLE[0..1][0..16383]

The MAC address forwarding table is in two parts; MA_TABLE[0,*] is for DMAC lookup, while MA_TABLE[1,*] is for SMAC lookup. The tables are 4-way tables indexed using four distinct hash keys derived from {fid,mac}.

Address:	**0xC80000 + 0x10000*j + 0x4*i + 0x0**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MACAddress	47:0	RW	0x0	MAC Address.
FID	59:48	RW	0x0	FID — Learning Group. In shared spanning tree mode, FID = 0b. In multiple spanning tree mode, FID = VID.
valid	60	RW	0b	Indicates if entry is in use. A lookup hits only if valid is 1b.

Address: **Atomicity:** **Reset Domains:**	**0xC80000 + 0x10000*j + 0x4*i + 0x0** **64** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
secure	61	RW	0b	SMAC entry is secure and does not have MacMoved events reported to the TCN FIFO. Frames with this SMAC arriving on a port different from the entry's glort field are silently dropped, though this behavior may be overridden by a trigger action. Ignored on DMAC lookups.
Reserved	63:62	RSV	00b	Reserved.
glort	79:64	RW	0x0	The GLORT associated with the entry. The destination GLORT for DMAC lookups MA_TABLE[0,*] and the source GLORT for SMAC lookups MA_TABLE[0,*]. For SMAC lookups, this field is compared with the frame's source GLORT to identify MAC moves.
trigId	85:80	RW	0x0	Each trigger has a MAC address ID for source address and one for destination address, as defined in TRIGGER_CONDITION_PARAM, that may be compared to this field. The trigger may match the source MAC address trigId, the destination MAC address trigId (which may be different than the source's) or both. ***Note:*** Two trigger IDs are pre-assigned and cannot be changed: 0x0 and 0x3F. See TRIGGER_CONDITION_PARAM (Section 11.17.3.2).
Reserved	127:86	RSV	0x0	Reserved.

11.11.2.2 INGRESS_VID_TABLE[0..4095]

Address: **Atomicity:** **Reset Domains:**	**0xC80000 + 0x4*i + 0x20000** **64** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
membership	47:0	RW	0x0	Defines port membership for this VLAN. A 1b in each bit position includes that port in the VLAN membership.
Reserved.	63:48	RSV	0x0	Reserved.
FID	75:64	RW	0x0	Learning group for this VLAN.
MST_Index	83:76	RW	0x0	Multiple spanning tree index for this ingress VLAN.
CounterIndex	89:84	RW	0x0	Specifies the RX stats bank 6 counter index to update as frames on this VLAN are received by the switch.
reflect	90	RW	0b	If this bit is set and PORT_CFG_2 destination mask include itself, layer 2 packets associated with this VLAN are allowed to go back on the port they come from (subject to LOOPBACK_SUPPRESS registers). If this bit is not set, layer 2 packets associated with this VLAN are not allowed to go back on the port they come from. Routed packets are always allowed to go back where they come from and this bit is not used in this case.
TrapIGMP	91	RW	0b	Indicates if the IGMP frames on this VLAN are trapped or switched normally. The IGMP frames are trapped to the CPU GLORT.
Reserved	127:92	RSV	0x0	Reserved.

11.11.2.3 EGRESS_VID_TABLE[0..4095]

Address:	0xC80000 + 0x4*i + 0x24000
Atomicity:	64
Reset Domains:	SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
membership	47:0	RW	0x0	Defines if this port is member of this VLAN or not.
Reserved	63:48	RSV	0x0	Reserved.
FID	75:64	RW	0x0	Learning group for this VLAN.
MST_Index	83:76	RW	0x0	Multiple spanning tree index for this egress VLAN.
MTU_Index	86:84	RW	000b	Defines the MTU size for this VLAN through a 3-bit index into MTU_TABLE. The MTU is checked only for routed frames, and the checking is done against the packet length defined in the IP header, not the actual frame length.
TrigID	92:87	RW	0x0	VLAN trigger match condition identifier.
Reserved	127:93	RSV	0x0	Reserved.

11.11.2.4 MA_USED_TABLE[0..1][0..511]

When a MAC lookup is performed, if the entry matches, then the corresponding bit is set in this table. Software can then clear each bit individually by writing a 1b to that entry.

Address:	0xC80000 + 0x200*j + 0x1*i + 0x28000
Atomicity:	32
Reset Domains:	SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Used	31:0	CW1	0x0	Bit i is set when source hit index modulo 32 equals i.

11.11.2.5 INGRESS_MST_TABLE[0..1][0..255]

INGRESS_MST_TABLE[0] is for ports 0-23 and INGRESS_MST_TABLE[1] is for ports 24-47.

Address:	0xC80000 + 0x200*j + 0x2*i + 0x28400
Atomicity:	64
Reset Domains:	SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
STPState[0..23]	47:0	RW	00b	(24 x 2-bits) Two bits of spanning tree state for each port. The states are: 00b = DISABLE 01b = LISTENING 10b = LEARNING 11b = FORWARD
Reserved	63:48	RSV	0x0	Reserved.

11.11.2.6 EGRESS_MST_TABLE[0..255]

Address: 0xC80000 + 0x2*i + 0x28800
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Forwarding	47:0	RW	0x0	Defines forwarding state per port for all ports.
Reserved	63:48	RSV	0x0	Reserved.

11.11.2.7 MA_TABLE_CFG_1

Address: 0xC80000 + 0x28A00
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
HashMode	0	RW	0b	0b = Each set in the MAC,FID hash uses a different hash function. 1b = (Intended for debug) All sets use the same hash function. Configures the MA_TABLE in set associative hash mode.
Reserved	31:1	RSV	0x0	Reserved.

11.11.2.8 MA_TABLE_CFG_2

Address: 0xC80000 + 0x28A02
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
FloodUnicastGlort	15:0	RW	0x0	Defines the GLORT to use when flooding unicast frames if the FFU does not already provide a flood GLORT. For DLF flood control, set to a GLORT with a zero destination mask.
FloodMulticastGlort	31:16	RW	0x0	Defines the GLORT to use when flooding multicast frames if the FFU does not already provide a flood GLORT. For DLF flood control, set to a GLORT with a zero destination mask.
BroadcastGlort	47:32	RW	0x0	Defines the GLORT used for broadcast.
Reserved	63:48	RSV	0x0	Reserved.

11.11.2.9 MTU_TABLE[0..7]

Address: 0xC80000 + 0x1*i + 0x28A08
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
mtu	13:0	RW	0x0	MTU size in bytes.
Reserved	31:14	RSV	0x0	Reserved.

11.11.2.10 IEEE_RESERVED_MAC_ACTION

Determines the action that is taken upon receipt of one of the reserved MAC addresses in the range 01-80-C2-00-00-00 through 01-80-C2-00-00-3F.

Address: 0xC80000 + 0x28A10
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
action[0..63]	127:0	RW	10b	(64 x 2-bits) Two bits per destination MAC address. Indicates the action to take upon receipt of that address: 00b = SwitchNormally 01b = Trap 10b = Drop 11b = Log

11.11.2.11 IEEE_RESERVED_MAC_TRAP_PRIORITY

Address: 0xC80000 + 0x28A14
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Select	63:0	RW	0x0	One bit per MAC address indicating the switch priority for the frame sent to the CPU when trapped: 0b = Switch priority remains unchanged. 1b = Switch priority changed to the value specified in IEEE_RESERVED_MAC_CFG.

11.11.2.12 IEEE_RESERVED_MAC_CFG

Address: 0xC80000 + 0x28A16
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
trapPri	3:0	RW	0x0	The switch priority of a frame with a reserved multicast MAC destination address that is trapped to the CPU when selected with IEEE_RESERVED_MAC_TRAP_PRIORITY.
Reserved	31:4	RSV	0x0	Reserved.

11.12 L2LOOKUP_TCN Registers Description

11.12.1 L2LOOKUP_TCN Map

Table 11-21 Layer 2 Lookup Registers (reserved 64 KW) Map

Name	Address (Base = 0xE70000)	Atomicity	Brief Description
MA_TCN_FIFO[0..511]	Base + 0x4*i + 0x0	64	MAC Table Change Notification FIFO.
MA_TCN_DEQUEUE	Base + 0x800	64	TCN FIFO Dequeue.
MA_TCN_PTR_HEAD	Base + 0x804	32	MAC Table Change software Read Pointer.
MA_TCN_PTR_TAIL	Base + 0x805	32	MAC Table Change HW Write Pointer.
MA_TCN_WM[0..47]	Base + 0x1*i + 0x840	32	Maximum number of TCN entries per port.
MA_TCN_USAGE[0..47]	Base + 0x1*i + 0x880	32	Number of TCN entries per port.
MA_TCN_IP	Base + 0x8C0	32	MAC Table Change Notification Interrupt Pending.
MA_TCN_IM	Base + 0x8C1	32	MAC Table Change Notification Interrupt Pending.

11.12.2 L2LOOKUP_TCN Registers

11.12.2.1 MA_TCN_FIFO[0..511]

Address: 0xE70000 + 0x4*i + 0x0
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
MACAddress	47:0	vRO	0x0	MAC Address.
VID	59:48	vRO	0x0	Ingress VLAN ID.
srcGlort	75:60	vRO	0x0	The frame's source GLORT that cause the notification.
Index	89:76	vRO	0x0	Entry index into the MA_TABLE.
Port	95:90	vRO	0x0	Source port that caused this event.
EntryType	96	vRO	0b	Type of entry. 0b = NewSource — Source MAC, FID was not found in the MA_TABLE.This event requires learning for this frame to not be disabled (either per-port or via a trigger). 1b = MacMoved — A source lookup was performed and a non-secure MAC address was found. The canonicalized source GLORT of the frame did not match the table entry's source GLORT, and needs to be updated in software. This event requires learning for this frame to not be disabled (either per-port, via TCN watermark or via a trigger).
U_err	97	vRO	0b	If set means there is an error (soft) in the read entry from FIFO, and SW needs to empty the FIFO.
Reserved	127:98	RSV	0x0	Reserved.

11.12.2.2 MA_TCN_DEQUEUE

Same format as the MA_TCN_FIFO register with the addition of one *Valid* bit. Reads to this address result in a FIFO dequeue event. The MA_TCN_PTR_HEAD pointer is automatically advanced by one when the upper 64 bits of this register is read. If the *Valid* bit is zero, the FIFO was empty and the fields in this register is undefined.

Address: 0xE70000 + 0x800
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
MACAddress	47:0	vRO	0x0	MAC Address.
VID	59:48	vRO	0x0	
srcGlort	75:60	vRO	0x0	Destination GLORT.
Index	89:76	vRO	0x0	Index into the MA_TABLE.
Port	95:90	vRO	0x0	
EntryType	96	vRO	0b	
Valid	97	vRO	0b	0b = The other fields in this register are undefined. 1b = There was an entry in the FIFO to dequeue.
U_err	98	vRO	0b	
Reserved	127:99	RSV	0x0	Reserved.

11.12.2.3 MA_TCN_PTR_HEAD

Address: 0xE70000 + 0x804
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Head	8:0	vRW	0x0	Entry the software reads next. Reading MA_TCN_DEQUEUE TCN FIFO increments this by one.
Reserved	31:9	RSV	0x0	Reserved.

11.12.2.4 MA_TCN_PTR_TAIL

Address: 0xE70000 + 0x805
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Tail	8:0	vRW	0x0	Entry the hardware writes next. Updated by the hardware.
Reserved	31:9	RSV	0x0	Reserved.

11.12.2.5 MA_TCN_WM[0..47]

Address: 0xE70000 + 0x1*i + 0x840
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Wm	8:0	RW	0xA	Maximum number of TCN entries. A value of 10 ensures that each port has space available in a 48-port configuration.
Reserved	31:9	RSV	0x0	Reserved.

11.12.2.6 MA_TCN_USAGE[0..47]

Address: 0xE70000 + 0x1*i + 0x880
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Usage	8:0	vRW	0x0	Number of TCN entries
Reserved	31:9	RSV	0x0	Reserved.

11.12.2.7 MA_TCN_IP

Writing 1b into any bit clears the corresponding bit, writing 0b has no effect.

Address: 0xE70000 + 0x8C0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
PendingEvents	0	vCW1	0b	Set whenever MA_TCN_PTR_TAIL is advanced and (512 + Tail(new) — Head) % 512 > 0 (FIFO not empty).
TCN_Overflow	1	vCW1	0b	Set whenever a new notification could not be enqueued to the MA_TCN_FIFO.
Reserved	31:2	RSV	0x0	Reserved.

11.12.2.8 MA_TCN_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0xE70000 + 0x8C1
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
PendingEvents	0	RW	0b	Mask PendingEvents interrupts: 0b = Do not mask 1b = Mask
TCN_Overflow	1	RW	0b	Mask TCN overflow interrupts: 0b = Do not mask 1b = Mask
Reserved	31:2	RSV	0x0	Reserved.

11.13 HANDLER Registers Description

11.13.1 HANDLER Map

Table 11-22 Frame Handler Generic Registers (reserved 4 KW) Map

Name	Address (Base = 0xD50000)	Atomicity	Brief Description
SYS_CFG_1	Base + 0x0	32	Layer 2 configuration
CPU_MAC	Base + 0x2	64	CPU MAC Address.
SYS_CFG_ROUTER	Base + 0x4	32	Global system configuration for ROUTER.
L34_HASH_CFG	Base + 0x5	32	Configures hashing of L3/L4 headers for ECMP and LAG.
L34_FLOW_HASH_CFG_1	Base + 0x6	32	Configures masking of flow fields in L34 hash.
L34_FLOW_HASH_CFG_2	Base + 0x7	32	Configures masking of flow fields in L34 hash.
L234_HASH_CFG	Base + 0x8	32	Configures hashing of L2/L3/L4 headers for LAG.
CPU_TRAP_MASK_FH	Base + 0xA	64	Defines the CPU mask for trapping.
TRAP_GLORT	Base + 0xC	32	Defines the trap GLORT when a frame must be trapped.
RX_MIRROR_CFG	Base + 0xD	32	Selects the mirror 1 profile index for FFU RX Mirroring.
LOG_MIRROR_PROFILE	Base + 0xE	32	Selects the mirror 0 profile index for various logging conditions.
FH_MIRROR_PROFILE_TABLE[0..63]	Base + 0x1*i + 0x40	32	Mirror profile entries for frame handler.
PORT_CFG_2[0..47]	Base + 0x2*i + 0x80	64	Destination bit-mask per source port.
PORT_CFG_3[0..47]	Base + 0x1*i + 0x100	32	Security and learning options by source port.
FH_LOOPBACK_SUPPRESS[0..47]	Base + 0x1*i + 0x140	32	GloRTs used by loopback suppression algorithm.
FH_HEAD_IP	Base + 0x180	64	
FH_HEAD_IM	Base + 0x182	64	
PARSER_EARLY_SRAM_CTRL	Base + 0x184	32	
PARSER_LATE_SRAM_CTRL	Base + 0x185	32	
MAPPER_SRAM_CTRL	Base + 0x186	32	
FFU_SRAM_CTRL[0..7]	Base + 0x4*i + 0x1A0	32	
ARP_SRAM_CTRL	Base + 0x1C0	32	
VLAN_LOOKUP_SRAM_CTRL	Base + 0x1C1	32	
MA_TABLE_SRAM_CTRL[0..1]	Base + 0x2*i + 0x1C4	64	
FID_GLORT_LOOKUP_SRAM_CTRL	Base + 0x1C8	64	

Table 11-22 Frame Handler Generic Registers (reserved 4 KW) Map [Continued]

Name	Address (Base = 0xD50000)	Atomicity	Brief Description
GLORT_RAM_SRAM_CTRL	Base + 0x1CA	32	
GLORT_TABLE_SRAM_CTRL	Base + 0x1CB	32	
FH_HEAD_OUTPUT_FIFO_SRAM_CTRL	Base + 0x1CC	32	

11.13.2 HANDLER Registers

11.13.2.1 SYS_CFG_1

Address: **Atomicity:** **Reset Domains:**	**0xD50000 + 0x0** **32** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
dropPause	0	RW	1b	Drop pause (ignoring length, regardless of whether PAUSE is enabled or disabled. This has no effect on PAUSE processing.
trapMTUViolations	1	RW	1b	Defines disposition for frames that exceed MTU size allowed on the egress VLAN (only applicable for routed frames). 0b = Frames are forwarded normally. 1b = Frames that exceed the MTU size are trapped to the CPU.
enableTrapPlusLog	2	RW	1b	0b = Trapping takes precedence over logging. 1b = Trapping and logging can both occur for a given frame.
dropInvalidSMAC	3	RW	1b	Drop any frame with an invalid SMAC of all zeros (00:00:00:00:00:00), all ones (FF:FF:FF:FF:FF:FF) or multicast (bit 40 is a 1b), and do not generate a MA_TCN_FIFO event.
dropMacCtrlEthertype	4	RW	1b	Drop MAC CTRL Ethertype 0x8808. This has no effect on PAUSE processing.
Reserved	31:5	RSV	0x0	Reserved.

11.13.2.2 CPU_MAC

Address: **Atomicity:** **Reset Domains:**	**0xD50000 + 0x2** **64** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MacAddr	47:0	RW	0x0	CPU MAC address.
Reserved	63:48	RSV	0x0	Reserved.

11.13.2.3 SYS_CFG_ROUTER

Address: 0xD50000 + 0x4
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
TTLdisposal	1:0	RW	00b	Defines disposition of routed frames with TTL 1 or 0. The options are: 00b = Drop silently. 01b = Trap (unicast) or log (multicast) if protocol is ICMP, and drop otherwise. 10b = Trap (unicast) or log (multicast) all frames. 11b = Reserved. ***Note:*** This field only applies to frames that are routed. If a unicast or multicast frame is not routed, this frame is switched normally regardless of its TTL value. For multicast frames that are both switched and routed, the frame is still switched within the VLAN it was received from regardless of its TTL value while the routing part is conditional to the TTL value received and the configuration of this field.
trapIPOptions	2	RW	0b	Defines disposition of IP frames with options: 0b = Forward normally. 1b = Trap to CPU.
Reserved	31:3	RSV	0x0	Reserved.

11.13.2.4 L34_HASH_CFG

Address: 0xD50000 + 0x5
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Symmetric	0	RW	0b	When set, SIP/DIP and L4SRC/L4DST fields are made symmetric prior to hashing.
UseSIP	1	RW	1b	Use SIP in L3/L4 hash.
UseDIP	2	RW	1b	Use DIP in L3/L4 hash.
UsePROT	3	RW	1b	Use PROT in L3/L4 hash.
UseTCP	4	RW	1b	Use L4SRC/L4DST when PROT is TCP.
UseUDP	5	RW	1b	Use L4SRC/L4DST when PROT is UDP.
UsePROT1	6	RW	1b	Use L4SRC/L4DST when PROT is PROT1.
UsePROT2	7	RW	1b	Use L4SRC/L4DST when PROT is PROT2.
UseL4SRC	8	RW	1b	Use L4SRC if L4 protocol is enabled.
UseL4DST	9	RW	1b	Use L4DST if L4 protocol is enabled.
ECMP_Rotation	11:10	RW	00b	Specifies one of three 12-bit hash rotations for use in ECMP binning. The value 0x3 is reserved. Not applicable to FM3000 devices.
RSVD	15:12	RW	0x0	Reserved.
PROT1	23:16	RW	0x1	PROT1 programmable L4 protocol.
PROT2	31:24	RW	0x1	PROT2 programmable L4 protocol.

11.13.2.5 L34_FLOW_HASH_CFG_1

Address: 0xD50000 + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DiffServMask	5:0	RW	0x0	Masks the IPv4 DiffServ field in the L34 hash function.
Reserved	7:6	RSV	00b	Reserved.
UserMask	15:8	RW	0x0	Masks the ISL tag's USER field in the L34 hash function.
Reserved	31:16	RSV	0x0	Reserved.

11.13.2.6 L34_FLOW_HASH_CFG_2

Address: 0xD50000 + 0x7
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
FlowLabelMask	19:0	RW	0x0	Masks the IPv6 Flow Label field in the L34 hash function.
Reserved	31:20	RSV	0x0	Reserved.

11.13.2.7 L234_HASH_CFG

Address: 0xD50000 + 0x8
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
UseL2ifIP	0	RW	1b	Include the Layer 2 header in the hash function if an IPv4/IPv6 packet.
UseL34	1	RW	1b	Include the L34 hash value in the L234 hash function.
Symmetric	2	RW	1b	When set, SMAC/DMAC fields are made symmetric prior to hashing.
UseDMAC	3	RW	1b	Include DMAC in the L2/L3/L4 hash.
UseSMAC	4	RW	1b	Include SMAC in the L2/L3/L4 hash.
UseType	5	RW	1b	Include EtherType field in the L2/L3/L4 hash. If EtherType is less than 0x600, a value of 0b is used for these two bytes.
UseVPRI	6	RW	1b	Include User (VLAN) Priority field in the L2/L3/L4 hash.
UseVID	7	RW	1b	Include VLAN ID field in the L2/L3/L4 hash.
RotationA	9:8	RW	01b	Specifies one of four 12-bit H234 rotations for use as Rotation A.
RotationB	11:10	RW	01b	Specifies one of four 12-bit H234 rotations for use as Rotation B.
Reserved	31:12	RSV	0x0	Reserved.

11.13.2.8 CPU_TRAP_MASK_FH

When a trapped frame is sent to the CPU, it is sent out on these ports. (This is a mask rather than a single port number, so that it is possible to multicast to multiple CPUs if desired.)

Address: 0xD50000 + 0xA
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DestMask	47:0	RW	0x1	Destination mask for CPU.
Reserved	63:48	RSV	0x0	Reserved.

11.13.2.9 TRAP_GLORT

Address: 0xD50000 + 0xC
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
trapGlort	15:0	RW	0x0	Destination GLORT for frames trapped to the CPU. Only the high-order 8 bits are used. The low-order 8 bits appearing in the DGLORT of the trapped frame is the Trap and Log Code, which indicates the reason the frame was trapped (refer to Section 5.7.1.5 and Section 5.7.1.6 for details).
Reserved	31:16	RSV	0x0	Reserved.

11.13.2.10 RX_MIRROR_CFG

Address: 0xD50000 + 0xD
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
MirrorProfileIdx	5:0	RW	0x0	Set the mirror 1 profile index to this value if the FFU RX Mirror action flag is set.
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.11 LOG_MIRROR_PROFILE

Address: 0xD50000 + 0xE
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
FFU	5:0	RW	0x0	Mirror 0 profile index due to an FFU log action.
ReservedMAC	11:6	RW	0x0	Mirror 0 profile index when a frame with an IEEE Reserved destination MAC address is logged, per IEEE_RESERVED_MAC_ACTION.
ARPRedirect	17:12	RW	0x0	Mirror 0 profile index on an ARP log (unicast frame routed back to ingress VLAN).
ICMP	23:18	RW	0x0	Mirror 0 profile index when a multicast ICMP frame is logged because TTL ≤ 1.
TTL	29:24	RW	0x0	Mirror 0 profile index when any multicast frame is logged because TTL ≤ 1.
Reserved	31:30	RSV	00b	Reserved.

11.13.2.12 FH_MIRROR_PROFILE_TABLE[0..63]

Address: 0xD50000 + 0x1*i + 0x40
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
port	5:0	RW	0x0	Destination port for the mirror profile entry.
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.13 PORT_CFG_2[0..47]

A destination mask ANDed with the GLORT destination mask result.

Address: 0xD50000 + 0x2*i + 0x80
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
destinationMask	47:0	RW	0xFFFFFFFFFFFF	A vector for each port i, a bit for each port j. Always applied on every frame.
Reserved	63:48	RSV	0x0	Reserved.

11.13.2.14 PORT_CFG_3[0..47]

Address: 0xD50000 + 0x1*i + 0x100
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
filterVLANIngress	0	RW	1b	If set to 1b, and the source port does not match the VLAN membership, it is a VLAN boundary violation and the packet is dropped
LearningEnable	1	RW	1b	If set to 1b, source MAC addresses from this port is put into the TCN FIFO.
Reserved	31:2	RSV	0x0	Reserved.

11.13.2.15 FH_LOOPBACK_SUPPRESS[0..47]

At the edge of the system, avoid retransmitting a frame over the same LAG it arrived on, even if the LAG is distributed across multiple edge chips. This is done by comparing the GloRTs for the source and destination LAG/port. If the GloRTs match, not counting the port-specific bits, the LAGs are the same.

Note: The destination port's GLORT is not necessarily the same as the destination GLORT (which might be multicast), so this comparison needs to be done for all external destination ports in parallel.The comparison is turned off for internal ports.

Address: 0xD50000 + 0x1*i + 0x140
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
glort	15:0	RW	0xFFFF	GLORT value.
mask	31:16	RW	0x0	GLORT mask.

11.13.2.16 FH_HEAD_IP

SramErr fields have a 1:1 mapping with _SRAM_CTRL registers. The 2-bit value is: Uncorrected Error (MSB), Corrected Error (LSB).

Address: 0xD50000 + 0x180
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ParserEarlySramErr	1:0	vCW1	00b	
ParserLateSramErr	3:2	vCW1	00b	
MapperSramErr	5:4	vCW1	00b	
FFUSramErr[0..7]	21:6	vCW1	00b	(8 x 2-bits)
ArpSramErr	23:22	vCW1	00b	
VlanSramErr	25:24	vCW1	00b	
MaTableSramErr[0..1]	29:26	vCW1	00b	(2 x 2-bits)
FGLSramErr	31:30	vCW1	00b	
GlortRamSramErr	33:32	vCW1	00b	
GlortTableSramErr	35:34	vCW1	00b	
FhHeadOutputFifoSramErr	37:36	vCW1	00b	
Trigger	38	vRO	0b	Trigger interrupt pending. See TRIGGER_IP for the source of the interrupt.
Reserved	63:39	RSV	0x0	Reserved.

11.13.2.17 FH_HEAD_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0xD50000 + 0x182
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ParserEarlySramErr	1:0	RW	11b	
ParserLateSramErr	3:2	RW	11b	
MapperSramErr	5:4	RW	11b	
FFUSramErr[0..7]	21:6	RW	11b	(8 x 2-bits)
ArpSramErr	23:22	RW	11b	
VlanSramErr	25:24	RW	11b	
MaTableSramErr[0..1]	29:26	RW	11b	(2 x 2-bits)
FGLSramErr	31:30	RW	11b	
GlortRamSramErr	33:32	RW	11b	
GlortTableSramErr	35:34	RW	11b	
FhHeadOutputFifoSramErr	37:36	RW	11b	
Trigger	38	RW	1b	
Reserved	63:39	RSV	0x0	Reserved.

11.13.2.18 PARSER_EARLY_SRAM_CTRL

PARSER_PORT_CFG_2. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x184**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.19 PARSER_LATE_SRAM_CTRL

PARSER_PORT_CFG_3 = 0, RX_VPRI_MAP = 1. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x185**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..1]	3:0	RW	00b	(2 x 2-bits)
CErr[0..1]	5:4	vCW1	0b	(2 x 1-bit)
UErr[0..1]	7:6	vCW1	0b	(2 x 1-bit)
BistDonePass[0..1]	9:8	vRO	0b	(2 x 1-bit)
BistDoneFail[0..1]	11:10	vRO	0b	(2 x 1-bit)
Reserved	31:12	RSV	0x0	Reserved.

11.13.2.20 MAPPER_SRAM_CTRL

FFU_MAP_VLAN. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x186**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.21 FFU_SRAM_CTRL[0..7]

Each entry covers four slices. Each slice has four consecutive field indices: SRAM=0-1, TCAM=2-3. For the TCAMs, only *BistDonePass* and *BistDoneFail* are used, and are active only when *BistMode* is set to regular BIST, not pattern fill. ErrWrite, CErr, and UErr are reserved for the TCAMs. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x4*i + 0x1A0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..15]	31:0	RW	00b	(16 x 2-bits)
CErr[0..15]	47:32	vCW1	0b	(16 x 1-bit)
UErr[0..15]	63:48	vCW1	0b	(16 x 1-bit)
BistDonePass[0..15]	79:64	vRO	0b	(16 x 1-bit)
BistDoneFail[0..15]	95:80	vRO	0b	(16 x 1-bit)
Reserved	127:96	RSV	0x0	Reserved.

11.13.2.22 ARP_SRAM_CTRL

ARP Table = 0-3, ARP Used = 4. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x1C0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..4]	9:0	RW	00b	(5 x 2-bits) Entry 4 (ARP_USED) is bitwise parity protected. Writing 1 or 2 produces *UErr*, and writing 3 corrupts data without producing *UErr*. Entries 0-3 are standard.
CErr[0..4]	14:10	vCW1	0b	(5 x 1-bit) Bit 4 is reserved and is always read as 0. ARP_USED is parity protected and does not have correctable errors.
UErr[0..4]	19:15	vCW1	0b	(5 x 1-bit)
BistDonePass[0..4]	24:20	vRO	0b	(5 x 1-bit)
BistDoneFail[0..4]	29:25	vRO	0b	(5 x 1-bit)
Reserved	31:30	RSV	00b	Reserved.

11.13.2.23 VLAN_LOOKUP_SRAM_CTRL

Ingress = 0-1, Egress = 2-3. Part of Fabric BIST domain.

Address:	**0xD50000 + 0x1C1**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..3]	7:0	RW	00b	(4 x 2-bits)
CErr[0..3]	11:8	vCW1	0b	(4 x 1-bit)
UErr[0..3]	15:12	vCW1	0b	(4 x 1-bit)

Address: **0xD50000 + 0x1C1**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
BistDonePass[0..3]	19:16	vRO	0b	(4 x 1-bit)
BistDoneFail[0..3]	23:20	vRO	0b	(4 x 1-bit)
Reserved	31:24	RSV	0x0	Reserved.

11.13.2.24 MA_TABLE_SRAM_CTRL[0..1]

In each, Key = 0-3, Payload = 4-7. Part of Fabric BIST domain.

Address: **0xD50000 + 0x2*i + 0x1C4**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..7]	15:0	RW	00b	(8 x 2-bits)
CErr[0..7]	23:16	vCW1	0b	(8 x 1-bit)
UErr[0..7]	31:24	vCW1	0b	(8 x 1-bit)
BistDonePass[0..7]	39:32	vRO	0b	(8 x 1-bit)
BistDoneFail[0..7]	47:40	vRO	0b	(8 x 1-bit)
Reserved	63:48	RSV	0x0	Reserved.

11.13.2.25 FID_GLORT_LOOKUP_SRAM_CTRL

Ingress = 0-1, Egress = 2, MA_USED = 3-4, GLORT_CAM = 5

For MA_USED and the TCAM, only *BistDonePass* and *BistDoneFail* are used. *ErrWrite*, *CErr*, and *UErr* are reserved for MA_USED and the TCAM. For the TCAM, *BistDonePass* and *BistDoneFail* will only be active when BistMode is set to regular BIST, not pattern fill. Part of Fabric BIST domain.

Address: **0xD50000 + 0x1C8**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..5]	11:0	RW	00b	(6 x 2-bits)
CErr[0..5]	17:12	vCW1	0b	(6 x 1-bit)
UErr[0..5]	23:18	vCW1	0b	(6 x 1-bit)
BistDonePass[0..5]	29:24	vRO	0b	(6 x 1-bit)
BistDoneFail[0..5]	35:30	vRO	0b	(6 x 1-bit)
Reserved	63:36	RSV	0x0	Reserved.

11.13.2.26 GLORT_RAM_SRAM_CTRL

Part of Fabric BIST domain.

Address: 0xD50000 + 0x1CA
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.27 GLORT_TABLE_SRAM_CTRL

Part of Fabric BIST domain.

Address: 0xD50000 + 0x1CB
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.13.2.28 FH_HEAD_OUTPUT_FIFO_SRAM_CTRL

Part of Fabric BIST domain.

Address: 0xD50000 + 0x1CC
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..4]	9:0	RW	00b	(5 x 2-bits)
CErr[0..4]	14:10	vCW1	0b	(5 x 1-bit)
UErr[0..4]	19:15	vCW1	0b	(5 x 1-bit)
BistDonePass[0..4]	24:20	vRO	0b	(5 x 1-bit)
BistDoneFail[0..4]	29:25	vRO	0b	(5 x 1-bit)
Reserved	31:30	RSV	00b	Reserved.

11.14 HANDLER_TAIL Registers Description

11.14.1 HANDLER_TAIL Map

Table 11-23 Frame Handler Generic Registers (reserved 4 KW) Map

Name	Address (Base = 0xE30000)	Atomicity	Brief Description
SAF_MATRIX[0..47]	Base + 0x2*i + 0x0	64	Store-and-forward matrix.
FRAME_TIME_OUT	Base + 0x80	32	Frame Timeout.
SAF_SRAM_CTRL	Base + 0x81	32	
EGRESS_PAUSE_SRAM_CTRL	Base + 0x82	32	
RX_STATS_SRAM_CTRL	Base + 0x84	64	
POLICER_USAGE_SRAM_CTRL	Base + 0x88	64	
TCN_SRAM_CTRL	Base + 0x8C	32	
FH_TAIL_IP	Base + 0x8D	32	
FH_TAIL_IM	Base + 0x8E	32	
TAIL_PERMIT_MGMT	Base + 0x8F	32	Permit Frame Control Tail management if not enough idle cycles.
TAIL_FORCE_IDLE	Base + 0x90	32	Force Frame Control Tail idle cycle to allow sweep.

11.14.2 HANDLER_TAIL Registers

11.14.2.1 SAF_MATRIX[0..47]

Defines the mode of operation, store-and-forward or cut-through, for each ingress to egress pair of ports. A frame received from port src_port is stored-and-forwarded if SAF_MATRIX[src_port].*EnableSNF* & DMASK != 0.

CutThruMode 0 and *IgnoreError* should not be used unless CM is configured to prevent an out-of-memory condition.

Address: 0xE30000 + 0x2*i + 0x0
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
EnableSNF	47:0	RW	0x0	Enable store-and-forward when a frame from this source port is queued to each egress port.
CutThruMode	49:48	RW	00b	Any frame that is not store-and-forwarded as a result of its destination port mask's *EnableSNF* test is eligible for cut-through forwarding. The forwarding behavior of such frames is selected from the following four modes: 00b = Full cut-thru — The frame is eligible for egress scheduling immediately. 01b = One segment SnF — The frame is eligible for egress scheduling once its first segment has been entirely received. 10b = Two segment SnF — The frame is eligible for egress scheduling once its second segment has been entirely received. 11b = End-of-frame SnF — Frames arriving from this port is always fully stored to memory before being egress scheduled.
IgnoreError	50	RW	00b	Send the frame as if it were in full-cut-through mode, even if it has an error.
Reserved	63:51	RSV	0b	Reserved.

11.14.2.2 FRAME_TIME_OUT

Defines the period at which the timeout mechanism drops frames.

Address: 0xE30000 + 0x80
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
timeoutMult	23:0	RW	0x0	Defines the frame time out duration in units of 2048 times the Switch Clock. The timeout is precised at +/- 50%. Writing a new value takes effect immediately, and a write of any non-zero value triggers an immediate advance of the global time value. Therefore successive writes should not be made faster than the maximum time required to time out the entire memory (5 ms). *timeOutMult* > 0 A frame times out in 2048 * timeOutMult * Switch clock. *timeOutMult* = 0 Frame timeout disabled. The frame timeout mechanism may not be reliable if *timeOutMult* is set less than 10 ms.
cnt	25:24	vRO	00b	The current time counter value. This counter increments by one every frame timeout period. The Egress Scheduler is updated with the current cnt whenever it changes.
Reserved	31:26	RSV	0x0	Reserved.

11.14.2.3 SAF_SRAM_CTRL

Part of Fabric BIST domain.

Address: 0xE30000 + 0x81
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..3]	7:0	RW	00b	(4 x 2-bits)
CErr[0..3]	11:8	vCW1	0b	(4 x 1-bit)
UErr[0..3]	15:12	vCW1	0b	(4 x 1-bit)
BistDonePass[0..3]	19:16	vRO	0b	(4 x 1-bit)
BistDoneFail[0..3]	23:20	vRO	0b	(4 x 1-bit)
Reserved	31:24	RSV	0x0	Reserved.

11.14.2.4 EGRESS_PAUSE_SRAM_CTRL

Two ports of the same memory. BIST signals of both entries will always be the same. Part of Fabric BIST domain.

Address: 0xE30000 + 0x82
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..1]	3:0	RW	00b	(2 x 2-bits) Bitwise parity protected:. Writing 1 or 2 produces *UErr*, and writing 3 corrupts data without producing *UErr*.
CErr[0..1]	5:4	vCW1	0b	(2 x 1-bit) Due to parity protection, will not have correctable errors.
UErr[0..1]	7:6	vCW1	0b	(2 x 1-bit)
BistDonePass[0..1]	9:8	vRO	0b	(2 x 1-bit)
BistDoneFail[0..1]	11:10	vRO	0b	(2 x 1-bit)
Reserved	31:12	RSV	0x0	Reserved.

11.14.2.5 RX_STATS_SRAM_CTRL

Even=frame, odd =byte. Part of Fabric BIST domain.

Address: 0xE30000 + 0x84
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..11]	23:0	RW	00b	(12 x 2-bits)
CErr[0..11]	35:24	vCW1	0b	(12 x 1-bit)
UErr[0..11]	47:36	vCW1	0b	(12 x 1-bit)
BistDonePass[0..11]	59:48	vRO	0b	(12 x 1-bit)
BistDoneFail[0..11]	71:60	vRO	0b	(12 x 1-bit)
Reserved	127:72	RSV	0x0	Reserved.

11.14.2.6 POLICER_USAGE_SRAM_CTRL

0-3=cfg, 4-7= count1, 8-11=count2. Part of Fabric BIST domain.

Note: Corresponding POLICER_STATE_*.*Count1* and .*Count2* fields are written simultaneously in a RMW mechanism, so a write to one of them can cause errors to be injected in both (if ErrWrite is set for both SRAMs).

Address: 0xE30000 + 0x88
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..11]	23:0	RW	00b	(12 x 2-bits)
CErr[0..11]	35:24	vCW1	0b	(12 x 1-bit)
UErr[0..11]	47:36	vCW1	0b	(12 x 1-bit)
BistDonePass[0..11]	59:48	vRO	0b	(12 x 1-bit)
BistDoneFail[0..11]	71:60	vRO	0b	(12 x 1-bit)
Reserved	127:72	RSV	0x0	Reserved.

11.14.2.7 TCN_SRAM_CTRL

Part of Fabric BIST domain.

Address: 0xE30000 + 0x8C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.14.2.8 FH_TAIL_IP

SramErr fields have a 1:1 mapping with _SRAM_CTRL registers. The two bit value is: Uncorrected Error (MSB), Corrected Error (LSB).

Address: 0xE30000 + 0x8D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
SafSramErr	1:0	vCW1	00b	
EgressPauseSramErr	3:2	vCW1	00b	
RxStatsSramErr	5:4	vCW1	00b	
PolicerUsageSramErr	7:6	vCW1	00b	
TcnSramErr	9:8	vCW1	00b	

Address: **0xE30000 + 0x8D**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
TCN	10	vRO	0b	MAC Table TCN FIFO interrupt pending. See MA_TCN_IP for the source of the interrupt.
Reserved	31:11	RSV	0x0	Reserved.

11.14.2.9 FH_TAIL_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: **0xE30000 + 0x8E**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SafSramErr	1:0	RW	11b	
EgressPauseSramErr	3:2	RW	11b	
RxStatsSramErr	5:4	RW	11b	
PolicerUsageSramErr	7:6	RW	11b	
TcnSramErr	9:8	RW	11b	
TCN	10	RW	1b	
Reserved	31:11	RSV	0x0	Reserved.

11.14.2.10 TAIL_PERMIT_MGMT

Address: **0xE30000 + 0x8F**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Timer	15:0	vRO	0x0	Resets to 0 on idle cycle and increments every non-idle cycle until reaching *TimerMax*. When *TimerMax* is reached, a management cycle is permitted regardless of whether the cycle is idle and *Timer* resets to 0.
TimerMax	31:16	RW	0xFFFF	Max count value of *Timer*. Set to 0b to disable.

11.14.2.11 TAIL_FORCE_IDLE

Address:	**0xE30000 + 0x90**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Timer	15:0	vRO	0x0	Increments every cycle until reaching *TimerMax*. When *TimerMax* is reached, an idle cycle with no management or frame is forced to allow a sweep.
TimerMax	31:16	RW	0x0	Max count value of *Timer*. Set to 0b to disable.

11.15 GLORT Registers Description

11.15.1 GLORT Map

Table 11-24 GLORT Control Registers and GLORT Table (reserved 16 KW) Map

Name	Address (Base = 0xCE0000)	Atomicity	Brief Description
GLORT_DEST_TABLE[0..4095]	Base + 0x2*i + 0x0	64	
GLORT_CAM[0..255]	Base + 0x1*i + 0x2000	32	
GLORT_RAM[0..255]	Base + 0x2*i + 0x2200	64	

11.15.2 GLORT Registers

11.15.2.1 GLORT_DEST_TABLE[0..4095]

Defines the destination ports for a given glort. The *DestMask* is a bit vector indicating which ports receive normal (not logged or mirrored) copies of the frame. If an *IP_MulticastIndex* is specified (with value different than 0), SCHED_MCAST_DEST_TABLE is used to retrieve lists of VLANs for each port to use during IP multicast replication.

Address: **0xCE0000 + 0x2*i + 0x0**
Atomicity: **64**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
DestMask	47:0	RW	0x0	Contains the destination port mask (for normal copies, not including logged or mirrored copies).
IP_MulticastIndex	59:48	RW	0x0	Selects the IP multicast profile. This value is used as an index into SCHED_MCAST_DEST_TABLE[0..4095].
Reserved	63:60	RSV	0x0	Reserved.

11.15.2.2 GLORT_CAM[0..255]

Configures an entry in the GLORT CAM. The CAM entries are readable, but the value returned is masked out according to the mask originally written along with the value.

A match occurs when the data searched is equal to the key and the entry is valid. The table below shows the encoded data value stored (Content) based on the state of the data (*Key*) and mask (*KeyInvert*) values in the TCAM bitcell.

Mask Storage	Data Storage	Content
0	0	x (always match)
0	1	1
1	0	0
1	1	Always mismatch state

The highest index has precedence in case there is more than one match.

Address: 0xCE0000 + 0x1*i + 0x2000
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Key	15:0	RW	0x0	GLORT value.
KeyInvert	31:16	RW	0x0	GLORT Mask. Write the inverse to match exactly.

11.15.2.3 GLORT_RAM[0..255]

Data associated with each CAM entry. The entry defines how to compute the index for indexing the GLORT_DEST_TABLE to retrieve the final destination.

The *Strict* field indicates whether the GLORT_DEST_TABLE index is generated deterministically (strict) or by hashing. When strict is used (Strict is 0x3 or it is 0x0 and the ISL tag's FTYPE is either special delivery or management), the index into the GLORT_DEST_TABLE is computed as follows:

index = *DestIndex* + (glort{B}<<width(A)) + glort{A}

where:

glort{A}: is the value of the bits extracted from the GLORT according to *RangeSubIndexA*.

glort{B}: is the value of the bits extracted from the GLORT according to *RangeSubIndexB*.

width{A}: is the number of bits extracted from the GLORT according to *RangeSubIndexA* (indicated by bits 21:18 in *RangeSubIndexA*).

The idea is to provide a single CAM entry to cover multiple LAGs with multiple ports in each LAG where the port numbers are encoded in *RangeSubIndexB* and the LAG numbers are encoded in *RangeSubIndexA*. The number of ports in each LAG need not be a power of 2 to make efficient use of the GLORT_DEST_TABLE. If the number of LAGs is not a power of 2, a choice can be made between wasting GLORT_DEST_TABLE entries or consuming additional GLORT_CAM/RAM entries by dividing the LAGs into multiple sets, each set containing a number of LAGs that is a power of 2.

When non—strict is used (Strict is 0x2 or it is 0x0 and the ISL tag's FTYPE is either routed or normal), the index into the GLORT_DEST_TABLE is computed as follows:

index = *DestIndex* + ((hash%DestCount)<<width(A)) + glort{A}

where:

glort{A}: is the value of the bits extracted from the GLORT according to *RangeSubIndexA*.

width{A}: is the number of bits extracted from the GLORT according to *RangeSubIndexA* (indicated by bits 21:18 in *RangeSubIndexA*).

hash%DestCount: is a modulo hash over the *DestCount* number of entries in the GLORT_DEST_TABLE per LAG.

If *DestCount* is equal to the highest value that would ever be seen encoded by *RangeSubIndexB*, the difference between strict and non—strict is essentially the difference between hashing over the ports in a LAG and addressing the individual ports specifically (as required by LACP).

Address:	**0xCE0000 + 0x2*i + 0x2200**
Atomicity:	**64**
Reset Domains:	**SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
Strict	1:0	RW	00b	Determines strict versus hashed destination mask selection. 00b = FTYPE — Strict property is inherited from the ISL tag's FTYPE. 01b = RSVD — Reserved. 10b = HASHED — Not strict (hashed). 11b = STRICT — Strict.
DestIndex	13:2	RW	0x0	Defines the base index of the first entry into the GLORT_DEST table which contains the destination.
RangeSubIndexA	21:14	RW	0x0	Defines the position of sub index A (first 4 bits is bit position, second 4 bits is number of bits) to use for indexing into the GLORT_DEST_TABLE.
RangeSubIndexB	29:22	RW	0x0	Defines the position of sub index B (first 4 bits is bit position, second 4 bits is number of bits) to use for indexing into the GLORT_DEST_TABLE.
DestCount	33:30	RW	0x0	Defines the number of DestMask entries in the GLORT_DEST_TABLE over which frames are hashed. A value of 0 is interpreted as 16. For distributed link aggregation pruning, this corresponds to the number of ports in the LAG.
HashRotation	34	RW	0b	Selects hash rotation for calculating the destination mask index. 0b = Rotation A 1b = Rotation B
Reserved	63:35	RSV	0x0	Reserved.

11.16 LAG Registers Description

11.16.1 LAG Map

Table 11-25 Link Aggregation Registers Map

Name	Address (Base = 0xD90000)	Atomicity	Brief Description
LAG_CFG[0..47]	Base + 0x1*i + 0x0	32	LAG Configuration.
CANONICAL_GLORT_CAM[0..15]	Base + 0x1*i + 0x40	32	Canonical GLORT lookup CAM.

11.16.2 LAG Registers

11.16.2.1 LAG_CFG[0..47]

Address:	**0xD90000 + 0x1*i + 0x0**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LagSize	3:0	RW	0x0	Defines the number of ports in the link aggregation group. The value 0 means 16.
Index	7:4	RW	0x0	Index of this port in the link aggregation port, must be between 0 and LAG_CFG.*LagSize* - 1.
HashRotation	8	RW	0b	Selects hash rotation for filtering. 0b = Rotation A 1b = Rotation B.
InLAG	9	RW	0b	Defines if the port is currently part of a LAG group: 0b = Not part of a LAG group. 1b = Part of a LAG group.
Reserved	31:10	RSV	0x0	Reserved.

11.16.2.2 CANONICAL_GLORT_CAM[0..15]

Address:	**0xD90000 + 0x1*i + 0x40**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LagGlort	15:0	RW	0x0	Each source GLORT is masked and compared to this LAG GLORT. If it matches, the lower *PortFieldSize* bits are cleared.
MaskSize	19:16	RW	0x0	Comparison to *LagGlort* is performed by first ANDing the source GLORT by ~(2***MaskSize* - 1).
PortFieldSize	22:20	RW	000b	Number of low-order bits to clear in a source GLORT that matches this entry's *LagGlort*.
Reserved	31:23	RSV	0x0	Reserved.

11.17 TRIG_APPLY Registers Description

11.17.1 TRIG_APPLY Map

Table 11-26 Triggers Registers (reserved 4 KW) Map

Name	Address (Base = 0xD70000)	Atomicity	Brief Description
TRIGGER_CONDITION_CFG[0..63]	Base + 0x1*i + 0x0	32	General per-trigger match condition configuration.
TRIGGER_CONDITION_PARAM[0..63]	Base + 0x1*i + 0x40	32	Miscellaneous match condition parameters.
TRIGGER_CONDITION_FFU[0..63]	Base + 0x1*i + 0x80	32	FFU match parameters.
TRIGGER_CONDITION_TYPE[0..63]	Base + 0x1*i + 0xC0	32	EtherType match parameters.
TRIGGER_CONDITION_GLORT[0..63]	Base + 0x1*i + 0x100	32	Destination GLORT match parameters.
TRIGGER_CONDITION_RX[0..63]	Base + 0x2*i + 0x180	64	Source port mask.
TRIGGER_CONDITION_TX[0..63]	Base + 0x2*i + 0x200	64	Destination port mask.
TRIGGER_CONDITION_AMASK_1[0..63]	Base + 0x1*i + 0x280	32	Frame Handler action mask match condition (actions 0 through 31).
TRIGGER_CONDITION_AMASK_2[0..63]	Base + 0x1*i + 0x2C0	32	Frame Handler action mask match condition (actions 32 through 44).
TRIGGER_ACTION_CFG_1[0..63]	Base + 0x1*i + 0x300	32	General per-trigger action configuration.
TRIGGER_ACTION_CFG_2[0..63]	Base + 0x1*i + 0x340	32	General per-trigger action configuration.
TRIGGER_ACTION_GLORT[0..63]	Base + 0x1*i + 0x380	32	Forwarding action GLORT assignment.
TRIGGER_ACTION_DMASK[0..63]	Base + 0x2*i + 0x400	64	Forwarding action destination mask assignment.
TRIGGER_ACTION_MIRROR[0..63]	Base + 0x1*i + 0x480	32	Mirroring action port and GLORT assignment.
TRIGGER_ACTION_DROP[0..63]	Base + 0x2*i + 0x500	64	Drop action port mask.
TRIGGER_STATS[0..63]	Base + 0x2*i + 0x580	64	Statistics on trigger actions.
TRIGGER_IP	Base + 0x600	64	Trigger IP registers.
TRIGGER_IM	Base + 0x602	64	Trigger IM registers.
TRIGGER_RATE_LIM_EMPTY	Base + 0x604	32	Rate limiter token-bucket status.
TRIGGER_RATE_LIM_CFG_2[0..15]	Base + 0x2*i + 0x620	64	Rate limiter configuration (Drop mask).

11.17.2 TRIG_APPLY Enumerated Data Types

Table 11-27 TRIG APPLY Enumerated Data Types

Name	Width	Description
AMASK	6	0-37 are action codes, 38-42 are log codes, and 43-44 are additional codes that can be used by triggers. For each action code, an unknown source MAC address is reported to the MA_TCN_FIFO unless otherwise indicated by "No learning." Highest to lowest precedence: 000000b = SPECIAL — Specially-handled frame. No Learning. 000001b = DROP_PARSE_ERR — Drop due to header parse error or discard eligible. No Learning. 000010b = DROP_PERR — Uncorrectable ECC or parity error detected while doing lookup. No Learning. 000011b = TRAP_RESERVED_MAC — Trap IEEE reserved MAC address frame. 000100b = TRAP_RESERVED_MAC_REMAP — Trap IEEE reserved MAC address frame at remapped priority. 001010b = DROP_CONTROL — Drop IEEE MAC CONTROL Ethertype (includes PAUSE frames). No Learning. 001011b = DROP_RESERVED_MAC — Drop IEEE Reserved MAC address frame. No Learning. 001100b = DROP_TAG — Drop due to tagging rules violation. No Learning. 001101b = DROP_SMAC — Drop due to invalid SMAC. No Learning. 001110b = DROP_SV_MOVED — Drop due to MAC security violation (moved address). No Learning. 001111b = TRAP_CPU_ADDR — Trap CPU MAC address. 010000b = DROP_IV — Drop due to VLAN ingress membership violation. No Learning. 010001b = DROP_INGRESS_STP_NON_LEARN — Drop due to ingress STP check (non-learning state). No Learning. 010010b = DROP_INGRESS_STP_LEARN — Drop due to ingress STP check (learning state). 010011b = DROP_FFU — Drop due to FFU action. No Learning. 010100b = TRAP_FFU — Trap due to FFU action. 010101b = TRAP_ICMP_TTL — Trap due to TTL ≤ 1 for ICMP frames. 010110b = TRAP_IP_OPTION — Trap IP frames with options. 010111b = TRAP_MTU_VIO — Trap due to MTU violation. 011000b = TRAP_IGMP — Trap due to IGMP. 011001b = TRAP_TTL — Trap due to TTL ≤ 1 for non-ICMP frames. 011010b = DROP_TTL — Drop due to TTL ≤ 1. 011011b = DROP_DLF — Drop due to flood control of DLF frames (null flood glortdest). No Learning. 011100b = DROP_CAM_MISS — Drop due to GLORT_CAM miss. No learning. 011101b = DROP_NULL_GLORTDEST — Drop due to a null destination mask. 011110b = DROP_EV — Drop due to VLAN egress violation. 011111b = DROP_POLICER — Drop due to policer. 100000b = DROP_EGRESS_STP — Drop due to egress STP check. 100001b = DROP_LOOPBACK — Drop due to port or VLAN reflection disabled. 100010b = GLORT — FFU/ARP DGLORT forwarded. 100011b = FLOOD — Flood due to destination MAC miss in MA_TABLE. 100100b = SWITCH_RESERVED_MAC — Switch IEEE reserved MAC address frame. 100101b = FORWARD_NORMAL — Forward normally. 100110b = LOG_INGRESS_FFU — Frame copied to the CPU as a result of FFU action. 100111b = LOG_RESERVED_MAC — Log IEEE reserved MAC address frame. 101000b = LOG_ARP_REDIRECT — Log ARP redirect. 101001b = LOG_IP_ICMP — Multicast ICMP frame was copied to the CPU because its TTL was ≤ 1. 101010b = LOG_IP_TTL — Multicast frame was copied to the CPU because its TTL was ≤ 1. 101011b = HDR_TIMEOUT — Header was too long to be completely parsed. 101100b = MIRROR_INGRESS_FFU — Frame was mirrored as a result of FFU action. All other values are reserved.

11.17.3 TRIG_APPLY Registers

11.17.3.1 TRIGGER_CONDITION_CFG[0..63]

Each 2-bit condition field specifies one of three defined match cases:

0x0 — Associated frame property must not equal the corresponding trigger parameter in order to match.

0x1 — Associated frame property must equal the corresponding trigger parameter in order to match.

0x2 — Match unconditionally.

A configuration value of 0x3 is reserved and causes undefined behavior if selected.

Address: **Atomicity:** **Reset Domains:**	**0xD70000 + 0x1*i + 0x0** **32** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MatchSA	1:0	RW	10b	Match frame's source MAC address against the *SA_ID* field in TRIGGER_CONDITION_PARAM (refer to Section 11.17.3.2). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchDA	3:2	RW	10b	Match frame's destination MAC address against the *DA_ID* field in TRIGGER_CONDITION_PARAM (refer to Section 11.17.3.2). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchHitSA	5:4	RW	10b	Match depending on whether a source MAC address lookup was performed and was found to exist in the table. 00b = Match if miss 01b = Match if hit 10b = Ignore this field 11b = Undefined
MatchHitDA	7:6	RW	10b	Match depending on whether the destination MAC address was found in the table. 00b = Match if miss 01b = Match if hit 10b = Ignore this field 11b = Undefined
MatchHitSADA	9:8	RW	10b	Match depending on whether the source or destination MAC address was found in the table (or both). 00b = Match if SMAC or DMAC is a miss (or both) 01b = Match if SMAC or DMAC is a hit (or both) 10b = Ignore this field 11b = Undefined
MatchVlan	11:10	RW	10b	Match depending on whether the *TrigID* from the VID_TABLE lookup matches *VID_ID* from TRIGGER_CONDITION_PARAM (refer to Section 11.17.3.2). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined

Address:	**0xD70000 + 0x1*i + 0x0**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MatchFFU	13:12	RW	10b	Match depending on whether the *TrigID* from the FFU's SET_TRIG action matches *FFU_ID*, when both are ANDed with *FFU_Mask* (refer to Section 11.17.3.3). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchSwitchPri	15:14	RW	10b	Match if the frame's switch priority matches *SwitchPri* from TRIGGER_CONDITION_PARAM (refer to Section 11.17.3.2). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchEtherType	17:16	RW	10b	Match if the frame's first non-VLAN EtherType matches the *EtherType* parameter in TRIGGER_CONDITION_TYPE when both are ANDed with *EtherTypeMask*. 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchDestGlort	19:18	RW	10b	Match if the destination glort associated with the frame matches *DestGlort* when both are ANDed with *GlortMask* (refer to Section 11.17.3.5). 00b = Match if not equal 01b = Match if equal 10b = Ignore this field 11b = Undefined
MatchByPrecedence	20	RW	0b	If 0b, begins a new precedence-evaluated trigger set, evaluated in parallel with all other precedence-evaluated sets.
MatchRandomNumber	21	RW	0b	Specifies one of two random number generators to compare against.
MatchRandomIfLess	22	RW	1b	When set to 1b, matches if 'random is less than or equal to 2^*MatchRandomThreshold*'; otherwise, matches if 'random is greater than or equal to 2^*MatchRandomThreshold*'.
MatchRandomThreshold	27:23	RW	0x18	Exponent used to calculate the threshold for rate-constrained matching. Threshold = 2^*MatchRandomThreshold*.
MatchTx	29:28	RW	00b	Compare destination mask for this frame to the destination mask defined in TRIGGER_CONDITION_TX[0..63]. 00b = MASK_Z — Match if (DMASK & TRIGGER_CONDITION_TX[x].*DestPortMask*) == 0 01b = MASK_NZ — Match if (DMASK & TRIGGER_CONDITION_TX[x].*DestPortMask*) != 0 10b = EXACT_EQ — Match if DMASK == TRIGGER_CONDITION_TX[x].*DestPortMask* 11b = EXACT_NE — Match if DMASK != TRIGGER_CONDITION_TX[x].*DestPortMask* To match any set of TX ports (including no TX ports), set *MatchTx* to 0b and set TRIGGER_CONDITION_TX[x].*DestPortMask* to 0b (default).
Reserved	31:30	RSV	00b	Reserved.

11.17.3.2 TRIGGER_CONDITION_PARAM[0..63]

Address:	**0xD70000 + 0x1*i + 0x40**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SA_ID	5:0	RW	0x0	Source MAC address ID. Compared against trigID from the source address' entry in MA_TABLE. A value of 0x0 can be used to match on no source address lookup (learning was disabled on the ingress port), and a value of 0x3F can be used to match on a source lookup miss.
DA_ID	11:6	RW	0x0	Destination MAC address ID. Compared against TrigID from the destination address' entry in MA_TABLE. A value of 0x3F may be used to match on a destination lookup miss.
VID_ID	17:12	RW	0x0	VLAN ID. Compared against *TrigID* from the VID_TABLE lookup.
SwitchPri	21:18	RW	0x0	Trigger parameter to compare against the frame's associated switch priority.
FrameClassMask	24:22	RW	0x7	Frame Class Mask. 001b = The trigger matches against unicast frames. 010b = The trigger matches against broadcast frames. 100b = The trigger matches against multicast frames. 111b = The trigger matches against all three types of frames (default). All other values are reserved.
RoutedMask	26:25	RW	11b	Routed Mask. 01b = The trigger matches on switched frames. 10b = The rigger matches on routed frames. 11b = The trigger matches on both switched and routed frames (default). All other values are reserved.
FtypeMask	30:27	RW	0x3	*FtypeMask*[i]==1 implies Trigger matches frames with FTYPE==i.
Reserved	31	RSV	0b	Reserved.

11.17.3.3 TRIGGER_CONDITION_FFU[0..63]

Address:	**0xD70000 + 0x1*i + 0x80**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FFU_ID	7:0	RW	0x0	ID to compare against the *TrigID* specified by an FFU SET_TRIG action. ***Note:*** 0b is a reserved value (it is the default value assigned by the FFU to the FFU_ID) and should not be used for normal use.
FFU_Mask	15:8	RW	0x0	Mask to be ANDed with both *FFU_ID* and the *TrigID* specified by an FFU SET_TRIG action prior to comparison.
Reserved	31:16	RSV	0x0	Reserved.

11.17.3.4 TRIGGER_CONDITION_TYPE[0..63]

Address: 0xD70000 + 0x1*i + 0xC0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
EtherType	15:0	RW	0x0	Type value to compare against the frame's first non-VLAN EtherType.
EtherTypeMask	31:16	RW	0x0	Mask to be ANDed with both the *EtherType* trigger parameter and the frame's first non-VLAN EtherType prior to comparison.

11.17.3.5 TRIGGER_CONDITION_GLORT[0..63]

Address: 0xD70000 + 0x1*i + 0x100
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DestGlort	15:0	RW	0x0	GLORT value to compare against the frame's destination GLORT.
GlortMask	31:16	RW	0x0	Mask to be ANDed with both the *DestGlort* trigger parameter and the frame's destination GLORT prior to comparison.

11.17.3.6 TRIGGER_CONDITION_RX[0..63]

The source port mask is unconditionally ANDed with this mask. The result must be non-zero for this condition to match.

Address: 0xD70000 + 0x2*i + 0x180
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
SrcPortMask	47:0	RW	0x0	Source port mask. The bit corresponding to the frame's source port must be set for the trigger to match. Setting to 0xFFFFFFFFFFFF causes the trigger to match regardless of where the frame is coming from. Setting to 0 causes the trigger to never match regardless of where the frame is coming from.
Reserved	63:48	RSV	0x0	Reserved.

11.17.3.7 TRIGGER_CONDITION_TX[0..63]

Address: 0xD70000 + 0x2*i + 0x200
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DestPortMask	47:0	RW	0x0	Trigger's destination port mask used to compare against actual destination mask of the frame. For details on comparison options, refer to TRIGGER_CONDITION_CFG[i].*MatchTx* (Section 11.17.3.1).
Reserved	63:48	RSV	0x0	Reserved.

11.17.3.8 TRIGGER_CONDITION_AMASK_1[0..63]

Address: 0xD70000 + 0x1*i + 0x280
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
HandlerActionMask	31:0	RW	0x0	Defines which action enables this trigger (first 32 bits). Exact bits defined in following location. Uses type *AMASK* (see Table 11-27).

11.17.3.9 TRIGGER_CONDITION_AMASK_2[0..63]

Address: 0xD70000 + 0x1*i + 0x2C0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
HandlerActionMask	12:0	RW	0x30	Defines which action enables this trigger (next 13 bits, action bits 32 to 44). Exact bits defined at following location. Uses type *AMASK* (see Table 11-27).
Reserved	31:13	RSV	0x0	Reserved.

11.17.3.10 TRIGGER_ACTION_CFG_1[0..63]

Address: 0xD70000 + 0x1*i + 0x300
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ForwardingAction	1:0	RW	00b	Forwarding action: 00b = Leave as is 01b = Forward 10b = Redirect 11b = Drop
TrapAction	3:2	RW	00b	Trap action: 00b = Leave as is 01b = Trap 10b = Log 11b = Do not Trap or Log
MirroringAction	4	RW	0b	Mirroring action: 0b = Leave as is 1b = Mirror
SwitchPriAction	5	RW	0b	Switch Priority modification action: 0b = Leave as-is 1b = Reassign
VlanAction	6	RW	0b	Egress VLAN modification action: 0b = Leave as-is 1b = Reassign

Address: **0xD70000 + 0x1*i + 0x300**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
LearningAction	8:7	RW	00b	Learning action: 00b = Normal TCN updates 01b = Do not send a TCN update even if the SMAC was learned or moved. 10b = Reserved.) 11b = Reserved.
RateLimitAction	9	RW	0b	Rate limiter action 0b = Leave as is 1b = Apply rate limit
Reserved	31:10	RSV	0x0	Reserved.

11.17.3.11 TRIGGER_ACTION_CFG_2[0..63]

Address: **0xD70000 + 0x1*i + 0x340**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
NewSwitchPri	3:0	RW	0x0	New switch priority (applicable when *SwitchPriAction* is 1b.)
NewEVID	15:4	RW	0x0	New egress VLAN ID (applicable when *VlanAction* is 1b.)
RateLimitNum	19:16	RW	0x0	Rate limiter number (applicable when *RateLimitAction* is 1b.)
Reserved	31:20	RSV	0x0	Reserved.

11.17.3.12 TRIGGER_ACTION_GLORT[0..63]

Address: **0xD70000 + 0x1*i + 0x380**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
NewDestGlort	15:0	RW	0x0	New destination GLORT when *ForwardingAction* is 1 or 2.
NewDestGlortMask	31:16	RW	0x0	Only bits that are set to 1b are overwritten by *NewDestGlort*.

11.17.3.13 TRIGGER_ACTION_DMASK[0..63]

Address: **0xD70000 + 0x2*i + 0x400**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
NewDestMask	47:0	RW	0x0	New port destination mask when *ForwardingAction* is 2.
FilterDestMask	48	RW	1b	Destination mask is LAG-filtered when set to 1b when *ForwardingAction* is 2.
Reserved	63:49	RSV	0x0	Reserved.

11.17.3.14 TRIGGER_ACTION_MIRROR[0..63]

Address: 0xD70000 + 0x1*i + 0x480
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
MirrorSelect	0	RW	0b	Select mirror 0 or mirror 1 to override the profile index.
MirrorProfileIndex	6:1	RW	0x0	Mirror profile index value to set for the profile selected.
Reserved	31:7	RSV	0x0	Reserved.

11.17.3.15 TRIGGER_ACTION_DROP[0..63]

The drop mask is applied to the frame's destination mask when *ForwardingAction* in TRIGGER_ACTION_CFG_1 is set to 3 (drop).

Address: 0xD70000 + 0x2*i + 0x500
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DropMask	47:0	RW	0x0	Bits set to 1b are cleared in the frame's destination mask.
Reserved	63:48	RSV	0x0	Reserved.

11.17.3.16 TRIGGER_STATS[0..63]

Address: 0xD70000 + 0x2*i + 0x580
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Count	63:0	vRW	0x0	Number of times that this trigger was applied to a frame. Does not include triggers that fire but are overridden by higher-precedence triggers. Applies even if the frame is tail dropped due to an error.

11.17.3.17 TRIGGER_IP

Address: 0xD70000 + 0x600
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Pending	63:0	vCW1	0x0	Interrupt pending bits. Set whenever the corresponding trigger fires.

11.17.3.18 TRIGGER_IM

Address:	**0xD70000 + 0x602**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Mask	63:0	RW	0xFFFFFFFFFFFFFFFF	Interrupt mask bits.

11.17.3.19 TRIGGER_RATE_LIM_EMPTY

Address:	**0xD70000 + 0x604**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Empty	15:0	vRO	0xFFFF	Bit set for each rate limiter with an empty token bucket (TRIGGER_RATE_LIM_USAGE=0)
Reserved	31:16	RSV	0x0	Reserved.

11.17.3.20 TRIGGER_RATE_LIM_CFG_2[0..15]

Address:	**0xD70000 + 0x2*i + 0x620**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DropMask	47:0	RW	0x0	Drop mask to apply to the frame when the rate limiter exceeds its configured rate.
Reserved	63:48	RSV	0x0	Reserved.

11.18 TRIG_USAGE Registers Description

11.18.1 TRIG_USAGE Map

Table 11-28 Triggers Usage Registers Map

Name	Address (Base = 0xE78000)	Atomicity	Brief Description
TRIGGER_RATE_LIM_CFG_1[0..15]	Base + 0x1*i + 0x0	32	Rate limiter configuration (Capacity and Rate).
TRIGGER_RATE_LIM_USAGE[0..15]	Base + 0x1*i + 0x10	32	Rate limiter token-bucket levels (units of 1/16th bytes).

11.18.2 TRIG_USAGE Registers

11.18.2.1 TRIGGER_RATE_LIM_CFG_1[0..15]

Address: **0xE78000 + 0x1*i + 0x0**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Capacity	11:0	RW	0x0	Capacity of rate limiter token bucket in 1024-byte units. Maximum burst allowance is 4 MB, or 2 MB for near-maximal rate limits. This must not be set within *refillAmount* of the maximum: `Capacity <= ((0xfff << 14) - ((RateMantissa * 32) >> (RateExponent + 2))) >> 14`
RateMantissa	23:12	RW	0x0	Floating point mantissa of the token bucket's refill rate. Amount of refill is (*RateMantissa* * 32) >> (*RateExponent*+2) per 16 FH clock cycles, in units of 1/16th bytes.
RateExponent	25:24	RW	00b	Floating point exponent of the token bucket's refill rate. Amount of refill is (*RateMantissa* * 32) >> (*RateExponent*+2) per 16 FH clock cycles, in units of 1/16th bytes.
Reserved	31:26	RSV	0x0	Reserved.

11.18.2.2 TRIGGER_RATE_LIM_USAGE[0..15]

Address: **0xE78000 + 0x1*i + 0x10**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Level	26:0	vRO	0x0	Current rate limiter token bucket level in units of 1/16th bytes. Value is signed two's complement notation.
Reserved	31:27	RSV	0x0	Reserved.

11.19 CM_APPLY Registers Description

11.19.1 CM_APPLY Map

Table 11-29 Congestion Management Registers (reserved 64 KW) Map

Name	Address (Base = 0xD40000)	Atomicity	Brief Description
CM_APPLY_SWITCH_PRI_TO_TC	Base + 0x0	64	Maps switch priority to traffic class.
CM_APPLY_TC_TO_SMP	Base + 0x2	32	Defines the SMP membership per traffic class.
CM_APPLY_TX_SOFTDROP_CFG[0..47][0..15]	Base + 0x10*j + 0x1*i + 0x400	32	Soft drop configuration per TX port, switch priority.
CM_APPLY_SOFTDROP_CFG[0..15]	Base + 0x1*i + 0x800	32	Soft drop configuration.
CM_APPLY_DROP_COUNT[0..47]	Base + 0x2*i + 0x880	64	Per-port TX CM DROP counter.
MCAST_EPOCH	Base + 0x900	32	Defines epoch.

11.19.2 CM_APPLY Registers

11.19.2.1 CM_APPLY_SWITCH_PRI_TO_TC

Address:	**0xD40000 + 0x0**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
tc[0..15]	47:0	RW	000b	(16 x 3-bits) Maps switch priority (SP) to traffic class [0..7]. An arbitrary 16-to-8 mapping.
Reserved	63:48	RSV	0x0	Reserved.

11.19.2.2 CM_APPLY_TC_TO_SMP

Defines the SMP membership per traffic class. The choices are:

- 0: SMP 0
- 1: SMP 1

Address:	**0xD40000 + 0x2**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
smp[0..7]	7:0	RW	0b	(8 x 1-bit) SMP Membership for traffic class N.
Reserved	31:8	RSV	0x0	Reserved.

11.19.2.3 CM_APPLY_TX_SOFTDROP_CFG[0..47][0..15]

Specifies soft drop configuration per TX port and switch priority.

Address:	**0xD40000 + 0x10*j + 0x1*i + 0x400**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SoftDropOnSmpFree	0	RW	0b	Enables dynamic soft drop with drop probabilities proportional to the remaining free space in the SMP.
SoftDropOnPrivate	1	RW	0b	If *SoftDropOnSmpFree* is not set, this enables soft drop when CM_TX_TC_PRIVATE_WM is exceeded.
Reserved	31:2	RSV	0x0	Reserved.

11.19.2.4 CM_APPLY_SOFTDROP_CFG[0..15]

Specifies software configuration.

Address:	**0xD40000 + 0x1*i + 0x800**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
JitterBits	2:0	RW	000b	Selects the number of random jitter bits that are added to *SoftDropSegmentLimit* (0..7).
Reserved	31:3	RSV	0x0	Reserved.

11.19.2.5 CM_APPLY_DROP_COUNT[0..47]

Address:	**0xD40000 + 0x2*i + 0x880**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Frames	47:0	vRW	0x0	Per-port TX CM DROP frame counter.
Reserved	63:48	RSV	0x0	Reserved.

11.19.2.6 MCAST_EPOCH

Address:	**0xD40000 + 0x900**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Current	0	RW	0b	Defines current epoch for multicast garbage collection.
Reserved	31:1	RSV	0x0	Reserved.

11.20 CM_USAGE Registers Description

11.20.1 CM_USAGE Map

Table 11-30 Congestion Management Registers (reserved 64 KW) Map

Name	Address (Base = 0xE60000)	Atomicity	Brief Description
CM_SWEEPER_SWITCH_PRI_TO_TC	Base + 0x0	64	Maps switch priority to traffic class.
CM_SWEEPER_TC_TO_SMP	Base + 0x2	32	Defines the SMP membership per traffic class.
CM_TX_TC_PRIVATE_WM[0..47][0..7]	Base + 0x8*j + 0x1*i + 0x200	32	Private watermark per traffic class per port.
CM_TX_TC_HOG_WM[0..47][0..7]	Base + 0x8*j + 0x1*i + 0x400	32	TX hog watermark.
CM_RX_SMP_PAUSE_WM[0..47][0..1]	Base + 0x2*j + 0x1*i + 0x600	32	RX pause watermark per SMP.
CM_RX_SMP_PRIVATE_WM[0..47][0..1]	Base + 0x2*j + 0x1*i + 0x680	32	RX private watermark per SMP.
CM_RX_SMP_HOG_WM[0..47][0..1]	Base + 0x2*j + 0x1*i + 0x700	32	Maximum number of segments allowed per receiver per SMP.
CM_PAUSE_RESEND_INTERVAL[0..47]	Base + 0x1*i + 0x780	32	Periodicity of Pause Message Transmits.
CM_PAUSE_BASE_FREQ	Base + 0x7C0	32	
CM_PAUSE_CFG[0..47]	Base + 0x1*i + 0x800	32	Various per-port configuration settings
CM_SHARED_WM[0..15]	Base + 0x1*i + 0x840	32	Shared watermark per switch priority.
CM_SHARED_SMP_PAUSE_WM[0..1]	Base + 0x2*j + 0x1*i + 0x850	32	RX shared pool pause watermark per SMP.
CM_GLOBAL_WM	Base + 0x852	32	Defines the global watermark.
CM_GLOBAL_CFG	Base + 0x853	32	
CM_TC_PC_MAP[0..47]	Base + 0x1*i + 0x880	32	PAUSE class to traffic class map.
CM_PC_SMP_MAP[0..47]	Base + 0x1*i + 0x8C0	32	Mapping from Pause Class to SMP.
CM_SOFTDROP_WM[0..15]	Base + 0x1*i + 0x900	32	Watermarks on RX controlling TX soft drop.
CM_SHARED_SMP_PAUSE_CFG[0..1]	Base + 0x2*i + 0x910	64	RX shared pool pause configuration per SMP.
TX_RATE_LIM_CFG[0..47][0..7]	Base + 0x8*j + 0x1*i + 0xA00	32	Egress Rate Limiter configuration per (TX, BSG).
TX_RATE_LIM_USAGE[0..47][0..7]	Base + 0x8*j + 0x1*i + 0xC00	32	Egress Rate Limiter token bucket credits per (TX, BSG).
CM_BSG_MAP[0..47]	Base + 0x1*i + 0xE00	32	Maps TCs into BSGs per port.
CM_TX_TC_USAGE[0..47][0..7]	Base + 0x8*j + 0x1*i + 0x1000	32	Stores the number of segments per traffic class per TX.
CM_RX_SMP_USAGE[0..47][0..1]	Base + 0x2*j + 0x1*i + 0x1200	32	Stores the number of segments per port per SMPs.
MCAST_EPOCH_USAGE[0..1]	Base + 0x1*i + 0x1280	32	Stores the number of segments per port per SMPs.
CM_SHARED_SMP_USAGE[0..1]	Base + 0x1*i + 0x1282	32	Stores per-epoch usage.

Table 11-30 Congestion Management Registers (reserved 64 KW) Map [Continued]

Name	Address (Base = 0xE60000)	Atomicity	Brief Description
CM_SMP_USAGE[0..1]	Base + 0x1*i + 0x1284	32	Stores the number of segments in the shared pool for each SMP.
CM_GLOBAL_USAGE	Base + 0x1286	32	Stores the total number of segments used.
CM_PAUSE_GEN_STATE[0..47]	Base + 0x1*i + 0x12C0	32	Specifies the pause state of each port per SMP.
CM_PAUSE_RCV_TIMER	Base + 0x1300	32	Base time reference for pause timers.
CM_PAUSE_RCV_PORT_TIMER[0..47]	Base + 0x1*i + 0x1340	32	Per port timer.
CM_EGRESS_PAUSE_COUNT[0..47]	Base + 0x4*i + 0x1400	64	

11.20.2 CM_USAGE Registers

11.20.2.1 CM_SWEEPER_SWITCH_PRI_TO_TC

Address:	**0xE60000 + 0x0**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
tc[0..15]	47:0	RW	000b	(16 x 3-bits) Maps switch priority N to traffic class [0..7].
Reserved	63:48	RSV	0x0	Reserved.

11.20.2.2 CM_SWEEPER_TC_TO_SMP

Defines the SMP membership per traffic class. The choices are:

- 0: SMP 0
- 1: SMP 1

Address:	**0xE60000 + 0x2**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
smp[0..7]	7:0	RW	0b	(8 x 1-bit) SMP Membership for traffic class N.
Reserved	31:8	RSV	0x0	Reserved.

11.20.2.3 CM_TX_TC_PRIVATE_WM[0..47][0..7]

Defines the watermark above which a transmit frame copy on this port can potentially be dropped.

Address: **0xE60000 + 0x8*j + 0x1*i + 0x200**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x7FFF	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.4 CM_TX_TC_HOG_WM[0..47][0..7]

Defines the hog watermark per port and per traffic class, above which frames are not queued on this port.

Address: **0xE60000 + 0x8*j + 0x1*i + 0x400**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x7FFF	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.5 CM_RX_SMP_PAUSE_WM[0..47][0..1]

Defines the watermark per SMP per RX above which pause frames are started or stopped.

Address: **0xE60000 + 0x2*j + 0x1*i + 0x600**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
PauseOn	14:0	RW	0x7FFF	Level at which a pause ON is sent.
Reserved	15	RSV	0b	Reserved.
PauseOff	30:16	RW	0x7FFF	Level at which a pause OFF is sent.
Reserved	31	RSV	0b	Reserved.

11.20.2.6 CM_RX_SMP_PRIVATE_WM[0..47][0..1]

Defines the private watermark per SMP per RX above which frames can get potentially dropped.

Address: **0xE60000 + 0x2*j + 0x1*i + 0x680**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x7FFF	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.7 CM_RX_SMP_HOG_WM[0..47][0..1]

Defines the maximum number of segments allowed per receiver per SMP before a new frame is discarded.

Address: 0xE60000 + 0x2*j + 0x1*i + 0x700
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x7FFF	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.8 CM_PAUSE_RESEND_INTERVAL[0..47]

The periodicity at which a pause ON frame (a frame with pause-timer value non-zero) is generated is given by:

(number of sweeper ports)* FABRIC_CLOCK_PERIOD * INTERVAL

This value is configurable per-port. Exact timing of when a pause frame appears depends load conditions, MTU.

Address: 0xE60000 + 0x1*i + 0x780
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
INTERVAL	15:0	RW	0x100	Interval between pause regenerations
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.9 CM_PAUSE_BASE_FREQ

Defines the base frequency of pause quanta, relative to the FC clock. Set based on PLL_FABRIC_CTRL.

Address: 0xE60000 + 0x7C0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
M	9:0	RW	0x0	Set to: `PLL_FABRIC_CTRL.FbDiv255 * (2*(PLL_FABRIC_CTRL.FbDiv4+1))`
N	19:10	RW	0xFF	Set to: `PLL_FABRIC_CTRL.RefDiv * PLL_FABRIC_CTRL.OutDiv`
Reserved	31:20	RSV	0x0	Reserved.

11.20.2.10 CM_PAUSE_CFG[0..47]

Address:	**0xE60000 + 0x1*i + 0x800**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
PauseMask	7:0	RW	0xFF	Bit mask enabling pause for each traffic class (TC). An egress queue does not pause if the corresponding bit is set to 0b.
PauseQuantaMultiplier	23:8	RW	0xA0	*PauseQuantaMultiplier* and *PauseQuantaDivisor* define the length of a pause quanta relative to a 1.28 ns base interval: `real_pause_time = pause_time * PauseQuantaMultiplier / PauseQuantaDivisor * 1.28 ns` For correct operation, *PauseQuantaMultiplier* must exceed a minimum value. The following (*PauseQuantaMultiplier*:*PauseQuantaDivisor*) settings are recommended for different port speeds: • 100 Gb/s — (64:16) • 40 Gb/s — (160:16) • 25 Gb/s — (160:10) • 10 Gb/s — (160:4) • 1 Gb/s — (400:1) • 100 Mb/s — (4000:1) • 10 Mb/s — (40000:1)
PauseQuantaDivisor	28:24	RW4	0x4	See description of *PauseQuantaMultiplier*.
Reserved	31:29	RSV	000b	Reserved.

11.20.2.11 CM_SHARED_WM[0..15]

Defines the shared watermark per switch priority above which packets are dropped if the shared pool usage, CM_SHARED_SMP_USAGE for the SMP used by this switch priority is above this limit.

Address:	**0xE60000 + 0x1*i + 0x840**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x21C	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.12 CM_SHARED_SMP_PAUSE_WM[0..1]

Defines the watermark per SMP compared against CM_SHARED_SMP_USAGE for sending pause frames.

Address:	**0xE60000 + 0x1*i + 0x850**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
PauseOn	14:0	RW	0x7FFF	Level at which a pause ON is sent.
Reserved	15	RSV	0b	Reserved
PauseOff	30:16	RW	0x7FFF	Level at which a pause OFF is sent.
Reserved	31	RSV	0b	Reserved.

11.20.2.13 CM_GLOBAL_WM

Defines the global watermark above which no new frames are accepted.

Address: 0xE60000 + 0x852
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
watermark	14:0	RW	0x5E20	Number of segments.
Reserved	31:15	RSV	0x0	Reserved.

11.20.2.14 CM_GLOBAL_CFG

Global Congestion Management Configuration.

Address: 0xE60000 + 0x853
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ifgPenalty	7:0	RW	0x14	The additional charge in bytes to apply per frame for the purposes of egress rate limiting.
forcePauseOn	8	RW	0b	Force all ports to be paused.
forcePauseOff	9	RW	0b	Force all ports to be unpaused.
WmSweeperEnable	10	RW	0b	Enable the CM watermark evaluation sweeper. Must be set to 1b for normal CM operation.
PauseGenSweeperEnable	11	RW	0b	Enable the CM pause frame generation sweeper. Must be set to 1b for normal CM operation.
PauseRecSweeperEnable	12	RW	0b	Enable the CM pause frame reception sweeper. This must be set to 1b for normal CM operation.
NumSweeperPorts	18:13	RW	0x0	Number of ports to sweep. Must be set to the number of ports (or a minimum of 4) for normal CM operation.
Reserved	31:19	RSV	0x0	Reserved

11.20.2.15 CM_TC_PC_MAP[0..47]

Specifies the mapping of pause classes referenced by class-based PAUSE frames to internal traffic classes. Since a given pause class may map to any number of internal traffic classes, the mapping is encoded in the inverted sense (TC to PC).

Address: 0xE60000 + 0x1*i + 0x880
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
PauseClass[0..7]	23:0	RW	000b	(8 x 3-bits) Specifies the Pause Class each scheduling traffic class [0..7] maps to. When a class-based PAUSE frame is received specifying that Pause Class 'pc' be paused, all traffic classes with *PauseClass*[tc] == pc are paused.
Reserved	31:24	RSV	0x0	Reserved.

11.20.2.16 CM_PC_SMP_MAP[0..47]

Specifies the mapping from class-based pause classes to RX memory partition number (SMP) for purposes of PAUSE frame generation. As the status of the PAUSE-ON/OFF watermarks defined on an SMP change, PAUSE frames are generated updating the flow-control status of all pause classes mapped to the SMP.

Address:	**0xE60000 + 0x1*i + 0x8C0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SMP[0..7]	31:0	RW	1000b	(8 x 4-bits) Specifies the SMP each Pause Class maps to. An out-of-bounds value (greater than 1) defines an inactive pause class, i.e. one that always remains in a PAUSE-OFF state.

11.20.2.17 CM_SOFTDROP_WM[0..15]

Configures TX soft dropping based on RX usage. Whenever the total RX usage within an SMP exceeds *SoftDropSegmentLimit* (plus a six-bit random jitter term), TX soft dropping is enabled such that the TX_TC queues consuming the most memory (and exceeding their TX_TC watermarks) are penalized most. Frames are dropped with 100% probability if the RX SMP usage exceeds *HogSegmentLimit*. Indexed by ISL_PRI. All watermarks are compared to CM_SHARED_USAGE[SMP].*SegmentCount* where SMP is the value mapped from the frame's ISL_PRI.

Address:	**0xE60000 + 0x1*i + 0x900**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SoftDropSegmentLimit	14:0	RW	0x7FFF	TX soft dropping is enabled whenever the total RX SMP segment usage exceeds this watermark plus a random six-bit jitter term.
Reserved	15	RSV	0b	Reserved.
HogSegmentLimit	30:16	RW	0x7FFF	Frames belonging to this ISL_PRI are dropped with 100% probability whenever the total RX SMP segment usage exceeds this watermark.
Reserved	31	RSV	0b	Reserved.

11.20.2.18 CM_SHARED_SMP_PAUSE_CFG[0..1]

Defines whether RX shared pool pause flow control is enabled per port.

Address:	**0xE60000 + 0x2*i + 0x910**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
EnableMask	47:0	RW	0x0	Port mask that enables RX shared pool per port. Set to 1b to enable.
Reserved	63:48	RSV	0x0	Reserved.

11.20.2.19 TX_RATE_LIM_CFG[0..47][0..7]

Address:	**0xE60000 + 0x8*j + 0x1*i + 0xA00**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Capacity	12:0	RW	0x1FFF	Defines saturation value in 1 KB units.
RateFrac	20:13	RW	0xFF	*RateFrac*/256 bytes are added to the token bucket every 48 FC clocks. If *RateFrac* and *RateUnit* are both set to their maximum value, the rate limiter is disabled.
RateUnit	31:21	RW	0x7FF	*RateUnit* bytes are added to the token bucket every 48 FC clocks. If *RateFrac* and *RateUnit* are both set to their maximum value, the rate limiter is disabled.

11.20.2.20 TX_RATE_LIM_USAGE[0..47][0..7]

Address:	**0xE60000 + 0x8*j + 0x1*i + 0xC00**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Frac	7:0	vRO	0x0	*Frac*/256 bytes of credit in the token bucket.
Units	31:8	vRO	0x0	Bytes of credit in the token bucket.

11.20.2.21 CM_BSG_MAP[0..47]

Address:	**0xE60000 + 0x1*i + 0xE00**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BSG[0..7]	31:0	RW	0x0	(8 x 4-bits) Indexed by TC, provides the BSG number for that TC. Normally *BSG*s are configured as supersets of scheduling groups, but the hardware supports a fully general mapping. The MSG of each *BSG* is Reserved.

11.20.2.22 CM_TX_TC_USAGE[0..47][0..7]

Stores the number of segments per TX port per traffic class.

Address:	**0xE60000 + 0x8*j + 0x1*i + 0x1000**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.23 CM_RX_SMP_USAGE[0..47][0..1]

Stores the number of segments per port per SMPs.

Address:	**0xE60000 + 0x2*j + 0x1*i + 0x1200**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.24 MCAST_EPOCH_USAGE[0..1]

Tracks the total number of segments used currently in the switch per value of MCAST_EPOCH. Used to safely garbage-collect old entries from MCAST_VLAN_TABLE, after all frames using them have left the switch.

Address:	**0xE60000 + 0x1*i + 0x1280**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.25 CM_SHARED_SMP_USAGE[0..1]

Contains the current number of segments stored in each SMP excluding those used in the RX private queues. A segment is only counted in this shared pool if all private watermarks have been exceeded.

Address:	**0xE60000 + 0x1*i + 0x1282**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.26 CM_SMP_USAGE[0..1]

Contains the current number of segments stored in each SMP.

Address:	**0xE60000 + 0x1*i + 0x1284**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.27 CM_GLOBAL_USAGE

Stores the total number of segments used currently in the switch.

Address:	**0xE60000 + 0x1286**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
count	15:0	vRO	0x0	Number of segments.
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.28 CM_PAUSE_GEN_STATE[0..47]

Specifies the pause state of each port per SMP. For each port in each SMP, 1b means it is paused and 0b means it is not paused.

Address:	**0xE60000 + 0x1*i + 0x12C0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
smp0	0	vRO	0b	
smp1	1	vRO	0b	
Reserved	31:2	RSV	0x0	Reserved.

11.20.2.29 CM_PAUSE_RCV_TIMER

Base time reference for pause timers, in units of 400 Gb pause quanta (1.28 ns).

Address:	**0xE60000 + 0x1300**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Q	15:0	vRO	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.30 CM_PAUSE_RCV_PORT_TIMER[0..47]

Per port timer, tracking CM_PAUSE_RCV_TIMER.

Address:	**0xE60000 + 0x1*i + 0x1340**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Q	15:0	vRO	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.20.2.31 CM_EGRESS_PAUSE_COUNT[0..47]

Per port, per pause class timer. Does not initialize using BIST pattern fill. Each entry must be written to 0 prior to setting PauseRecSweeperEnable. No writes should be issued after that time.

Address:	**0xE60000 + 0x4*i + 0x1400**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
PauseTime[0..7]	127:0	vRW	0x0	(8 x 16-bits) Remaining pause time per pause class, in units of pause quanta.

11.21 MOD Registers Description

11.21.1 MOD Map

Table 11-31 Egress Modification Registers Map

Name	Address (Base = 0xE80000)	Atomicity	Brief Description
MOD_MAX_MGMT_WAIT_CYCLE	Base + 0xF8	64	
MOD_IP	Base + 0x106	64	
MOD_IM	Base + 0x108	64	
MOD_SRAM_BIST_OUT	Base + 0x10A	64	
MOD_SRAM_ERROR_WRITE	Base + 0x10C	64	
MOD_PAUSE_SMAC	Base + 0x10E	64	Pause SMAC
MOD_ROUTER_SMAC[0..15]	Base + 0x2*i + 0x120	64	Router SMAC
MOD_MCAST_VLAN_TABLE[0..32767]	Base + 0x2*i + 0x10000	64	
MOD_VLAN_TAG_VID1_MAP[0..4095]	Base + 0x2*i + 0x20000	64	
MOD_VID2_MAP[0..4095]	Base + 0x2*i + 0x22000	64	
MOD_MIRROR_PROFILE_TABLE[0..63]	Base + 0x2*i + 0x24000	64	
MOD_PER_PORT_CFG_1[0..47]	Base + 0x2*i + 0x24080	64	
MOD_PER_PORT_CFG_2[0..47]	Base + 0x2*i + 0x24100	64	
MOD_VPRI1_MAP[0..47]	Base + 0x2*i + 0x24180	64	
MOD_VPRI2_MAP[0..47]	Base + 0x2*i + 0x24200	64	
MOD_VLAN_ETYPE[0..3]	Base + 0x2*i + 0x24280	64	
MOD_STATS_CFG[0..47]	Base + 0x2*i + 0x24300	64	
MOD_STATS_BANK_FRAME[0..1][0..767]	Base + 0x800*j + 0x2*i + 0x25000	64	
MOD_STATS_BANK_BYTE[0..1][0..767]	Base + 0x800*j + 0x2*i + 0x26000	64	

11.21.2 MOD Enumerated Data Types

Table 11-32 MOD Enumerated Data Types

Name	Width	Description
TX_STATS_BANK_7	4	0000b = L2Ucast — L2 unicast DMAC transmitted (good CRC). 0001b = L2Mcast — L2 multicast DMAC transmitted (good CRC). 0010b = L2Bcast — L2 broadcast DMAC transmitted (good CRC). 0011b = TxError — Frames transmitted with bad CRC. 0100b = TimeoutDrop — Dropped due to timeout in switch memory. 0101b = TxErrorDrop — Dropped due to tx error (SNF error from FC). 0110b = TxEccDrop — Dropped due to uncorrectable error (incl scheduler memories). 0111b = LoopbackDrop — Dropped due to loopback suppression in MODIFY. 1000b = TTL1Drop — Dropped due to routed frame having TTL 1 or 0. 1001b = PAUSE — PAUSE frame sent. 1010b = CBPause — Class-Based (PFC) PAUSE frame sent. All other values are reserved.
TX_STATS_BANK_8	4	0000b = Len_lt_64 — length < 64 0001b = Len_eq_64 — length = 64 0010b = Len_65_127 — 64 < length < 128 0011b = Len_128_255 — 128 ≤ length ≤ 255 0100b = Len_256_511 — 256 ≤ length ≤ 511 0101b = Len_512_1023 — 512 ≤ length ≤ 1023 0110b = Len_1024_1522 — 1024 ≤ length ≤ 1522 0111b = Len_1523_2047 — 1523 ≤ length ≤ 2047 1000b = Len_2048_4095 — 2048 ≤ length ≤ 4095 1001b = Len_4096_8191 — 4096 ≤ length ≤ 8191 1010b = Len_8192_10239 — 8192 ≤ length ≤ 10239 1011b = Len_ge_10240 — length ≥ 10240 All other values are reserved.

11.21.3 MOD Registers

DEBUG ONLY. Reserved in normal operation.
CDC debug shim configuration register for channel array_sub[5]. For debug only.

11.21.3.1 MOD_MAX_MGMT_WAIT_CYCLE

Defines maximal cycles that MGMT request can wait

Address: 0xE80000 + 0xF8
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Value	7:0	RW	0x10	Maximal waiting clock cycles of MGMT request when it is blocked by frame flow. Only a positive value is valid.
Reserved	31:8	RSV	0x0	Reserved.

11.21.3.2 MOD_IP

Defines MODIFY IP register. 1b indicates an interrupt is pending.

Address:	**0xE80000 + 0x106**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
HeadStorage_CERR	0	CW1	0b	Head storage SRAM correctable error.
HeadStorage_UERR	1	CW1	0b	Head storage SRAM uncorrectable error.
McVlanTable_CERR	2	CW1	0b	MC_VLAN_TABLE SRAM correctable error.
McVlanTable_UERR	3	CW1	0b	MC_VLAN_TABLE SRAM uncorrectable error.
PerPortCfg1_CERR	4	CW1	0b	PER_PORT_CFG_1 SRAM correctable error.
PerPortCfg1_UERR	5	CW1	0b	PER_PORT_CFG_1 SRAM uncorrectable error.
PerPortCfg2_CERR	6	CW1	0b	MOD_PER_PORT_CFG_2 SRAM correctable error.
PerPortCfg2_UERR	7	CW1	0b	PORT_CFG_2 SRAM uncorrectable error.
Vpri1Map_CERR	8	CW1	0b	VPRI1_MAP SRAM correctable error.
Vpri1Map_UERR	9	CW1	0b	VPRI1_MAP SRAM uncorrectable error.
Vpri2Map_CERR	10	CW1	0b	VPRI2_MAP SRAM correctable error.
Vpri2Map_UERR	11	CW1	0b	VPRI2_MAP SRAM uncorrectable error.
MirProfTable_CERR	12	CW1	0b	MIRROR_PROFILE_TABLE SRAM correctable error.
MirProfTable_UERR	13	CW1	0b	MIRROR_PROFILE_TABLE SRAM uncorrectable error.
VtagVid1Map_CERR	14	CW1	0b	VTAG_VID1_MAP SRAM correctable error.
VtagVid1Map_UERR	15	CW1	0b	VTAG_VID1_MAP SRAM uncorrectable error.
Vid2Map_CERR	16	CW1	0b	VID2_MAP SRAM correctable error.
Vid2Map_UERR	17	CW1	0b	VID2_MAP SRAM uncorrectable error.
Ctrl2DpFifo0_CERR	18	CW1	0b	Ctrl2Dp FIFO 0 SRAM correctable error.
Ctrl2DpFifo0_UERR	19	CW1	0b	Ctrl2Dp FIFO 0 SRAM uncorrectable error.
Ctrl2DpFifo1_CERR	20	CW1	0b	Ctrl2Dp FIFO 1 SRAM correctable error.
Ctrl2DpFifo1_UERR	21	CW1	0b	Ctrl2Dp FIFO 1 SRAM uncorrectable error.
Ctrl2DpFifo2_CERR	22	CW1	0b	Ctrl2Dp FIFO 2 SRAM correctable error.
Ctrl2DpFifo2_UERR	23	CW1	0b	Ctrl2Dp FIFO 2 SRAM uncorrectable error.
Ctrl2DpFifo3_CERR	24	CW1	0b	Ctrl2Dp FIFO 3 SRAM correctable error.
Ctrl2DpFifo3_UERR	25	CW1	0b	Ctrl2Dp FIFO 3 SRAM uncorrectable error.
StatsFrameCnt0_CERR	26	CW1	0b	Stats frame count 0 SRAM correctable error.
StatsFrameCnt0_UERR	27	CW1	0b	Stats frame count 0 SRAM uncorrectable error.
StatsFrameCnt1_CERR	28	CW1	0b	Stats frame count 1 SRAM correctable error.
StatsFrameCnt1_UERR	29	CW1	0b	Stats frame count 1 SRAM uncorrectable error.
StatsByteCnt0_CERR	30	CW1	0b	Stats byte count 0 SRAM correctable error.
StatsByteCnt0_UERR	31	CW1	0b	Stats byte count 0 SRAM uncorrectable error.
StatsByteCnt1_CERR	32	CW1	0b	Stats byte count 1 SRAM correctable error.
StatsByteCnt1_UERR	33	CW1	0b	Stats byte count 1 SRAM uncorrectable error.
Refcount_CERR	34	CW1	0b	Refcount SRAM correctable error.
Refcount_UERR	35	CW1	0b	Refcount SRAM uncorrectable error.
EschedTxInfoFifo_CERR	36	CW1	0b	Esched TX Info FIFO correctable error.
EschedTxInfoFifo_UERR	37	CW1	0b	Esched TX Info FIFO uncorrectable error.
CmTxInfoFifo_CERR	38	CW1	0b	Cm TX Info FIFO correctable error.

Address: 0xE80000 + 0x106 **Atomicity:** 64 **Reset Domains:** DEVICE, MASTER, COLD, SWITCH				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CmTxInfoFifo_UERR	39	CW1	0b	Cm TX Info FIFO uncorrectable error.
MgmtRdFifo_CERR	40	CW1	0b	Mgmt read FIFO correctable error.
MgmtRdFifo_UERR	41	CW1	0b	Mgmt read FIFO uncorrectable error.
Reserved	63:42	RSV	0x0	Reserved.

11.21.3.3 MOD_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0xE80000 + 0x108 **Atomicity:** 64 **Reset Domains:** DEVICE, MASTER, COLD, SWITCH				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
HeadStorage_CERR	0	RW	1b	Head storage SRAM correctable error.
HeadStorage_UERR	1	RW	1b	Head storage SRAM uncorrectable error.
McVlanTable_CERR	2	RW	1b	MC_VLAN_TABLE SRAM correctable error.
McVlanTable_UERR	3	RW	1b	MC_VLAN_TABLE SRAM uncorrectable error.
PerPortCfg1_CERR	4	RW	1b	PER_PORT_CFG_1 SRAM correctable error.
PerPortCfg1_UERR	5	RW	1b	PER_PORT_CFG_1 SRAM uncorrectable error.
PerPortCfg2_CERR	6	RW	1b	MOD_PER_PORT_CFG_2 SRAM correctable error.
PerPortCfg2_UERR	7	RW	1b	PORT_CFG_2 SRAM uncorrectable error.
Vpri1Map_CERR	8	RW	1b	VPRI1_MAP SRAM correctable error.
Vpri1Map_UERR	9	RW	1b	VPRI1_MAP SRAM uncorrectable error.
Vpri2Map_CERR	10	RW	1b	VPRI2_MAP SRAM correctable error.
Vpri2Map_UERR	11	RW	1b	VPRI2_MAP SRAM uncorrectable error.
MirProfTable_CERR	12	RW	1b	MIRROR_PROFILE_TABLE SRAM correctable error.
MirProfTable_UERR	13	RW	1b	MIRROR_PROFILE_TABLE SRAM uncorrectable error.
VtagVid1Map_CERR	14	RW	1b	VTAG_VID1_MAP SRAM correctable error.
VtagVid1Map_UERR	15	RW	1b	VTAG_VID1_MAP SRAM uncorrectable error.
Vid2Map_CERR	16	RW	1b	VID2_MAP SRAM correctable error.
Vid2Map_UERR	17	RW	1b	VID2_MAP SRAM uncorrectable error.
Ctrl2DpFifo0_CERR	18	RW	1b	Ctrl2Dp FIFO 0 SRAM correctable error.
Ctrl2DpFifo0_UERR	19	RW	1b	Ctrl2Dp FIFO 0 SRAM uncorrectable error.
Ctrl2DpFifo1_CERR	20	RW	1b	Ctrl2Dp FIFO 1 SRAM correctable error.
Ctrl2DpFifo1_UERR	21	RW	1b	Ctrl2Dp FIFO 1 SRAM uncorrectable error.
Ctrl2DpFifo2_CERR	22	RW	1b	Ctrl2Dp FIFO 2 SRAM correctable error.
Ctrl2DpFifo2_UERR	23	RW	1b	Ctrl2Dp FIFO 2 SRAM uncorrectable error.
Ctrl2DpFifo3_CERR	24	RW	1b	Ctrl2Dp FIFO 3 SRAM correctable error.
Ctrl2DpFifo3_UERR	25	RW	1b	Ctrl2Dp FIFO 3 SRAM uncorrectable error.
StatsFrameCnt0_CERR	26	RW	1b	Stats frame count 0 SRAM correctable error.
StatsFrameCnt0_UERR	27	RW	1b	Stats frame count 0 SRAM uncorrectable error.

Address: **0xE80000 + 0x108**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
StatsFrameCnt1_CERR	28	RW	1b	Stats frame count 1 SRAM correctable error.
StatsFrameCnt1_UERR	29	RW	1b	Stats frame count 1 SRAM uncorrectable error.
StatsByteCnt0_CERR	30	RW	1b	Stats byte count 0 SRAM correctable error.
StatsByteCnt0_UERR	31	RW	1b	Stats byte count 0 SRAM uncorrectable error.
StatsByteCnt1_CERR	32	RW	1b	Stats byte count 1 SRAM correctable error.
StatsByteCnt1_UERR	33	RW	1b	Stats byte count 1 SRAM uncorrectable error.
Refcount_CERR	34	RW	1b	Refcount SRAM correctable error.
Refcount_UERR	35	RW	1b	Refcount SRAM uncorrectable error.
EschedTxInfoFifo_CERR	36	RW	1b	Esched TX Info FIFO correctable error.
EschedTxInfoFifo_UERR	37	RW	1b	Esched TX Info FIFO uncorrectable error.
CmTxInfoFifo_CERR	38	RW	1b	Cm TX Info FIFO correctable error.
CmTxInfoFifo_UERR	39	RW	1b	Cm TX Info FIFO uncorrectable error.
MgmtRdFifo_CERR	40	RW	1b	Mgmt read FIFO correctable error.
MgmtRdFifo_UERR	41	RW	1b	Mgmt read FIFO uncorrectable error.
Reserved	63:42	RSV	0x0	Reserved.

11.21.3.4 MOD_SRAM_BIST_OUT

Read only register to record the status of bist_done_pass and bist_done_fail of SRAMs.

Address: **0xE80000 + 0x10A**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
HeadStorage_PASS	0	RO	0b	Head storage SRAM bist_done_pass state.
HeadStorage_FAIL	1	RO	0b	Head storage SRAM bist_done_fail state.
McVlanTable_PASS	2	RO	0b	MC_VLAN_TABLE SRAM bist_done_pass state.
McVlanTable_FAIL	3	RO	0b	MC_VLAN_TABLE SRAM bist_done_fail state.
PerPortCfg1_PASS	4	RO	0b	PER_PORT_CFG_1 SRAM bist_done_pass state.
PerPortCfg1_FAIL	5	RO	0b	PER_PORT_CFG_1 SRAM bist_done_fail state.
PerPortCfg2_PASS	6	RO	0b	MOD_PER_PORT_CFG_2 SRAM bist_done_pass state.
PerPortCfg2_FAIL	7	RO	0b	PORT_CFG_2 SRAM bist_done_fail state.
Vpri1Map_PASS	8	RO	0b	VPRI1_MAP SRAM bist_done_pass state.
Vpri1Map_FAIL	9	RO	0b	VPRI1_MAP SRAM bist_done_fail state.
Vpri2Map_PASS	10	RO	0b	VPRI2_MAP SRAM bist_done_pass state.
Vpri2Map_FAIL	11	RO	0b	VPRI2_MAP SRAM bist_done_fail state.
MirProfTable_PASS	12	RO	0b	MIRROR_PROFILE_TABLE SRAM bist_done_pass state.
MirProfTable_FAIL	13	RO	0b	MIRROR_PROFILE_TABLE SRAM bist_done_fail state.
VtagVid1Map_PASS	14	RO	0b	VTAG_VID1_MAP SRAM bist_done_pass state.
VtagVid1Map_FAIL	15	RO	0b	VTAG_VID1_MAP SRAM bist_done_fail state.
Vid2Map_PASS	16	RO	0b	VID2_MAP SRAM bist_done_pass state.
Vid2Map_FAIL	17	RO	0b	VID2_MAP SRAM bist_done_fail state.

Address:	**0xE80000 + 0x10A**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Ctrl2DpFifo0_PASS	18	RO	0b	Ctrl2Dp FIFO 0 SRAM bist_done_pass state.
Ctrl2DpFifo0_FAIL	19	RO	0b	Ctrl2Dp FIFO 0 SRAM bist_done_fail state.
Ctrl2DpFifo1_PASS	20	RO	0b	Ctrl2Dp FIFO 1 SRAM bist_done_pass state.
Ctrl2DpFifo1_FAIL	21	RO	0b	Ctrl2Dp FIFO 1 SRAM bist_done_fail state.
Ctrl2DpFifo2_PASS	22	RO	0b	Ctrl2Dp FIFO 2 SRAM bist_done_pass state.
Ctrl2DpFifo2_FAIL	23	RO	0b	Ctrl2Dp FIFO 2 SRAM bist_done_fail state.
Ctrl2DpFifo3_PASS	24	RO	0b	Ctrl2Dp FIFO 3 SRAM bist_done_pass state.
Ctrl2DpFifo3_FAIL	25	RO	0b	Ctrl2Dp FIFO 3 SRAM bist_done_fail state.
StatsFrameCnt0_PASS	26	RO	0b	Stats frame count 0 SRAM bist_done_pass state.
StatsFrameCnt0_FAIL	27	RO	0b	Stats frame count 0 SRAM bist_done_fail state.
StatsFrameCnt1_PASS	28	RO	0b	Stats frame count 1 SRAM bist_done_pass state.
StatsFrameCnt1_FAIL	29	RO	0b	Stats frame count 1 SRAM bist_done_fail state.
StatsByteCnt0_PASS	30	RO	0b	Stats byte count 0 SRAM bist_done_pass state.
StatsByteCnt0_FAIL	31	RO	0b	Stats byte count 0 SRAM bist_done_fail state.
StatsByteCnt1_PASS	32	RO	0b	Stats byte count 1 SRAM bist_done_pass state.
StatsByteCnt1_FAIL	33	RO	0b	Stats byte count 1 SRAM bist_done_fail state.
Refcount_PASS	34	RO	0b	Refcount SRAM bist_done_pass state.
Refcount_FAIL	35	RO	0b	Refcount SRAM bist_done_fail state.
EschedTxInfoFifo_PASS	36	RO	0b	Esched TX Info FIFO bist_done_pass state.
EschedTxInfoFifo_FAIL	37	RO	0b	Esched TX Info FIFO bist_done_fail state.
CmTxInfoFifo_PASS	38	RO	0b	Cm TX Info FIFO bist_done_pass state.
CmTxInfoFifo_FAIL	39	RO	0b	Cm TX Info FIFO bist_done_fail state.
MgmtRdFifo_PASS	40	RO	0b	Mgmt read FIFO bist_done_pass state.
MgmtRdFifo_FAIL	41	RO	0b	Mgmt read FIFO bist_done_fail state.
Reserved	63:42	RSV	0x0	Reserved.

11.21.3.5 MOD_SRAM_ERROR_WRITE

Defines the control signal to force errors in SRAM write. Each 2 bits controls one SRAM.

00b = Do not cause erroneous write.
01b = Force correctable error.
10b = Force correctable error.
11b = Force uncorrectable error.

Address:	**0xE80000 + 0x10C**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
HeadStorage	1:0	RW	00b	Head storage SRAM error write control.
McVlanTable	3:2	RW	00b	MC_VLAN_TABLE SRAM error write control.
PerPortCfg1	5:4	RW	00b	PER_PORT_CFG_1 SRAM error write control.
PerPortCfg2	7:6	RW	00b	MOD_PER_PORT_CFG_2 SRAM error write control.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x10C** **64** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Vpri1Map	9:8	RW	00b	VPRI1_MAP SRAM error write control.
Vpri2Map	11:10	RW	00b	VPRI2_MAP SRAM error write control.
MirProfTable	13:12	RW	00b	MIRROR_PROFILE_TABLE SRAM error write control.
VtagVid1Map	15:14	RW	00b	VTAG_VID1_MAP SRAM error write control.
Vid2Map	17:16	RW	00b	VID2_MAP SRAM error write control.
Ctrl2DpFifo0	19:18	RW	00b	Ctrl2Dp FIFO 0 SRAM error write control.
Ctrl2DpFifo1	21:20	RW	00b	Ctrl2Dp FIFO 1 SRAM error write control.
Ctrl2DpFifo2	23:22	RW	00b	Ctrl2Dp FIFO 2 SRAM error write control.
Ctrl2DpFifo3	25:24	RW	00b	Ctrl2Dp FIFO 3 SRAM error write control.
StatsFrameCnt0	27:26	RW	00b	Stats frame count 0 SRAM error write control.
StatsFrameCnt1	29:28	RW	00b	Stats frame count 1 SRAM error write control.
StatsByteCnt0	31:30	RW	00b	Stats byte count 0 SRAM error write control.
StatsByteCnt1	33:32	RW	00b	Stats byte count 1 SRAM error write control.
Refcount	35:34	RW	00b	Refcount SRAM error write control.
EschedTxInfoFifo	37:36	RW	00b	Esched TX Info FIFO error write control.
CmTxInfoFifo	39:38	RW	00b	Cm TX Info FIFO error write control.
MgmtRdFifo	41:40	RW	00b	Mgmt read FIFO error write control.
Reserved	63:42	RSV	0x0	Reserved.

11.21.3.6 MOD_PAUSE_SMAC

Defines SMAC to use for PAUSE frame.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x10E** **64** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SMAC	47:0	RW	0x0	
Reserved	63:48	RSV	0x0	Reserved.

11.21.3.7 MOD_ROUTER_SMAC[0..15]

Defines which SMAC to use for routed frames.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x2*i + 0x120** **64** **DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SMAC	47:0	RW	0x0	
Reserved	63:48	RSV	0x0	Reserved.

11.21.3.8 MOD_MCAST_VLAN_TABLE[0..32767]

Table used during for packet replication on same port. The table indicates new VLAN and new DGLORT to use.

Note: Entry 32767 is not available and is not used.

Address: 0xE80000 + 0x2*i + 0x10000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
VID	11:0	RW	0x0	The new VID to use if update is requested.
DGLORT	27:12	RW	0x0	The new DGLORT to use if update is requested.
ReplaceVID	28	RW	0b	Indicates if the VID must be updated or not.
ReplaceDGLORT	29	RW	0b	Indicates if the DGLORT must be updated or not.
Reserved	31:30	RSV	00b	Reserved.

11.21.3.9 MOD_VLAN_TAG_VID1_MAP[0..4095]

Defines for each VLAN and port if the packet leaves tagged or untagged and VID1 for a given VID.

Address: 0xE80000 + 0x2*i + 0x20000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Tag	47:0	RW	0x0	Each bit defines the egress tagging option for the corresponding port. Combined with the *VlanTagging* bit from the MOD_PER_PORT_CFG_2 for full interpretation. Refer to Section 5.0 for details.
VID	59:48	RW	0x0	New VID1 associated with VID. If set to 0b, VID1 tag is not transmitted regardless of the *TAG* bit.
Reserved	63:60	RSV	0x0	Reserved.

11.21.3.10 MOD_VID2_MAP[0..4095]

Defines egress VID2 for a given VID/DGLORT/SGLORT. Index selection configurable per port.

Address: 0xE80000 + 0x2*i + 0x22000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
VID	11:0	RW	0x0	New VID2 associated with VID. If set to 0b, VID2 tag is not transmitted regardless of the *TAG* bit.
Reserved	31:12	RSV	0x0	Reserved.

11.21.3.11 MOD_MIRROR_PROFILE_TABLE[0..63]

Defines mirror profiles in MODIFY.

Address:	**0xE80000 + 0x2*i + 0x24000**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
GLORT	15:0	RW	0x0	New DGLORT to use for mirrored or logged copies.
TRUNC	16	RW	0b	Define whether mirrored copies should be truncated or not.
VID	28:17	RW	0x0	New VID to use for mirrored or logged copies.
VPRI	32:29	RW	0x0	New VPRI to use for mirrored or logged copies.
Reserved	63:33	RSV	0x0	Reserved.

11.21.3.12 MOD_PER_PORT_CFG_1[0..47]

Defines the port configuration. LOOPBACK_SUPPRESS: At the edge of the system, one must avoid retransmitting a frame over the same LAG it arrived on, even if the LAG is distributed across multiple edge chips. This is done by comparing the frame's source GLORT against the port's LAG GLORT, not counting the port-specific bits. The test performed in MCAST_POST is applied only to L3-replicated multicast frames. If the VID returned by MCAST_VLAN_TABLE matches either L2_VID, and the two glorts match, the replicated copy is not compared, and should be disabled on internal ports by setting Mask to 0b and GLORT to some non-zero value.

Address:	**0xE80000 + 0x2*i + 0x24080**			
Atomicity:	**64**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LoopbackSuppressGlort	15:0	RW	0xFFFF	Loopback suppress GLORT. LAG GLORT to which the port belongs. Comparison performed is MOD_DATA.W16A & Mask == GloRT.
LoopbackSuppressMask	31:16	RW	0x0	Canonicalization mask of loopback suppress. Port-specific bits should be set to 0b in the mask.
VID2MapIndex	33:32	RW	00b	Defines which index to use for indexing the MOD_VID2_MAP table. 00b = VID — Index by VID from fabric after MCAST_TABLE. 01b = SGLORT — Only lower 12 bits used. 10b = DGLORT — Index by DGLORT after MCAST_TABLE. Only lower 12 bits used. 11b = Reserved.
EnableVLANUpdate	34	RW	1b	Defines if egress or ingress VID is used to update transmit VIDs. 0b = Use ingress VID 1b = Use egress VID (after multicast update)
Reserved	63:35	RSV	0x0	Reserved.

11.21.3.13 MOD_PER_PORT_CFG_2[0..47]

Defines the port configuration.

Address:	**0xE80000 + 0x2*i + 0x24100**
Atomicity:	**64**
Reset Domains:	**SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
MirrorTruncationLen	5:0	RW	0x3F	Defines the maximum frame length if truncation is asserted. Length is specified in words (4 bytes), minimum is 16 words (64 bytes), maximum is 48 words (192 bytes). Setting to 63 disables truncation on this port even if frame handler request it. Setting to any other value outside of 16..48,63 is unpredictable. The length includes the FTAG (2 x 32-bit words).
EnablePcp1Update	6	RW	0b	Defines if the VLAN1 PCP is updated or not. 0b = The VLAN priority is not updated, even if the VID or DEI/CFI is updated in the tag. 1b = The 4-bit VPRI received from the frame handler is mapped to a new 3-bit Priority Code Point using MOD_VPRI1_MAP. This new code is used to update the priority (independent of whether the VID or DEI/CFI in the VLAN tag was updated). This bit is not used if the received VLAN 1 tag is removed or if a new VLAN tag is added.
EnablePcp2Update	7	RW	0b	Defines if the VLAN2 PCP is updated or not. 0b = The VLAN priority is not updated, even if the VID or DEI/CFI is updated in the tag. 1b = The 4-bit VPRI received from the frame handler is mapped to a new 3-bit Priority Code Point using MOD_VPRI2_MAP. This new code is used to update the priority (independent of whether the VID or DEI/CFI in the VLAN tag was updated). This bit is not used if the received VLAN 2 tag is removed or if a new VLAN tag is added.
EnableDei1Update	8	RW	0b	Defines if the VLAN CFI/DEI field is updated or not. 0b = The VLAN CFI/DEI is not updated when the VLAN ID is updated. 1b = The 4-bit VPRI received from the frame handler is mapped through the 1-bit code using MOD_VPRI1_MAP, and the new 1-bit code is used to update the CFI/DEI (independent of whether the VID or CFI in the VLAN tag was updated). This bit is not used if the received VLAN 1 tag is removed or if a new VLAN tag is added.
EnableDei2Update	9	RW	0b	Defines if the VLAN CFI/DEI field is updated or not. 0b = The VLAN CFI/DEI is not updated when the VLAN ID is updated. 1b = The 4-bit VPRI received from the frame handler is mapped through the 1-bit code using MOD_VPRI2_MAP, and the new 1-bit code is used to update the CFI/DEI (independent of whether the VID or CFI in the VLAN tag was updated). This bit is not used if the received VLAN 2 tag is removed or if a new VLAN tag is added.
VLAN1_EType	11:10	RW	01b	Defines the VLAN 1 Ethernet type to use.
VLAN2_EType	13:12	RW	01b	Defines the VLAN 2 Ethernet type to use.
EnableDMACRouting	14	RW	1b	When set, enables the replacement of Destination MAC addresses on outgoing routed (unicast and multicast) frames.
EnableSMACRouting	15	RW	1b	When set, enables the replacement of Source MAC addresses on outgoing routed (unicast and multicast) frames.
EnableTTLDecrement	16	RW	1b	When set, enables the decrement of the TTL field on outgoing routed frames.
EnableDSCPModification	17	RW	1b	When set, enables the modification of the DSCP field on outgoing IP frames.
FTAG	18	RW	0b	Defines if this port is FTAG or not.
VID2First	19	RW	0x0b0	Indicates if VID2 is sent before VID1 or after VID1.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x2*i + 0x24100** **64** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VlanTagging	22:20	RW	000b	Indicates how VLAN tagging is done. Combined with the tagging bit from the MOD_VLAN_TAG_VID1_MAP for full interpretation. Refer to Section 5.0 for details. Only values 0..4 are defined, values of 5..7 are undefined.
TxPausePriEnVec	30:23	RW	0x0	Defines the value used for _priority_enable_vector_ in class based pause frame
TxPauseType	31	RW	0b	Indicates the pause type. 0b = Normal pause 1b = Class-based
TxPauseValue	47:32	RW	0xFFFF	Pause Value used for TX pause frames in terms of the number of 512-bit times the link partner needs to Pause. The same value is used for all class in class based flow control.
MinFrameSize	48	RW	0b	Indicates the minimum frame size of egress port. 0b = Indicates no minimum frame size constraints. 1b = Indicates 64 bytes.
Reserved	63:49	RSV	0x0	Reserved.

11.21.3.14 MOD_VPRI1_MAP[0..47]

Defines VPRI1 priority map per port.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x2*i + 0x24180** **64** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VPRI[0..15]	63:0	RW	0000b	(16 x 4-bits) Maps internal VPRI to transmit VPRI1 (PCP, DEI/CFI). Bits 3..1 is PCP and bit 0 is DEI/CFI.

11.21.3.15 MOD_VPRI2_MAP[0..47]

Defines VPRI2 priority map per port.

Address: **Atomicity:** **Reset Domains:**	**0xE80000 + 0x2*i + 0x24200** **64** **SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VPRI[0..15]	63:0	RW	0000b	(16 x 4-bits) Maps internal VPRI to transmit VPRI2 (PCP, DEI/CFI). Bits 3..1 is PCP and bit 0 is DEI/CFI.

11.21.3.16 MOD_VLAN_ETYPE[0..3]

Defines 4 unique VLAN tag type. Which one is used depends on port configuration.

Address: 0xE80000 + 0x2*i + 0x24280
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
TagType	15:0	RW	0x8100	Value to be used as ETYPE.
Reserved	31:16	RSV	0x0	Reserved.

11.21.3.17 MOD_STATS_CFG[0..47]

Defines per port configuration of MODIFY stats.

Address: 0xE80000 + 0x2*i + 0x24300
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
EnableGroup7	0	RW	0b	Enables group 7 counters.
EnableGroup8	1	RW	0b	Enables group 8 counters.
Reserved	31:2	RSV	0x0	Reserved.

11.21.3.18 MOD_STATS_BANK_FRAME[0..1][0..767]

Defines MODIFY stats frame counters. The first index identifies the group, the second index is the counter for that group.

Group 7: MOD_STATS_BANK_FRAME[0,port*16+type], type defined in TX_STATS_BANK_7

Group 8: MOD_STATS_BANK_FRAME[1,port*16+type], type defined in TX_STATS_BANK_8

Address: 0xE80000 + 0x800*j + 0x2*i + 0x25000
Atomicity: 64
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
FrameCounter	47:0	vRW	0x0	Frame counter.
Reserved	63:48	RSV	0x0	Reserved.

11.21.3.19 MOD_STATS_BANK_BYTE[0..1][0..767]

Defines MODIFY stats byte counters. The first index identifies the group, the second index is the counter for that group.

Group 7: MOD_STATS_BANK_BYTE[0,port*16+type], type defined in TX_STATS_BANK_7

Group 8: MOD_STATS_BANK_BYTE[1,port*16+type], type defined in TX_STATS_BANK_8

Address:	**0xE80000 + 0x800*j + 0x2*i + 0x26000**
Atomicity:	**64**
Reset Domains:	**SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
ByteCounter	55:0	vRW	0x0	Byte counter.
Reserved	63:56	RSV	0x0	Reserved.

11.22 RX_STATS Registers Description

11.22.1 RX_STATS Map

Table 11-33 Ingress Statistics Counters (reserved 256 KW) Map

Name	Address (Base = 0xE00000)	Atomicity	Brief Description
RX_STATS_BANK[0..5][0..767]	Base + 0x1000*j + 0x4*i + 0x0	64	Statistics counter by bank and entry.
RX_STATS_CFG[0..47]	Base + 0x1*i + 0x10000	32	Per-port statistics configuration.

11.22.2 RX_STATS Enumerated Data Types

Table 11-34 RX_STATS Enumerated Data Types

Name	Width	Description
RX_STATS_BANK_1	4	0000b = NonIP_L2Ucast — Non-IP frame with unicast DMAC. 0001b = NonIP_L2Mcast — Non-IP frame with multicast DMAC. 0010b = NonIP_L2Bcast — Non-IP frame with broadcast DMAC. 0011b = IPv4_L2Ucast — IPv4 frame with unicast DMAC. 0100b = IPv4_L2Mcast — IPv4 frame with multicast DMAC. 0101b = IPv4_L2Bcast — IPv4 frame with broadcast DMAC. 0110b = IPv6_L2Ucast — IPv6 frame with unicast DMAC. 0111b = IPv6_L2Mcast — IPv6 frame with multicast DMAC. 1000b = IPv6_L2Bcast — IPv6 frame with broadcast DMAC. 1001b = IEEE802_3_pause — IEEE 802.3 pause frame. 1010b = Class_based_pause — Class based pause frame. 1011b = Framing_Err — Frame received with framing or internal error. 1100b = FCS_Err — Frame received with FCS error. All other values are reserved.
RX_STATS_BANK_2	4	0000b = Len_lt_64 — length < 64 0001b = Len_eq_64 — length = 64 0010b = Len_65_127 — 64 < length < 128 0011b = Len_128_255 — 128 ≤ length ≤ 255 0100b = Len_256_511 — 256 ≤ length ≤ 511 0101b = Len_512_1023 — 512 ≤ length ≤ 1023 0110b = Len_1024_1522 — 1024 ≤ length ≤ 1522 0111b = Len_1523_2047 — 1523 ≤ length ≤ 2047 1000b = Len_2048_4095 — 2048 ≤ length ≤ 4095 1001b = Len_4096_8191 — 4096 ≤ length ≤ 8191 1010b = Len_8192_10239 — 8192 ≤ length ≤ 10239 1011b = Len_ge_10240 — length ≥ 10240 All other values are reserved.

Table 11-34 RX_STATS Enumerated Data Types [Continued]

Name	Width	Description
RX_STATS_BANK_3	4	0000b = Pri_0 — Priority = 0 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0001b = Pri_1 — Priority = 1 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0010b = Pri_2 — Priority = 2 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0011b = Pri_3 — Priority = 3 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0100b = Pri_4 — Priority = 4 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0101b = Pri_5 — Priority = 5 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0110b = Pri_6 — Priority = 6 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 0111b = Pri_7 — Priority = 7 (TC or SWITCH_PRI as selected by RX_STATS_CFG) 1000b = Pri_8 — Priority = 8 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1001b = Pri_9 — Priority = 9 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1010b = Pri_10 — Priority = 10 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1011b = Pri_11 — Priority = 11 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1100b = Pri_12 — Priority = 12 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1101b = Pri_13 — Priority = 13 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1110b = Pri_14 — Priority = 14 (0 or SWITCH_PRI as selected by RX_STATS_CFG) 1111b = Pri_15 — Priority = 15 (0 or SWITCH_PRI as selected by RX_STATS_CFG)
RX_STATS_BANK_4	4	0000b = FIDForwarded — Forwarded normally with hit in MA_TABLE. 0001b = FloodForwarded — Flooded due to miss in MA_TABLE. 0010b = SpeciallyHandled — FTYPE=0x2 (special delivery). 0011b = ParserErrorDrop — Dropped due to parser errors. 0100b = EccErrorDrop — Dropped due to uncorrectable ECC errors. 0101b = Trapped — Trapped to CPU. 0110b = PauseDrops — Dropped due to SYS_CFG_1.*dropPause* or *dropMacCtrlEthertype*. 0111b = STPDrops — Fully dropped due to STP violations, partial drops not counted. 1000b = SecurityViolations — See GEN_MASK. 1001b = VlanTagDrops — Dropped due to VLAN tag. 1010b = VlanIngressDrops — Ingress VLAN boundary violation. 1011b = VlanEgressDrops — Egress VLAN boundary violation. 1100b = GlortMissDrops — Dropped due to miss in GLORT_TABLE. 1101b = FFUDrops — Dropped due to FFU action. 1110b = TriggerDrops — Dropped due to Trigger action. 1111b = Reserved
RX_STATS_BANK_5	4	0000b = PolicerDrops — Dropped due to Policer action. 0001b = TTLDrops — Routed frame dropped due to TTL ≤ 1. 0010b = CMPrivDrops — Dropped due to CM_GLOBAL_WM. 0011b = CMSMP0Drops — Dropped due to insufficient memory in SMP 0. 0100b = CMSMP1Drops — Dropped due to insufficient memory in SMP 1. 0101b = CMRxHog0Drops — Dropped due to hog watermark in SMP 0. 0110b = CMRxHog1Drops — Dropped due to hog watermark in SMP 1. 0111b = CMTxHog0Drops — Dropped due to hog/softdrop watermark in SMP 0. 1000b = CMTxHog1Drops — Dropped due to hog/softdrop watermark in SMP 1. 1001b = Reserved — Unused and reserved. 1010b = TriggerRedirects — Redirected by triggers. 1011b = FloodControlDrops — Dropped due to DLF flood control. 1100b = GlortForwarded — Forwarded due to FFU/ARP DGLORT rules. 1101b = LoopbackSuppDrops — Dropped due to loopback suppress. 1110b = OtherDrops — Dropped frame due to one of the following: (a) Reserved multicast DMAC (01-80-C2-00-00-XX) (b) Invalid SMAC. (c) Null GLORT destmask. 1111b = ReservedUnused and reserved.

11.22.3 RX_STATS Registers

11.22.3.1 RX_STATS_BANK[0..5][0..767]

Each entry corresponds to a port x statistics class. There are 6 banks in total. Refer to specification chapter for more details.

The first index identifies the group:

Group 1:	RX_STATS_BANK[0,port*16+type], type defined in RX_STATS_BANK_1
Group 2:	RX_STATS_BANK[1,port*16+type], type defined in RX_STATS_BANK_2
Group 3:	RX_STATS_BANK[2,port*16+type], type defined in RX_STATS_BANK_3
Group 4:	RX_STATS_BANK[3,port*16+type], type defined in RX_STATS_BANK_4
Group 5:	RX_STATS_BANK[4,port*16+type], type defined in RX_STATS_BANK_5
Group 6:	RX_STATS_BANK[5,type*64+vcnt], type is 0=UCAST, 1=MCAST, 2=BCAST

Address: **0xE00000 + 0x1000*j + 0x4*i + 0x0**
Atomicity: **64**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
Frames	47:0	vRW	0x0	Number of frames received.
Reserved	63:48	RSV	0x0	Reserved.
Bytes	119:64	vRW	0x0	Number of bytes received.
Reserved	127:120	RSV	0x0	Reserved.

11.22.3.2 RX_STATS_CFG[0..47]

Per-port statistics configuration.

Address: **0xE00000 + 0x1*i + 0x10000**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
PerFrameAdjustment	7:0	RW	0x0	Number of bytes subtracted from the received packet length. Intended for the removal of the FTAG bytes from the packet length. Legal values are 0 and 8. Setting to unsupported values larger than *minFrameSize* will corrupt byte statistics. Does not affect length binning.
EnableMask	13:8	RW	0x0	Enable mask to allow counting in each of the 6 banks.
PrioritySelect	14	RW	0b	Stats bank 3. Select between: 0b = TC 1b = SWITCH_PRI
Reserved	31:15	RSV	0x0	Reserved.

11.23 SCHED Registers Description

11.23.1 SCHED Map

Table 11-35 Scheduler Registers Map

Name	Address (Base = 0xF00000)	Atomicity	Brief Description
SCHED_RX_SCHEDULE[0..1023]	Base + 0x1*i + 0x20000	32	Receiver (rx) schedule.
SCHED_TX_SCHEDULE[0..1023]	Base + 0x1*i + 0x20400	32	Transmit (tx) schedule.
SCHED_SCHEDULE_CTRL	Base + 0x20800	32	Schedule ctrl (both rx and tx).
SCHED_CONFIG_SRAM_CTRL	Base + 0x20801	32	sched_config memory controller.
SCHED_SSCHED_RX_PERPORT[0..47]	Base + 0x1*i + 0x20840	32	Initialize memory pointers.
SCHED_SSCHED_SRAM_CTRL	Base + 0x30050	32	Memory controller.
SCHED_ESCHED_CFG_1[0..47]	Base + 0x1*i + 0x30080	32	Scheduling group configuration.
SCHED_ESCHED_CFG_2[0..47]	Base + 0x1*i + 0x300C0	32	Traffic-class behaviors.
SCHED_ESCHED_CFG_3[0..47]	Base + 0x1*i + 0x30100	32	Traffic class inner priority.
SCHED_MGMT_TIMER_ESCHED	Base + 0x30240	32	Minimum MGMT bandwidth.
SCHED_ESCHED_SRAM_CTRL	Base + 0x30242	64	Memory controller.
SCHED_FREELIST_INIT	Base + 0x30244	32	Initialize segment pool.
SCHED_FREELIST_SRAM_CTRL	Base + 0x30246	32	Memory controller.
SCHED_MONITOR_DRR_Q_PERQ[0..383]	Base + 0x1*i + 0x30400	32	Scheduling group configuration.
SCHED_MONITOR_DRR_CFG_PERPORT[0..47]	Base + 0x1*i + 0x30600	32	Scheduling group configuration.
SCHED_MGMT_TIMER_MONITOR	Base + 0x1*i + 0x30A40	32	Minimum MGMT bandwidth.
SCHED_MONITOR_SRAM_CTRL	Base + 0x30A41	32	Memory controller.
SCHED_MCAST_LEN_TABLE[0..16383]	Base + 0x1*i + 0x34000	32	VLAN configuration.
SCHED_MCAST_DEST_TABLE[0..4095]	Base + 0x2*i + 0x38000	64	VLAN configuration.
SCHED_MCAST_LIMITED_SKEW_MULTICAST	Base + 0x3A000	32	Multicast configuration.
SCHED_RXQ_STORAGE_POINTERS[0..7]	Base + 0x2*i + 0x60400	64	Linked-list state in RXQ_MCAST.
SCHED_RXQ_FREELIST_INIT	Base + 0x60410	32	Initialize RXQ_MCAST address pool.
SCHED_MGMT_TIMER_RXQ	Base + 0x60412	32	Minimum MGMT bandwidth.
SCHED_MGMT_TIMER_LG	Base + 0x60413	32	Minimum MGMT bandwidth.
SCHED_RXQ_SRAM_CTRL	Base + 0x60414	64	Memory controller.
SCHED_TXQ_TAIL0_PERQ[0..383]	Base + 0x1*i + 0x60600	32	Txq initialization and debug.
SCHED_TXQ_TAIL1_PERQ[0..383]	Base + 0x1*i + 0x60800	32	Txq initialization and debug.
SCHED_TXQ_HEAD_PERQ[0..383]	Base + 0x1*i + 0x60A00	32	Txq initialization and debug.
SCHED_TXQ_FREELIST_INIT	Base + 0x62000	32	TX freelist pool.
SCHED_TXQ_SRAM_CTRL	Base + 0x62002	64	Memory controller.
SCHED_FIFO_SRAM_CTRL	Base + 0x62004	64	Memory controller.

Table 11-35 Scheduler Registers Map [Continued]

Name	Address (Base = 0xF00000)	Atomicity	Brief Description
SCHED_IP	Base + 0x62006	32	Interrupt pending.
SCHED_IM	Base + 0x62007	32	Interrupt mask.

11.23.2 SCHED Registers

11.23.2.1 SCHED_RX_SCHEDULE[0..1023]

Consists of two 512-entry pages.

If written while traffic is flowing, must use activation procedure via SCHED_SCHEDULE_CTRL.

Address: 0xF00000 + 0x1*i + 0x20000
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
PhysPort	7:0	RW	0x0	Physical fabric port (index in ingress crossbar). Port-PhysPort mapping must remain consistent throughout the schedule and between switch resets.
Port	13:8	RW	0x0	Logical fabric port (index in FC).
Quad	14	RW	0b	0b = Single-lane operation for 25 Gb/s or less. 1b = Multi-lane operation up to 100 Gb/s.
Color	15	RW	0b	CYA. In case we need color striping.
Idle	16	RW	0b	Set if this entry is an idle cycle, available for MGMT or sweep operations.
Reserved	31:17	RSV	0x0	Reserved.

11.23.2.2 SCHED_TX_SCHEDULE[0..1023]

Consists of two 512-entry pages.

Address: 0xF00000 + 0x1*i + 0x20400
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
PhysPort	7:0	RW	0x0	Physical fabric port (index in ingress crossbar). Port-PhysPort mapping must remain consistent throughout the schedule and between switch resets. If written while traffic is flowing, must use activation procedure via SCHED_SCHEDULE_CTRL.
Port	13:8	RW	0x0	Logical fabric port (index in FC).
Quad	14	RW	0b	0b = Single-lane operation for 25 Gb/s or less. 1b = Multi-lane operation up to 100 Gb/s.
Color	15	RW	0b	CYA. In case we need color striping.
Idle	16	RW	0b	Set if this entry is an idle cycle, available for MGMT or sweep operations.
Reserved	31:17	RSV	0x0	Reserved.

11.23.2.3 SCHED_SCHEDULE_CTRL

Address: 0xF00000 + 0x20800
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
RxEnable	0	RW	0b	Indicate if RX scheduler is active or not. If set to 0b, then no ingress ports are serviced.
RxPage	1	RW	0b	Defines which RX schedule page is active (0b or 1b).
RxMaxIndex	10:2	RW	0x0	Highest index used for the RX table.
TxEnable	11	RW	0b	Indicate if TX scheduler is active or not. If set to 0b, then no egress ports are serviced.
TxPage	12	RW	0b	Defines which TX schedule page is active (0b or 1b).
TxMaxIndex	21:13	RW	0x0	Highest index used for the TX table.
Reserved	31:22	RSV	0x0	Reserved.

11.23.2.4 SCHED_CONFIG_SRAM_CTRL

This register controls the sched_config memory cores: rx=0, tx=1

Address: 0xF00000 + 0x20801
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..1]	3:0	RW	00b	(2 x 2-bits)
CErr[0..1]	5:4	vCW1	0b	(2 x 1-bit)
UErr[0..1]	7:6	vCW1	0b	(2 x 1-bit)
BistDonePass[0..1]	9:8	vRO	0b	(2 x 1-bit)
BistDoneFail[0..1]	11:10	vRO	0b	(2 x 1-bit)
Reserved	31:12	RSV	0x0	Reserved.

11.23.2.5 SCHED_SSCHED_RX_PERPORT[0..47]

May not be written while traffic is flowing.

Address: 0xF00000 + 0x1*i + 0x20840
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
Next	15:0	vRW	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.6 SCHED_SSCHED_SRAM_CTRL

This register controls the ssched memory cores: rxpp=0, txpp=1, link=2, modpf=3

Address:	**0xF00000 + 0x30050**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ErrWrite[0..3]	7:0	RW	00b	(4 x 2-bits)
CErr[0..3]	11:8	vCW1	0b	(4 x 1-bit)
UErr[0..3]	15:12	vCW1	0b	(4 x 1-bit)
BistDonePass[0..3]	19:16	vRO	0b	(4 x 1-bit)
BistDoneFail[0..3]	23:20	vRO	0b	(4 x 1-bit)
Reserved	31:24	RSV	0x0	Reserved.

11.23.2.7 SCHED_ESCHED_CFG_1[0..47]

This register defines the relative egress scheduling-priority of the scheduler groups and which traffic-classes belong to which group.

If written while traffic is flowing, temporary violation of DRR limits may occur.

Address:	**0xF00000 + 0x1*i + 0x30080**			
Atomicity:	**32**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PrioritySetBoundary	7:0	RW	0xFF	Must be set to the bitwise OR of SCHED_ESCHED_CFG_2.*StrictPriority* and SCHED_ESCHED_CFG_2.*StrictPriority* (for the same port) shifted one position to the right. The behavior of the scheduler is undefined if this constraint is violated.
TcGroupBoundary	15:8	RW	0xFF	Defines groups of classes that use a common deficit counter for the purpose of delay-deficit round-robin scheduling. Classes within a scheduling group have strict priority relative to each other. There is 1 bit per class. 0b = The class belongs to the same group as the class to the left (the class with higher index). 1b = The class is at the head of a new group. Must be set the same as SCHED_MONITOR_DRR_CFG_PERPORT.*groupBoundary*; the behavior of the scheduler is undefined if this constraint is violated.
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.8 SCHED_ESCHED_CFG_2[0..47]

This register defines whether a Traffic Class is subject to WRR credit calculations.

Address: **0xF00000 + 0x1*i + 0x300C0**
Atomicity: **32**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
StrictPriority	7:0	RW	0xFF	One bit per traffic class, true if strict-low or strict-high. The strict-DRR classes must be between the strict-low and strict-high classes (although any one of the three groups could be empty). SCHED_ESCHED_CFG_1.*PrioritySetBoundary* for the same port must be set to the bitwise OR of this register and this register shifted right by one position.The behavior of the scheduler is undefined if these constraints are violated.
TcEnable	15:8	RW	0xFF	Sets whether scheduling is enabled on a given traffic class. Writing 0x0000 to the field pauses the port.
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.9 SCHED_ESCHED_CFG_3[0..47]

Defines the relative strict “inner” prioritization of the traffic classes. By default, higher-numbered TCs have higher priority than lower-numbered TCs. Setting TcInnerPriority to a non-zero value changes this behavior.

Address: **0xF00000 + 0x1*i + 0x30100**
Atomicity: **32**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
TcInnerPriority	7:0	RW	0x0	Specifies one bit of inner priority per TC. If TC1 and TC2 belong to the same scheduling group and *TcInnerPriority*[TC1] > *TcInnerPriority*[TC2], TC1 is serviced with strict high priority relative to TC2. Among all TCs of equal inner priority in a scheduling group, higher-numbered TCs are still strictly preferred over lower-numbered TCs.
Reserved	31:8	RSV	0x0	Reserved.

11.23.2.10 SCHED_MGMT_TIMER_ESCHED

Ensures a minimum number of idle cycles for MGMT access to the SRAMs in ESCHED.

Address: **0xF00000 + 0x30240**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Timer	15:0	vRO	0x0	Resets to 0x0 on idle cycle and increments every non-idle cycle until reaching *TimerMax*. When *TimerMax* is reached, a management cycle is allowed and any other activity must wait. *Timer* then resets to 0.
TimerMax	31:16	RW	0x0	Max count value of *Timer*. Set to 0x0 to disable.

11.23.2.11 SCHED_ESCHED_SRAM_CTRL

This register controls the esched memory cores: cfg1=0, cfg2=1, cfg3=2, dc=3, ub=4, ptr=5, pause=6

Address: 0xF00000 + 0x30242
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits)
CErr[0..6]	20:14	vCW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	vCW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	vRO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	vRO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.23.2.12 SCHED_FREELIST_INIT

The available pool of main segments is stored in a large FIFO. The FIFO is initialized by repeatedly writing to the SCHED_FREELIST_ADDR register. Writing to this address.

This list is initialized once after chip reset and normally loaded automatically by the boot state machine. This register is not intended to be accessed directly by software.

May not be written while traffic is flowing.

Address: 0xF00000 + 0x30244
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Address	15:0	WO	0x0	Fill segment pool.
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.13 SCHED_FREELIST_SRAM_CTRL

Address: 0xF00000 + 0x30246
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	
CErr	2	vCW1	0b	
UErr	3	vCW1	0b	
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.23.2.14 SCHED_MONITOR_DRR_Q_PERQ[0..383]

DRR quantum Q per queue. The proportions of egress bandwidth committed for each QID (port, traffic-class) depend on the ratios of their Q values. e.g., given two traffic-classes, one with a committed rate of 90% and one with a committed rate of 10%, the ratio of their respective Q values should be 9:1.

Indexed by QID = PORT*8 + TC.

Address: 0xF00000 + 0x1*i + 0x30400
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
q	23:0	RW	0x0	Sets the number of credits (in bytes) assigned to each queue. Must be set to at least 192 bytes. To ensure reliable adherence to the bandwidth weightings, Q should be set no lower than the 2*MTU of the frames in a given queue.
Reserved	31:24	RSV	0x0	Reserved.

11.23.2.15 SCHED_MONITOR_DRR_CFG_PERPORT[0..47]

This register defines the relative egress scheduling-priority of the traffic-classes.

If written while traffic is flowing, temporary violation of DRR limits may occur.

Address: 0xF00000 + 0x1*i + 0x30600
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
zeroLength	7:0	RW	0xFF	Each bit corresponds to a traffic-class. Setting the bit to 1b causes all associated segments to be counted as having zero length. No IFG charge is applied either. Must be set for strict-low or strict-high priorities; the behavior of the scheduler is undefined if this constraint is violated.
groupBoundary	15:8	RW	0xFF	Defines groups of classes that use a common deficit counter for the purpose of delay-deficit round-robin scheduling. Classes within a scheduling group have strict priority relative to each other. There is one bit per class. 0b = The class belongs to the same group as the class to the left (the class with higher index). 1b = The class is at the head of a new group. Must be set the same as SCHED_ESCHED_CFG_1.*TcGroupBoundary*; the behavior of the scheduler is undefined if this constraint is violated.
ifgPenalty	23:16	RW	0x0	The additional charge in bytes to apply per-frame for the purposes scheduling DRR.
Reserved	31:24	RSV	0x0	Reserved.

11.23.2.16 SCHED_MGMT_TIMER_MONITOR

Ensures a minimum number of idle cycles for MGMT access to the SRAMs in MONITOR.

Address: 0xF00000 + 0x30A40
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Timer	15:0	vRO	0x0	Resets to 0b on idle cycle and increments every non-idle cycle until reaching *TimerMax*. When *TimerMax* is reached, a management cycle is allowed and any other SRAM activity must wait. *Timer* then resets to 0b.
TimerMax	31:16	RW	0x0	Max count value of *Timer*. Set to 0b to disable.

11.23.2.17 SCHED_MONITOR_SRAM_CTRL

This register controls the monitor memory cores: drrq=0, drrdc=1, drrcfg=2, pep=3

Address: 0xF00000 + 0x30A41
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..3]	7:0	RW	00b	(4 x 2-bits)
CErr[0..3]	11:8	vCW1	0b	(4 x 1-bit)
UErr[0..3]	15:12	vCW1	0b	(4 x 1-bit)
BistDonePass[0..3]	19:16	vRO	0b	(4 x 1-bit)
BistDoneFail[0..3]	23:20	vRO	0b	(4 x 1-bit)
Reserved	31:24	RSV	0x0	Reserved.

11.23.2.18 SCHED_MCAST_LEN_TABLE[0..16383]

Defines the location and length for each VLAN list in MOD_MCAST_VLAN_TABLE. To modify safely while traffic is flowing, coordinate with MOD_MCAST_VLAN_TABLE.

Address: 0xF00000 + 0x1*i + 0x34000
Atomicity: 32
Reset Domains: SWITCH_MEM

Field Name	Bit(s)	Type	Default	Description
L3_McastIdx	14:0	RW	0x0	Defines the base location of the VLAN list in MOD_MCAST_VLAN_TABLE.
L3_Repcnt	26:15	RW	0x0	Defines the number of multicast copies to make. The number of copies made is one more than the value of this field, so that 0 indicates that a single frame is made. Legal values are 0 through 4093, so that 1 to 4094 copies are made.
Reserved	31:27	RSV	0x0	Reserved.

11.23.2.19 SCHED_MCAST_DEST_TABLE[0..4095]

Indexed by GLORT_DEST_TABLE.*IP_MulticastIndex*. To modify safely while traffic is flowing, coordinate with SCHED_MCAST_LEN_TABLE.

Address:	**0xF00000 + 0x2*i + 0x38000**
Atomicity:	**64**
Reset Domains:	**SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
PortMask	47:0	RW	0x0	Defines which ports have a list of destination VLANs specified in SCHED_MCAST_LEN_TABLE. If a port does not have a list, and the destination mask from the Frame Processing Pipeline is 1 for that port, a packet is sent to that port without a VLAN replication attached to it.
LenTableIdx	61:48	RW	0x0	Defines the base location of the list of VLAN lists in the SCHED_MCAST_LEN_TABLE.
Reserved	63:62	RSV	00b	Reserved.

11.23.2.20 SCHED_MCAST_LIMITED_SKEW_MULTICAST

Address:	**0xF00000 + 0x3A000**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
McastTcSkew	7:0	RW	0x0	One bit per traffic class. If set to 1, all copies of frames in this traffic class queue are made before moving to the next tc.
Reserved	31:8	RSV	0x0	Reserved.

11.23.2.21 SCHED_RXQ_STORAGE_POINTERS[0..7]

May not be written while traffic is flowing.

Address:	**0xF00000 + 0x2*i + 0x60400**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
HeadPage	9:0	vRW	0x0	Defines the page field of head pointer. Each page includes 24 pointers.
TailPage	19:10	vRW	0x0	Defines the page field of tail pointer. Each page includes 24 pointers.
HeadIdx	24:20	vRW	0x0	Defines the index field of head pointer in a page. Valid value is from 0 to 23.
TailIdx	29:25	vRW	0x0	Defines the index field of tail pointer in a page. Valid value is from 0 to 23.
NextPage	39:30	vRW	0x0	Defines the next page for head pointer.
Reserved	63:40	RSV	0x0	Reserved.

11.23.2.22 SCHED_RXQ_FREELIST_INIT

The available pool of addresses for RXQ is stored in a FIFO. The FIFO is initialized by repeatedly writing to this address.

This list is initialized once after chip reset and normally loaded automatically by the boot state machine. This register is not intended to be accessed directly by software.

May not be written while traffic is flowing.

Address:	**0xF00000 + 0x60410**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Address	9:0	WO	0x0	Fill RXQ freelist pool.
Reserved	31:10	RSV	0x0	Reserved.

11.23.2.23 SCHED_MGMT_TIMER_RXQ

Ensures a minimum number of idle cycles for MGMT access to the SRAMs in RXQ_MCAST.RXQ_STORAGE.

Address:	**0xF00000 + 0x60412**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Timer	15:0	vRO	0x0	Resets to 0b on idle cycle and increments every non-idle cycle until reaching *TimerMax*. When *TimerMax* is reached, a management cycle is allowed and any other activity must wait. *Timer* then resets to 0b.
TimerMax	31:16	RW	0x0	Max count value of *Timer*. Set to 0b to disable.

11.23.2.24 SCHED_MGMT_TIMER_LG

Ensures a minimum number of idle cycles for MGMT access to the SRAMs in RXQ_MCAST.LOOKINGGLASS as well as some SRAMs in TXQ.

Address:	**0xF00000 + 0x60413**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Timer	15:0	vRO	0x0	Resets to 0b on idle cycle and increments every non-idle cycle until reaching *TimerMax*. When *TimerMax* is reached, a management cycle is allowed, and any SRAM activity must wait. *Timer* then resets to 0b.
TimerMax	31:16	RW	0x0	Max count value of *Timer*. Set to 0b to disable.

11.23.2.25 SCHED_RXQ_SRAM_CTRL

This register controls the rxq_mcast memory cores: data_lo=0, data_hi=1, link=2, freelist=3, dst_table=4, len_table=5, output_fifo=6

Address: **0xF00000 + 0x60414**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits)
CErr[0..6]	20:14	vCW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	vCW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	vRO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	vRO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.23.2.26 SCHED_TXQ_TAIL0_PERQ[0..383]

Indexed by QID = PORT*8 + TC.

Address: **0xF00000 + 0x1*i + 0x60600**
Atomicity: **32**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
Tail	15:0	vWO (RW)	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.27 SCHED_TXQ_TAIL1_PERQ[0..383]

Indexed by QID = PORT*8 + TC.

Address: **0xF00000 + 0x1*i + 0x60800**
Atomicity: **32**
Reset Domains: **SWITCH_MEM**

Field Name	Bit(s)	Type	Default	Description
Tail	15:0	vWO (RW)	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.28 SCHED_TXQ_HEAD_PERQ[0..383]

Indexed by QID = PORT*8 + TC.

Address:	**0xF00000 + 0x1*i + 0x60A00**			
Atomicity:	**32**			
Reset Domains:	**SWITCH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Head	15:0	vWO (RW)	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.29 SCHED_TXQ_FREELIST_INIT

The available pool of RXTIME and PLINK memory is stored in a large FIFO. The FIFO is initialized by repeatedly writing to this address. Addresses between 0 and 24 K are written during initialization, leaving out bad memory locations in PLINK and RXTIME.

This list is initialized once after chip reset and normally loaded automatically by the boot state machine. This register is not intended to be accessed directly by software.

Address:	**0xF00000 + 0x62000**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Address	15:0	WO	0x0	Fill TXQ PLINK and RXTIME pool.
Reserved	31:16	RSV	0x0	Reserved.

11.23.2.30 SCHED_TXQ_SRAM_CTRL

This register controls the txq memory cores: head=0, plink=1, rep_perq=2, tail0_perq=3, tail1_perq=4, freelist=5, rxtime=6

Address:	**0xF00000 + 0x62002**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits)
CErr[0..6]	20:14	vCW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	vCW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	vRO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	vRO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.23.2.31 SCHED_FIFO_SRAM_CTRL

This register controls the memory cores used to implement the top-level FIFOs: schedule_fc=0, rx_port=1, tx_port=2, tail_info=3, array_ctrl=4, modify_ctrl=5, egress_ctrl=6

Address: **0xF00000 + 0x62004**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits)
CErr[0..6]	20:14	vCW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	vCW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	vRO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	vRO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.23.2.32 SCHED_IP

Interrupt pending register for Scheduler.

Address: **0xF00000 + 0x62006**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SchedConfigSramErr	1:0	vCW1	00b	
FreelistSramErr	3:2	vCW1	00b	
SschedSramErr	5:4	vCW1	00b	
MonitorSramErr	7:6	vCW1	00b	
EschedSramErr	9:8	vCW1	00b	
TxqSramErr	11:10	vCW1	00b	
RxqMcastSramErr	13:12	vCW1	00b	
FifoSramErr	15:14	vCW1	00b	
RxPortSpacing	16	vCW1	0b	
TxPortSpacing	17	vCW1	0b	
RxPortRange	18	vCW1	0b	
TxPortRange	19	vCW1	0b	
RxPerformance	20	vCW1	0b	
TxPerformance	21	vCW1	0b	
OutOfMemory	22	vCW1	0b	
RxFlowcPortFifo	23	vCW1	0b	
RxFlowcScheduleFc	24	vCW1	0b	
RxFlowcScheduleRx	25	vCW1	0b	
RxFlowcArrayCtrl	26	vCW1	0b	
RxFlowcTailInfo	27	vCW1	0b	
TxFlowcPortFifo	28	vCW1	0b	
TxFlowcArrayCtrl	29	vCW1	0b	

Address: **0xF00000 + 0x62006**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
TxFlowcModifyCtrl	30	vCW1	0b	
TxFlowcEgressCtrl	31	vCW1	0b	

11.23.2.33 SCHED_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: **0xF00000 + 0x62007**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SchedConfigSramErr	1:0	RW	11b	
FreelistSramErr	3:2	RW	11b	
SschedSramErr	5:4	RW	11b	
MonitorSramErr	7:6	RW	11b	
EschedSramErr	9:8	RW	11b	
TxqSramErr	11:10	RW	11b	
RxqMcastSramErr	13:12	RW	11b	
FifoSramErr	15:14	RW	11b	
RxPortSpacing	16	RW	1b	
TxPortSpacing	17	RW	1b	
RxPortRange	18	RW	1b	
TxPortRange	19	RW	1b	
RxPerformance	20	RW	1b	
TxPerformance	21	RW	1b	
OutOfMemory	22	RW	1b	
RxFlowcPortFifo	23	RW	1b	
RxFlowcScheduleFc	24	RW	1b	
RxFlowcScheduleRx	25	RW	1b	
RxFlowcArrayCtrl	26	RW	1b	
RxFlowcTailInfo	27	RW	1b	
TxFlowcPortFifo	28	RW	1b	
TxFlowcArrayCtrl	29	RW	1b	
TxFlowcModifyCtrl	30	RW	1b	
TxFlowcEgressCtrl	31	RW	1b	

11.24 FIBM Registers Description

11.24.1 FIBM Map

Table 11-36 FIBM Registers (reserved 2x4 KW) Map

Name	Address (Base = 0x008000)	Atomicity	Brief Description
FIBM_CFG	Base + 0x0	32	FIBM configuration register.
FIBM_SGLORT	Base + 0x1	32	The GLORT for the FIBM.
FIBM_INT	Base + 0x2	32	Interrupt GLORT and interrupt interval.
FIBM_INT_FRAME	Base + 0x3	32	Fields for an FIBM interrupt frame.
FIBM_INT_FRAME_DMAC	Base + 0x4	32	FIBM Interrupt Frame DMAC.
FIBM_INT_FRAME_SMAC	Base + 0x6	32	FIBM Interrupt Frame SMAC.
FIBM_REQUEST_CTR	Base + 0x8	32	Valid FIBM request frame counter.
FIBM_DROP_CTR	Base + 0x9	32	Invalid FIBM request frame counter.
FIBM_RESPONSE_CTR	Base + 0xA	32	FIBM response frame counter.
FIBM_INTR_CTR_0	Base + 0xB	32	Interrupt counter.
FIBM_INTR_CTR_1	Base + 0xC	32	Interrupt seq. counter.
FIBM_SRAM_CTRL	Base + 0xD	32	FIBM Memory Errors.
FIBM_IP	Base + 0xE	32	FIBM Interrupt Pending.
FIBM_IM	Base + 0xF	32	FIBM Interrupt Mask
FIBM_SCRATCH_0	Base + 0x10	32	Scratch register 0.
FIBM_SCRATCH_1	Base + 0x11	32	Scratch register 1.
FIBM_SCRATCH_2	Base + 0x12	32	Scratch register 2.
FIBM_SCRATCH_3	Base + 0x13	32	Scratch register 3.
FIBM_SCRATCH_4	Base + 0x14	32	Scratch register 4.
FIBM_SCRATCH_5	Base + 0x15	32	Scratch register 5.
FIBM_SCRATCH_6	Base + 0x16	32	Scratch register 6.
FIBM_SCRATCH_7	Base + 0x17	32	Scratch register 7.
FIBM_SCRATCH_8	Base + 0x18	32	Scratch register 8.
FIBM_SCRATCH_9	Base + 0x19	32	Scratch register 9.
FIBM_SCRATCH_10	Base + 0x1A	32	Scratch register 10.
FIBM_SCRATCH_11	Base + 0x1B	32	Scratch register 11.
FIBM_SCRATCH_12	Base + 0x1C	32	Scratch register 12.
FIBM_SCRATCH_13	Base + 0x1D	32	Scratch register 13.
FIBM_SCRATCH_14	Base + 0x1E	32	Scratch register 14.
FIBM_SCRATCH_15	Base + 0x1F	32	Scratch register 15.
PTI_TX_CTRL	Base + 0x20	32	Packet Test Interface Tx.
PTI_TX_DATA0	Base + 0x21	32	Packet Test Interface Data.

Table 11-36 FIBM Registers (reserved 2x4 KW) Map [Continued]

Name	Address (Base = 0x008000)	Atomicity	Brief Description
PTI_TX_DATA1	Base + 0x22	32	Packet Test Interface Data.
PTI_TX_CNT	Base + 0x23	32	Packet Test Interface Statistics.
PTI_RX_CTRL	Base + 0x24	32	Packet Test Interface Rx.
PTI_RX_DATA0	Base + 0x25	32	Packet Test Interface Data.
PTI_RX_DATA1	Base + 0x26	32	Packet Test Interface Data.
PTI_RX_CNT	Base + 0x27	32	Packet Test Interface Statistics.

11.24.2 FIBM Registers

11.24.2.1 FIBM_CFG

Address: 0x008000 + 0x0
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
disableIbm	0	RW	1b	Disable FIBM operation. If this bit is set, FIBM Write operations can only be done on FIBM_CFG.
padIbmResponse	1	RW	1b	Pad FIBM response frames to a length of 64 bytes.
ibmGlortEn	2	RW	0b	FIBM GLORT enable. 0b = The source GLORT for FIBM response frames is the destination GLORT of the FIBM request frame. 1b = The source GLORT for FIBM response frames and FIBM interrupt frames is ibmGlort.
interruptGlortEn	3	RW	0b	Interrupt GLORT enable. 0b = No FIBM interrupt frames are sent. 1b = Enable interrupt.
requestType	7:4	RW	0x0	Defines FTYPE/MTYPE for FIBM request frame. ** OBSOLETE, NOT IMPLEMENTED **
responseType	11:8	RW	0x0	Defines FTYPE/MTYPE for FIBM response frame.
interruptType	15:12	RW	0x0	Defines FTYPE/MTYPE for FIBM interrupt frame.
append4bytes	16	RW	0b	Defines if 4 zero bytes must be added to the response or not. If padIbmResponse is enabled, length minimal length is 64 including the 4 bytes pad.
ignoreLast4Bytes	17	RW	0b	Defines if last 4 bytes of the incoming frame is to be ignored.
tagPosition	18	RW	1b	Defines the position of the ISL tag. Only one mode supported: 0b = Byte0 1b = Reserved
Reserved	22:19	RSV	0x0	Reserved.
muxSelect	23	RW	0b	Select between FIBM function or the Packet Test Interface. 0b = FIBM 1b = PTI Cannot be changed while FIBM is in operation.

Address: **0x008000 + 0x0**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
aplCfgActive	24	RW	1b	Configure loopback mode of the Asynchronous Port Logic. Used for testing purposes only. May not be written while traffic is flowing. 0b = Standby — Loopback 1b = Active — Normal operation. FIBM Port Channel connected to the switch fabric
Reserved	31:25	RSV	0x0	Reserved.

11.24.2.2 FIBM_SGLORT

Address: **0x008000 + 0x1**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
SGlort	15:0	RW	0x0	The source GLORT for FIBM response frames and FIBM interrupt frames. This field is not used for response frames if the *ibmGlortEn* is cleared.
Reserved	31:16	RSV	0x0	Reserved.

11.24.2.3 FIBM_INT

Address: **0x008000 + 0x2**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
interruptGlort	15:0	RW	0x0	Destination GLORT for FIBM interrupt frames.
interruptInterval	31:16	RW	0x00FF	Interval for repeating FIBM interrupt frames. Increments are 8192 clock cycles (21.8 µs). Counting starts at 0b. For interruptInterval=N, the actual interval is (N+1)*81.92 µs. FIBM interrupt frames are sent repeatedly until the interrupt is cleared.

11.24.2.4 FIBM_INT_FRAME

Address: **0x008000 + 0x3**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
etherType	15:0	RW	0xFFFF	Ethertype for FIBM response frames and FIBM interrupt frames.
islSysPri	19:16	RW	0xF	FTAG.SWPRI for FIBM interrupt frames.
islUserBits	27:20	RW	0x00	FTAG.USER for FIBM interrupt frames. ** OBSOLETE — Forced to 0x00 by hardware. **
Reserved	31:28	RSV	0x0	Reserved.

11.24.2.5 FIBM_INT_FRAME_DMAC

Address: 0x008000 + 0x4
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
MacAddress	47:0	RW	0xFFFF	DMAC address to use when generating an FIBM interrupt frame.
Reserved	63:48	RSV	0x0	Reserved.

11.24.2.6 FIBM_INT_FRAME_SMAC

Address: 0x008000 + 0x6
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
MacAddress	47:0	RW	0xFFFF	SMAC address to use when generating an FIBM interrupt frame.
Reserved	63:48	RSV	0x0	Reserved.

11.24.2.7 FIBM_REQUEST_CTR

Address: 0x008000 + 0x8
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	31:0	RW	0x0	Valid FIBM request frames received.

11.24.2.8 FIBM_DROP_CTR

Address: 0x008000 + 0x9
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	31:0	RW	0x0	Invalid FIBM frames received.

11.24.2.9 FIBM_RESPONSE_CTR

Address: 0x008000 + 0xA
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	31:0	RW	0x0	Total FIBM responses sent.

11.24.2.10 FIBM_INTR_CTR_0

Address: 0x008000 + 0xB
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	31:0	RW	0x0	Total FIBM interrupt frames sent.

11.24.2.11 FIBM_INTR_CTR_1

Address: 0x008000 + 0xC
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	31:0	RW	0x0	FIBM interrupt frames sent since interrupt condition was set.

11.24.2.12 FIBM_SRAM_CTRL

FIBM memory error injection and BIST status reporting register. There is only one memory in FIBM used for the rx frame buffer.

Address: 0x008000 + 0xD
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	For error injection. 00b = Normal write operations. 01b = Generate a correctable error. 10b = Generate a correctable error. 11b = Generate an uncorrectable error.
CErr	2	vCW1	0b	Correctable errors.
UErr	3	vCW1	0b	Uncorrectable errors.
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.24.2.13 FIBM_IP

FIBM Interrupt Pending register.

Address:	**0x008000 + 0xE**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PtiTxCorrupt	0	vCW1	0b	Set when new Packet Test Interface TX register data is written before the previous data has been sent.
SramErr[0..1]	2:1	vCW1	0b	(2 x 1-bit) Set when a memory error is detected in FIBM. Bits correspond to memory error type: Bit 0 = Correctable Bit 1 = Uncorrectable See FIBM_SRAM_CTRL register to determine source of error.
Reserved	31:3	RSV	0x0	Reserved.

11.24.2.14 FIBM_IM

The interrupt mask register controls whether the corresponding interrupt source in FIBM_IP is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address:	**0x008000 + 0xF**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PtiTxCorrupt	0	RW	1b	
SramErr[0..1]	2:1	RW	1b	(2 x 1-bit)
Reserved	31:3	RSV	0x0	Reserved.

11.24.2.15 FIBM_SCRATCH_0

Address:	**0x008000 + 0x10**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch0	31:0	RW	0x0	Scratch register 0.

11.24.2.16 FIBM_SCRATCH_1

Address:	**0x008000 + 0x11**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch1	31:0	RW	0x0	Scratch register 1.

11.24.2.17 FIBM_SCRATCH_2

Address: **0x008000 + 0x12**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
scratch2	31:0	RW	0x0	Scratch register 2.

11.24.2.18 FIBM_SCRATCH_3

Address: **0x008000 + 0x13**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
scratch3	31:0	RW	0x0	Scratch register 3.

11.24.2.19 FIBM_SCRATCH_4

Address: **0x008000 + 0x14**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
scratch4	31:0	RW	0x0	Scratch register 4.

11.24.2.20 FIBM_SCRATCH_5

Address: **0x008000 + 0x15**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
scratch5	31:0	RW	0x0	Scratch register 5.

11.24.2.21 FIBM_SCRATCH_6

Address: **0x008000 + 0x16**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
scratch6	31:0	RW	0x0	Scratch register 6.

11.24.2.22 FIBM_SCRATCH_7

Address: 0x008000 + 0x17 **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch7	31:0	RW	0x0	Scratch register 7.

11.24.2.23 FIBM_SCRATCH_8

Address: 0x008000 + 0x18 **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch8	31:0	RW	0x0	Scratch register 8.

11.24.2.24 FIBM_SCRATCH_9

Address: 0x008000 + 0x19 **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch9	31:0	RW	0x0	Scratch register 9.

11.24.2.25 FIBM_SCRATCH_10

Address: 0x008000 + 0x1A **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch10	31:0	RW	0x0	Scratch register 10.

11.24.2.26 FIBM_SCRATCH_11

Address: 0x008000 + 0x1B **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
scratch11	31:0	RW	0x0	Scratch register 11.

11.24.2.27 FIBM_SCRATCH_12

Address: 0x008000 + 0x1C
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
scratch12	31:0	RW	0x0	Scratch register 12.

11.24.2.28 FIBM_SCRATCH_13

Address: 0x008000 + 0x1D
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
scratch13	31:0	RW	0x0	Scratch register 13.

11.24.2.29 FIBM_SCRATCH_14

Address: 0x008000 + 0x1E
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
scratch14	31:0	RW	0x0	Scratch register 14.

11.24.2.30 FIBM_SCRATCH_15

Address: 0x008000 + 0x1F
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
scratch15	31:0	RW	0x0	Scratch register 15.

11.24.2.31 PTI_TX_CTRL

For sending frame data through the port channel of the FIBM when muxSelect is set to 1b.

Setting *TxValid* to 1b initiates a data transmission using the values in this register and PTI_TX_DATA* registers. If another write occurs to any of these registers before *TxValid* is lowered, the data may be corrupted, and an interrupt is reported to FIBM_IP.*PtiTxCorrupt*.

Address: 0x008000 + 0x20
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Mark	1:0	RW	00b	Sets the mark bits on the WidePortChannel.
Info	3:2	RW	00b	Sets the info bits on the WidePortChannel.

Address: **0x008000 + 0x20**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Len	7:4	RW	0x0	Sets the len bits on the WidePortChannel.
TxValid	8	vRW	0b	A token transfer is initiated each time a 1b is written. This field automatically resets to 0b after an operation completes.
TxEnable	9	vRO	0b	Reports the enable bit from the PTI TX logic. A frame data token is sent when both *TxValid* and *TxEnable* are high.
Reserved	31:10	RSV	0x0	Reserved.

11.24.2.32 PTI_TX_DATA0

Write to this register only when PTI_TX_CTRL.*TxValid* is 0b. Otherwise, the frame data may become corrupted.

Address: **0x008000 + 0x21**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Lower 32 bits of the WidePortChannel data.

11.24.2.33 PTI_TX_DATA1

Write to this register only when PTI_TX_CTRL.*TxValid* is 0b. Otherwise, the frame data may become corrupted.

Address: **0x008000 + 0x22**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Upper 32 bits of the WidePortChannel data.

11.24.2.34 PTI_TX_CNT

Address: **0x008000 + 0x23**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Count	31:0	vRW	0x0	Number of valid frame data tokens sent. Value increments with each transaction.

11.24.2.35 PTI_RX_CTRL

For receiving frame data through the port channel of the FIBM when *muxSelect* is set to 1b.

Data in this register and PTI_RX_DATA* registers are valid when *RxValid* is 1b. After reading the data, clear the token by writing 1b to *RxEnable*.

Address: 0x008000 + 0x24
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Mark	1:0	vRO	00b	From the mark bits on the WidePortChannel.
Info	3:2	vRO	00b	From the info bits on the WidePortChannel.
Len	7:4	vRO	0x0	From the len bits on the WidePortChannel.
RxValid	8	vRO	0b	Reports whether valid frame data is present.
RxEnable	9	vRW	0b	A frame data token is cleared each time a 1b is written. This field automatically resets to 0b after an operation completes.
ContinuousDrain	10	RW	0b	Mode to allow continuous draining of frame data. If set to 1b, RxEnable is not automatically reset to 0b after one transfer cycle.
Reserved	31:11	RSV	0x0	Reserved.

11.24.2.36 PTI_RX_DATA0

Address: 0x008000 + 0x25
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	Lower 32 bits of the WidePortChannel data.

11.24.2.37 PTI_RX_DATA1

Address: 0x008000 + 0x26
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	Upper 32 bits of the WidePortChannel data.

11.24.2.38 PTI_RX_CNT

Address: 0x008000 + 0x27
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Count	31:0	vRW	0x0	Number of valid frame data tokens received. Value increments with each transaction.

11.25 MGMT Registers Description

11.25.1 MGMT Map

Table 11-37 Management Registers (reserved 16 KW) Map

Name	Address (Base = 0x000000)	Atomicity	Brief Description
FATAL_CODE	Base + 0x0	32	Fatal Code received from crossbar.
LAST_FATAL_CODE	Base + 0x1	32	Fatal Code registered at last reset.
FATAL_COUNT	Base + 0x2	32	Fatal reset count register.
SOFT_RESET	Base + 0x3	32	Soft reset control register.
DEVICE_CFG	Base + 0x4	32	Global device configuration.
RESET_CFG	Base + 0x5	32	Reset configuration register.
WATCHDOG_CFG	Base + 0x6	32	Watchdog configuration register.
MGMT_SCRATCH[0..1]	Base + 0x1*i + 0x8	32	Scratch registers.
VITAL_PRODUCT_DATA	Base + 0x304	32	Product identification register.
GLOBAL_INTERRUPT_DETECT	Base + 0x400	32	Global interrupt status and control register.
INTERRUPT_MASK_INT	Base + 0x402	32	Interrupt MASK to pin INT_N.
INTERRUPT_MASK_PCIE[0..8]	Base + 0x2*i + 0x420	32	Interrupt MASK to PCI Express.
INTERRUPT_MASK_FIBM	Base + 0x440	32	Interrupt MASK to FIBM.
INTERRUPT_MASK_BSM	Base + 0x442	32	Interrupt MASK to Boot State Machine.
CORE_INTERRUPT_DETECT	Base + 0x444	32	Core Interrupt Detect.
CORE_INTERRUPT_MASK	Base + 0x445	32	Core Interrupt Mask.
SRAM_ERR_IP	Base + 0x446	32	SRAM Error IP.
SRAM_ERR_IM	Base + 0x448	32	SRAM Error IM.
PINS_STAT	Base + 0x44A	32	
PINS_IP	Base + 0x44B	32	
PINS_IM	Base + 0x44C	32	
SW_IP	Base + 0x44D	32	Software Writeable Interrupt.
SW_IM	Base + 0x44E	32	Software Writeable Interrupt Mask.
SW_TEST_AND_SET	Base + 0x44F	32	Software Test and Set Register and Mask.
LSM_CLKOBS_CTRL	Base + 0x450	32	
CHIP_VERSION	Base + 0x452	32	Chip Mask Version Register.
BSM_SCRATCH[0..1023]	Base + 0x1*i + 0x800	32	Boot State Machine scratch memory.
BSM_CTRL	Base + 0xC00	32	Boot State Machine Control Register.
BSM_ARGS	Base + 0xC01	32	Boot State Machine Arguments Register.
BSM_ADDR_OFFSET[0..3]	Base + 0x1*i + 0xC04	32	Boot State Machine Address Offset.
BSM_COUNTER[0..1]	Base + 0x1*i + 0xC08	32	Boot State Machine Loop Counter.
BSM_SRAM_CTRL	Base + 0xC0A	32	BSM Memory Errors.

Table 11-37 Management Registers (reserved 16 KW) Map [Continued]

Name	Address (Base = 0x000000)	Atomicity	Brief Description
BSM_IP	Base + 0xC0B	32	BSM Interrupt Pending.
BSM_IM	Base + 0xC0C	32	BSM Interrupt Mask.
PIN_STRAP_STAT	Base + 0xC0D	32	Pin Strapping Status.
FUSE_DATA_0	Base + 0xC0E	32	Fuse contents.
FUSE_DATA_1	Base + 0xC0F	32	Fuse contents.
BIST_CTRL	Base + 0xC10	32	BIST Control Register.
REI_CTRL	Base + 0xC12	32	RAM EFUSE Interface Control Register.
REI_STAT	Base + 0xC13	32	RAM EFUSE Interface Status Register.
GPIO_CFG	Base + 0xC15	32	GPIO Configuration Register.
GPIO_DATA	Base + 0xC16	32	GPIO Data Register.
GPIO_IP	Base + 0xC17	32	GPIO Interrupt Pending Register.
GPIO_IM	Base + 0xC18	32	GPIO Interrupt Mask Register.
I2C_CFG	Base + 0xC19	32	I^{2}C Configuration.
I2C_DATA[0..2]	Base + 0x1*i + 0xC1C	32	I^{2}C Data Write Register.
I2C_CTRL	Base + 0xC20	32	I^{2}C Control.
MDIO_CFG	Base + 0xC22	32	MDIO Configuration
MDIO_DATA	Base + 0xC23	32	MDIO Data Register
MDIO_CTRL	Base + 0xC24	32	MDIO Control
SPI_TX_DATA	Base + 0xC26	32	
SPI_RX_DATA	Base + 0xC27	32	
SPI_HEADER	Base + 0xC28	32	
SPI_CTRL	Base + 0xC29	32	
LED_CFG	Base + 0xC2B	32	LED Configuration Register.
SCAN_DATA_IN	Base + 0xC2D	32	Scan requests to SCAN_TOP.
CRM_DATA[0..2047][0..1]	Base + 0x2*j + 0x1*i + 0x1000	32	Command data space.
CRM_CTRL	Base + 0x2000	32	Control Register.
CRM_STATUS	Base + 0x2001	32	Status Register.
CRM_TIME	Base + 0x2002	32	Current Time.
CRM_SRAM_CTRL	Base + 0x2004	32	
CRM_IP	Base + 0x2008	32	
CRM_IM	Base + 0x200C	32	
CRM_COMMAND[0..63]	Base + 0x2*i + 0x2080	32	Commands Definition.
CRM_REGISTER[0..63]	Base + 0x2*i + 0x2100	32	Register Addressing Definition.
CRM_PERIOD[0..63]	Base + 0x2*i + 0x2180	32	Command Time Control.
CRM_PARAM[0..63]	Base + 0x1*i + 0x2200	32	Command Parameters.
PLL_PCIE_CTRL	Base + 0x2241	32	PLL configuration register.
PLL_PCIE_STAT	Base + 0x2242	32	PLL status register.

Table 11-37 Management Registers (reserved 16 KW) Map [Continued]

Name	Address (Base = 0x000000)	Atomicity	Brief Description
SBUS_PCIE_CFG	Base + 0x2243	32	SBUS Controller configuration.
SBUS_PCIE_COMMAND	Base + 0x2244	32	SBUS Command Execution.
SBUS_PCIE_REQUEST	Base + 0x2245	32	SBUS Data Write Request.
SBUS_PCIE_RESPONSE	Base + 0x2246	32	SBUS DATA Read Response.
SBUS_PCIE_SPICO_IN	Base + 0x2247	32	
SBUS_PCIE_SPICO_OUT	Base + 0x2248	32	
SBUS_PCIE_IP	Base + 0x2249	32	
SBUS_PCIE_IM	Base + 0x224A	32	
SYSTIME_CFG	Base + 0x224C	32	
SYSTIME	Base + 0x224E	32	
SYSTIME0	Base + 0x2250	32	
SYSTIME_PULSE[0..1]	Base + 0x1*i + 0x2252	32	
SYSTIME_CAPTURE[0..3]	Base + 0x2*i + 0x2258	64	
PCIE_XPLL_CTRL	Base + 0x3000	32	
PCIE_CLK_CTRL	Base + 0x3001	32	
PCIE_CLK_CTRL2	Base + 0x3002	32	
PCIE_CLKMON_RATIO_CFG	Base + 0x3003	32	
PCIE_CLKMON_TOLERANCE_CFG	Base + 0x3004	32	
PCIE_CLKMON_DEADLINES_CFG	Base + 0x3005	32	
PCIE_CLK_STAT	Base + 0x3006	32	
PCIE_CLK_IP	Base + 0x3007	32	
PCIE_CLK_IM	Base + 0x3008	32	
PCIE_WARM_RESET_DELAY	Base + 0x3009	32	

11.25.2 MGMT Registers

11.25.2.1 FATAL_CODE

Writing to this register causes a master reset to the chip if the *FatalResetEnable* is set. The code written is saved into LAST_FATAL_CODE. Any value can be written by software.

The hardware automatically writes a fatal code (CRM=1, only possible event) if the event is hardware initiated. Software could write any value.

Address: 0x000000 + 0x0
Atomicity: 32
Reset Domains: DEVICE

Field Name	Bit(s)	Type	Default	Description
FatalCode	7:0	WO	0x0	
Reserved	31:8	RSV	0x0	Reserved.

11.25.2.2 LAST_FATAL_CODE

Address:	**0x000000 + 0x1**			
Atomicity:	**32**			
Reset Domains:	**DEVICE**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FatalCode	7:0	vRW	0x0	Latched copy of FATAL_CODE after reset.
Reserved	31:8	RSV	0x0	Reserved.

11.25.2.3 FATAL_COUNT

Address:	**0x000000 + 0x2**			
Atomicity:	**32**			
Reset Domains:	**DEVICE**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ResetCount	7:0	vCW	0x0	Counts the number of fatal interrupt resets that occurred since last time the CHIP_RESET_N was asserted. This register can be reset to zero by writing any value to it (clear-on-write).
Reserved	31:8	RSV	0x0	Reserved.

11.25.2.4 SOFT_RESET

Address:	**0x000000 + 0x3**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ColdReset	0	RW	1b	Controls global cold reset. Automatically asserted (set to 1b) at device reset. Boot sequencer clears this bit only when it is known that all PLLs are locked and clocks are stable.
EPLReset	1	RW	1b	Controls reset on all EPLs. When de-asserted (0), must remain de-asserted for a minimum of 100 ns.
SwitchReset	2	RW	1b	Controls reset of switch (fabric) and Tunneling Engines. When de-asserted (0), must remain de-asserted for a minimum of 100 ns
SwitchReady	3	RW	0b	Controls ready state of switch (fabric).
PCIeReset[0..8]	12:4	vRW	1b	(9 x 1-bit) Controls reset of each PCIe module. The PCIe[i] module is in reset when the following is true: `(PCIeReset[i] | ~(PCIE_RESET_N[i] | PCIeActive[i]))` Thus, when neither *PCIeReset* nor *PCIeActive* is set, the reset state of PCIe[i] is controlled by the PCIE_RESET_N[i] pin. When DEVICE_CFG.*FeatureCode* is nonzero, certain bits of *PCIeReset* are forced to 0x1.
PCIeActive[0..8]	21:13	RW	0b	(9 x 1-bit) See description of *PCIeReset* field (bits 12:4).
Reserved	31:22	RSV	0x0	Reserved.

11.25.2.5 DEVICE_CFG

May not be written while traffic is flowing.

Address:	**0x000000 + 0x4**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PCIeMode[0..3]	3:0	vRW	0b	(4 x 1-bit) Defines the mode of operation for each pair of PCIe Host Interfaces: 0b = MODE_1x8 — The host interface operates as a single 8-lane interfaces. 1b = MODE_2x4 — The host interface operates as two independent 4-lane interfaces. This field may be modified only when the two PEPs are both in WARM reset. It is stable 1 µs before the PEPs exit the WARM reset.
Eth100GDisabled	4	SW1	0b	Indicates if 100 GbE is disabled or not. Some SKUs are not supporting 100 GbE. If disabled (set to 1b), is not allowed to enable (set to 0b) again.
FeatureCode	6:5	SW1	00b	Indicates which PCIe host interface modes are supported: 00b = FULL — All hosts can operate in either PCIeMode (except for host 8, which is always 1 lane). 01b = HALF — PCIe hosts 0,1,2,3,4,6,8 can be used. Hosts 0,1,2,3 can operate in either mode. Hosts 4 and 6 can only be used in 4-lane mode. 10b = BASIC — PCIe host 0 or 8 can be used but not both. PCIe host 0 can only be used in 4-lane mode. 11b = Reserved. For BASIC, if system board attempts to enable both, only PCIe host 0 is enabled.
PCIeEnable[0..8]	15:7	RW	1b	(9 x 1-bit) Enable bit per PCIe Host Interface. Used to gate clock and save power for unused interfaces. Bits of this field may be modified only when the corresponding PEP(s) are in WARM reset, and are stable 1 µs before the PEP(s) exit the warm reset. Accessing registers of disabled PEP is not allowed, as it creates a deadlock.
SystimeClockSource	16	RW	0b	Defines the reference clock source for SYSTIME. This field can only be modified when cold reset is asserted. The choices are: 0b = PCIE_REFCLK (100 MHz) 1b = IEEE1588_REFCLK (50-256 MHz)
Reserved	31:17	RSV	0x0	Reserved.

11.25.2.6 RESET_CFG

Address:	**0x000000 + 0x5**			
Atomicity:	**32**			
Reset Domains:	**DEVICE**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MasterResetAssertCount	7:0	RW	0x10	Number of cycles the MASTER_RESET signal is asserted. A minimum count of 1 is required.
MasterResetDelayCount	15:8	RW	0x40	Number of cycles the Watchdog waits before asserting MASTER_RESET after a fatal condition is detected.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.7 WATCHDOG_CFG

Address:	**0x000000 + 0x6**			
Atomicity:	**32**			
Reset Domains:	**DEVICE**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FatalResetEnable	0	RW	0b	Defines if the writing to FATAL_CODE causes a chip reset or not.
Reserved	31:1	RSV	0x0	Reserved.

11.25.2.8 MGMT_SCRATCH[0..1]

Address:	**0x000000 + 0x1*i + 0x8**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Data	31:0	RW	0x0	Scratch registers. No function.

11.25.2.9 VITAL_PRODUCT_DATA

Address:	**0x000000 + 0x304**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PartNumber	15:0	RO	0xAE21	The part number.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.10 GLOBAL_INTERRUPT_DETECT

This register reports the active interrupt sources in the switch. The contents of this register are masked by the INTERRUPT_MASK registers to select which interrupts get posted by PCIe Host Interfaces, on INT_N pin, via FIBM, or handled by the BSM.

Address:	**0x000000 + 0x400**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PCIE_BSM[0..8]	8:0	vCW1	0b	(9 x 1-bit) PCIe interrupt pending. Enabled by PCIE_IB (1 per host). See corresponding PCIE_IP for the source of interrupt.
PCIE[0..8]	17:9	vCW1	0b	(9 x 1-bit) PCIe interrupt pending. Enabled by PCIE_IM (1 per host). See corresponding PCIE_IP for the source of interrupt.
EPL[0..8]	26:18	vCW1	0b	(9 x 1-bit) EPL interrupt pending (1 per port). See corresponding EPL_IP[0..8] to determine the source of interrupt.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x400** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TUNNEL[0..1]	28:27	vCW1	0b	(2 x 1-bit) TUNNEL interrupt pending (1 per tunnel engine). See corresponding TE_IP[0..1] to determine the source of interrupt.
CORE	29	vCW1	0b	General CORE interrupts. See CORE_INTERRUPT_DETECT to determine which ports have generated the source of the interrupt.
SOFTWARE	30	vCW1	0b	Software interrupt pending. See SW_IP to determine the source of the interrupt.
GPIO	31	vCW1	0b	GPIO interrupt pending. See GPIO_IP to determine the source of the interrupt.
I2C	32	vCW1	0b	I^{2}C interrupt pending. Write 1b into I2C_CTRL.*InterruptEnable* to clear the interrupt (refer to Section 11.25.2.47).
MDIO	33	vCW1	0b	MDIO interrupt pending. Write 1b into MDIO_CTRL.*InterruptEnable* to clear the interrupt (refer to Section 11.25.2.50).
CRM	34	vCW1	0b	Counter Rate Monitor interrupt pending. See CRM_IP to determine the source of interrupt.
FH_TAIL	35	vCW1	0b	Frame Handler tail interrupt. See FH_TAIL_IP to determine the source of the interrupt.
FH_HEAD	36	vCW1	0b	Frame Handler head interrupt. See FH_HEAD_IP to determine the source of the interrupt.
SBUS_EPL	37	vCW1	0b	SBUS EPL interrupt notification. See SBUS_EPL_IP to determine the source of the interrupt.
SBUS_PCIE	38	vCW1	0b	SBUS PCIE interrupt notification. See SBUS_PCIE_IP to determine the source of the interrupt.
PINS	39	vCW1	0b	Detected transition on various pins. See PINS_STAT, PINS_IP and PINS_IM.
FIBM	40	vCW1	0b	FIBM interrupt pending. See FIBM_IP to determine the source of interrupt.
BSM	41	vCW1	0b	FIBM interrupt pending. See BSM_IP to determine the source of interrupt.
XCLK	42	vCW1	0b	XCLK interrupt pending. See PCIE_CLK_IP to determine the source of interrupt.
Reserved	63:43	RSV	0x0	Reserved.

11.25.2.11 INTERRUPT_MASK_INT

Select which interrupt sources from GLOBAL_INTERRUPT_DETECT is presented to the INT_N pin.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x402** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PCIE_BSM[0..8]	8:0	RW	1b	(9 x 1-bit)
PCIE[0..8]	17:9	RW	1b	(9 x 1-bit)
EPL[0..8]	26:18	RW	1b	(9 x 1-bit)

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x402** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TUNNEL[0..1]	28:27	RW	1b	(2 x 1-bit)
CORE	29	RW	1b	
SOFTWARE	30	RW	1b	
GPIO	31	RW	1b	
I2C	32	RW	1b	
MDIO	33	RW	1b	
CRM	34	RW	1b	
FH_TAIL	35	RW	1b	
FH_HEAD	36	RW	1b	
SBUS_EPL	37	RW	1b	
SBUS_PCIE	38	RW	1b	
PINS	39	RW	1b	
FIBM	40	RW	1b	
BSM	41	RW	1b	
XCLK	42	RW	1b	
Reserved	63:43	RSV	0x0	Reserved.

11.25.2.12 INTERRUPT_MASK_PCIE[0..8]

Select which interrupt sources from GLOBAL_INTERRUPT_DETECT is presented for generation of in-band PCI Express interrupts.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2*i + 0x420** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PCIE_BSM[0..8]	8:0	RW	1b	(9 x 1-bit)
PCIE[0..8]	17:9	RW	1b	(9 x 1-bit)
EPL[0..8]	26:18	RW	1b	(9 x 1-bit)
TUNNEL[0..1]	28:27	RW	1b	(2 x 1-bit)
CORE	29	RW	1b	
SOFTWARE	30	RW	1b	
GPIO	31	RW	1b	
I2C	32	RW	1b	
MDIO	33	RW	1b	
CRM	34	RW	1b	
FH_TAIL	35	RW	1b	
FH_HEAD	36	RW	1b	
SBUS_EPL	37	RW	1b	
SBUS_PCIE	38	RW	1b	
PINS	39	RW	1b	
FIBM	40	RW	1b	

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2*i + 0x420** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BSM	41	RW	1b	
XCLK	42	RW	1b	
Reserved	63:43	RSV	0x0	Reserved.

11.25.2.13 INTERRUPT_MASK_FIBM

Select which interrupt sources from GLOBAL_INTERRUPT_DETECT is presented for generation of in-band FIBM interrupts.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x440** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PCIE_BSM[0..8]	8:0	RW	1b	(9 x 1-bit)
PCIE[0..8]	17:9	RW	1b	(9 x 1-bit)
EPL[0..8]	26:18	RW	1b	(9 x 1-bit)
TUNNEL[0..1]	28:27	RW	1b	(2 x 1-bit)
CORE	29	RW	1b	
SOFTWARE	30	RW	1b	
GPIO	31	RW	1b	
I2C	32	RW	1b	
MDIO	33	RW	1b	
CRM	34	RW	1b	
FH_TAIL	35	RW	1b	
FH_HEAD	36	RW	1b	
SBUS_EPL	37	RW	1b	
SBUS_PCIE	38	RW	1b	
PINS	39	RW	1b	
FIBM	40	RW	1b	
BSM	41	RW	1b	
XCLK	42	RW	1b	
Reserved	63:43	RSV	0x0	Reserved.

11.25.2.14 INTERRUPT_MASK_BSM

Select which interrupt sources from GLOBAL_INTERRUPT_DETECT are presented for execution of commands from serial ROM.

Address: 0x000000 + 0x442
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
PCIE_BSM[0..8]	8:0	RW	1b	(9 x 1-bit)
PCIE[0..8]	17:9	RW	1b	(9 x 1-bit)
EPL[0..8]	26:18	RW	1b	(9 x 1-bit)
TUNNEL[0..1]	28:27	RW	1b	(2 x 1-bit)
CORE	29	RW	1b	
SOFTWARE	30	RW	1b	
GPIO	31	RW	1b	
I2C	32	RW	1b	
MDIO	33	RW	1b	
CRM	34	RW	1b	
FH_TAIL	35	RW	1b	
FH_HEAD	36	RW	1b	
SBUS_EPL	37	RW	1b	
SBUS_PCIE	38	RW	1b	
PINS	39	RW	1b	
FIBM	40	RW	1b	
BSM	41	RW	1b	
XCLK	42	RW	1b	
Reserved	63:43	RSV	0x0	Reserved.

11.25.2.15 CORE_INTERRUPT_DETECT

Holds various errors and interrupts from different blocks in the system. This register is not normally used except to recover source of SRAM errors.

Address: 0x000000 + 0x444
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
MODIFY	0	vRO	0b	MODIFY interrupt. See MOD_IP to determine the source of interrupt.
FH_HEAD	1	vRO	0b	Frame Handler head interrupt. See FH_HEAD_IP to determine the source of interrupt.
FH_TAIL	2	vRO	0b	Frame Handler tail interrupt. See FH_TAIL_IP to determine the source of interrupt.
SCHEDULER	3	vRO	0b	Scheduler interrupt. See SCHED_IP to determine the source of interrupt.
Reserved	28:4	RSV	0x0	Reserved.

Address: **0x000000 + 0x444**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
SRAM_ERR	29	vRO	0b	SRAM memory errors from ARRAY_SUBSEG units. See SRAM_ERR_IP to determine the source of interrupt.
INGRESS_ERR	30	vRO	0b	Asynchronous memory error from the INGRESS crossbar.
EGRESS_ERR	31	vRO	0b	Asynchronous memory error from the EGRESS crossbar.

11.25.2.16 CORE_INTERRUPT_MASK

Select which core interrupts are represented in GLOBAL_INTERRUPT_DETECT.*Core*. Setting to 1b blocks the interrupt to be reported in GLOBAL_INTERRUPT_DETECT.*Core*.

Address: **0x000000 + 0x445**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
MODIFY	0	RW	1b	
FH_HEAD	1	RW	1b	
FH_TAIL	2	RW	1b	
SCHEDULER	3	RW	1b	
Reserved	28:4	RW	0x0	
SRAM_ERR	29	RW	1b	
INGRESS_ERR	30	RW	1b	
EGRESS_ERR	31	RW	1b	

11.25.2.17 SRAM_ERR_IP

Logs SRAM errors on main shared frame memory or ingress XBAR data memory. Two bits are provided for every sub-segment (total 24 sub-segments, each sub-segment is 64-bits long). The bits are encoded "subseg*2+i" where "i" is 0 for correctable and 1 for uncorrectable. Correctable error is logged for main memory only and is never logged for XBAR data memory, as the XBAR data memory are parity protected only and thus uncorrectable. For uncorrectable error, there is no method to differentiate between XBAR data memory and main shared frame memory.

Address: **0x000000 + 0x446**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Error[0..47]	47:0	vCW1	0b	(48 x 1-bit) Logs if an error was detected. On any specific bit: 0b = No effect. 1b = Clears the error.
Reserved	63:48	RSV	0x0	Reserved.

11.25.2.18 SRAM_ERR_IM

Defines if errors are reported up or not.

Address: 0x000000 + 0x448
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Mask[0..47]	47:0	RW	1b	(48 x 1-bit) Defines if error are RW reported in CORE_INTERRUPT_DETECT. On any specific bit: 0b = Enables reporting. 1b = Blocks reporting this error.
Reserved	63:48	RSV	0x0	Reserved.

11.25.2.19 PINS_STAT

Reports current state of various pins.

Address: 0x000000 + 0x44A
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
PCIE_RESET_N[0..8]	8:0	vRO	0b	(9 x 1-bit) Current status of PCIE_RESET_N pins.
Reserved	31:9	RSV	0x0	Reserved.

11.25.2.20 PINS_IP

Reports variation on some pins.

Address: 0x000000 + 0x44B
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
DataDetectHigh	8:0	vCW1	0x0	Detected transition from 0b to 1b on various pins. See PINS_STAT for pins watched. On any specific bit: 0b = No effect. 1b = Clears that bit.
Reserved	15:9	RSV	0x0	Reserved.
DataDetectLow	24:16	vCW1	0x0	Detected transition from 1b to 0b on various pins. See PINS_STAT for pins watched. On any specific bit: 0b = No effect. 1b = Clears that bit.
Reserved	31:25	RSV	0x0	Reserved.

11.25.2.21 PINS_IM

Mask interrupts from pins transition detections.

Address: 0x000000 + 0x44C
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
MaskDataDetectHigh	8:0	RW	0x1FF	Mask detection of transition from 0b to 1b on various pins. 0b = Enables interrupt. 1b = Blocks low to high interrupt. See PINS_STAT for pins watched.
Reserved	15:9	RSV	0x0	Reserved.
MaskDataDetectLow	24:16	RW	0x1FF	Mask detection of transition from 1b to 0b various pins. 0b = Enables interrupt. 1b = Blocks low to high interrupt. See PINS_STAT for pins watched.
Reserved	31:25	RSV	0x0	Reserved.

11.25.2.22 SW_IP

Address: 0x000000 + 0x44D
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SoftIP	31:0	RW	0x0	

11.25.2.23 SW_IM

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0x000000 + 0x44E
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SoftIM	31:0	RW	0xFFFFFFFF	

11.25.2.24 SW_TEST_AND_SET

Test and set register. The mask is used to exclude some bits from the operation. If the unmasked register content is zero and the new value is not zero, the new value is loaded into the register unmasked bits. If the new value is zero, then this new value is loaded into the register unmasked bits regardless of its current content.

The operation is atomic and the pseudo code is:

```
if ( (SW_TEST_AND_SET.Data & ~Mask) == 0 || (Data & ~Mask) == 0 )
     SW_TEST_AND_SET.Data = (SW_TEST_AND_SET.Data & Mask) | (Data & ~Mask)
     SW_TEST_AND_SET.Mask = Mask;
```

Address: 0x000000 + 0x44F
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	15:0	RW	0x0	New value to write.
Mask	31:16	RW	0x0	Defines bits to keep unchanged in the register. If *Mask* bit "i" is set to 1b, the *Data* bit "i" is not affected during the TEST and SET operation.

11.25.2.25 LSM_CLKOBS_CTRL

Control register for the clock observation pads in LSM_PADS.

Address: 0x000000 + 0x450
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SelC	3:0	RW	0x0	Clock mux selection for clkobs pad CLKOUT_C: 0b = CLKOUT_C held low 1b = PCIe PLL output divided by 8.
DivC	5:4	RW	00b	Additional output clock divider: 00b = No division 01b = Divide-by-2 10b = Divide-by-3 11b = Divide-by-8
Reserved	31:6	RSV	0x0	Reserved.

11.25.2.26 CHIP_VERSION

Address: 0x000000 + 0x452
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Version	6:0	RO	0x1	Returns the chip mask version. 0x1 = FM10000 B0 step.
Reserved	31:7	RSV	0x0	Reserved.

11.25.2.27 BSM_SCRATCH[0..1023]

Available for general scratch memory use.

Address: 0x000000 + 0x1*i + 0x800
Atomicity: 32
Reset Domains: COLD_MEM

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	

11.25.2.28 BSM_CTRL

Address:	**0x000000 + 0xC00**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Command	3:0	RW	0x0	Command to execute. Command must be reset to 0b before a new command can be executed. The valid commands are: 0000b = NONE 1000b = LOAD_FROM_SPI — Request loading from boot. 1001b = STOP_BOOT — Request stopping boot load. All other values are reserved.
CommandDone	4	vRO	0b	Indicates if the command requested is completed.
EepromLoadDone	5	vRO	0b	Indicates if loading from the EEPROM is complete.
EepromError	6	vRO	0b	Indicates if loading stopped because an error was encountered.
EepromEnable	7	vRO	0b	Indicates if loading is active.
EepromAddr	31:8	vRO	0x0	Indicates next address to read from the EEPROM. If the EEPROM terminates before the end of stream (error), the address points to the instruction following the instructions that failed.

11.25.2.29 BSM_ARGS

Defines extra data to used for commands.

Address:	**0x000000 + 0xC01**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Data	31:0	RW	0x0	For LOAD_FROM_SPI, the address must be loaded in DATA[23:0] and the options in DATA[31:24], in the same format as bytes 0-3 of a serial ROM.

11.25.2.30 BSM_ADDR_OFFSET[0..3]

Configurable base address for BSM. Index 0 if unused. There are 3 programmable offsets: BSM_ADDR_OFFSET[1..3]. BSM_ADDR_OFFSET[0] is read-only and always treated as 0x0.

Address:	**0x000000 + 0x1*i + 0xC04**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Offset	23:0	RW	0x0	Configurable base address referred to in the BSM instruction encoding.
Reserved	31:24	RSV	0x0	Reserved.

11.25.2.31 BSM_COUNTER[0..1]

Address: 0x000000 + 0x1*i + 0xC08
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Counter	15:0	vRW	0x0	Loop counter used by the BSM LOOP instruction.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.32 BSM_SRAM_CTRL

Boot State Machine memory error injection and BIST status reporting register. There is only one memory in BSM used for BSM_SCRATCH.

Address: 0x000000 + 0xC0A
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	For error injection. 00b = Normal write operations. 01b = Generate a correctable error. 10b = Generate a correctable error. 11b = Generate an uncorrectable error.
CErr	2	vCW1	0b	Correctable errors.
UErr	3	vCW1	0b	Uncorrectable errors.
BistDonePass	4	vRO	0b	
BistDoneFail	5	vRO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.25.2.33 BSM_IP

Address: 0x000000 + 0xC0B
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SramErr[0..1]	1:0	vCW1	0b	(2 x 1-bit) Set when a memory error is detected in the BSM. Bits correspond to memory error type: Bit 0 = Correctable Bit 1 = Uncorrectable See BSM_SRAM_CTRL register to determine source of error.
Reserved	31:2	RSV	0x0	Reserved.

11.25.2.34 BSM_IM

The interrupt mask register controls whether the corresponding interrupt source in BSM_IP is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0x000000 + 0xC0C
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SramErr[0..1]	1:0	RW	1b	(2 x 1-bit)
Reserved	31:2	RSV	0x0	Reserved.

11.25.2.35 PIN_STRAP_STAT

Address: 0x000000 + 0xC0D
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ethClkTerm	0	vRO	0b	Indicates the strapping status of ETHCLK_TERMINATION.
pcieClkTerm	1	vRO	0b	Indicates the strapping status of PCIECLK_TERMINATION.
ieee1588ClkTerm	2	vRO	0b	Indicates the strapping status of 1588CLK_TERMINATION.
Reserved	31:3	RSV	0x0	Reserved.

11.25.2.36 FUSE_DATA_0

Contains fuse data (first 32 bits). Valid only 200 µs after chip reset has been de-asserted.

Address: 0x000000 + 0xC0E
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	

11.25.2.37 FUSE_DATA_1

Contains fuse data (second 32 bits). Valid only 200 µs after chip reset has been de-asserted.

Address: 0x000000 + 0xC0F
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	

11.25.2.38 BIST_CTRL

Controls the memory BIST state machines, which can be used to activate the memory fill procedure or the memory BIST to check for faults. Each reset domain has associated control bits for *BistRun* (to activate the FSM) and *BistMode* (memory fill or memory BIST).

- *BistRun_**: Set to 1b to activate the BIST FSM. Must be cleared before starting normal operation.
- *BistMode_**: If cleared, *BistRun_** will start regular BIST. If set, *BistRun_** will start memory initialization.

Address: **0x000000 + 0xC10**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
BistRun_PCIE[0..8]	8:0	RW	0b	(9 x 1-bit)
BistRun_EPL	9	RW	0b	
BistRun_FABRIC	10	RW	0b	
BistRun_TUNNEL	11	RW	0b	
BistRun_BSM	12	RW	0b	
BistRun_CRM	13	RW	0b	
BistRun_FIBM	14	RW	0b	
BistRun_SBM	15	RW	0b	
Reserved	31:16	RSV	0x0	Reserved.
BistMode_PCIE[0..8]	40:32	RW	0x0	(9 x 1-bit)
BistMode_EPL	41	RW	0b	
BistMode_FABRIC	42	RW	0b	
BistMode_TUNNEL	43	RW	0b	
BistMode_BSM	44	RW	0b	
BistMode_CRM	45	RW	0b	
BistMode_FIBM	46	RW	0b	
BistMode_SBM	47	RW	0b	
Reserved	63:48	RSV	0x0	Reserved.

11.25.2.39 REI_CTRL

Controls the RAM EFUSE interface (REI) block. The REI needs to be taken out of reset before issuing the *Run* or *AutoLoadEnable* commands.

Address: **0x000000 + 0xC12**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Reset	0	RW	1b	REI reset, clear this bit to take REI out of reset.
Mode	1	RW	0b	0b = Read pattern and load into REI. 1b = Decompress pattern and load redundancy into SRAMs and TCAMs.
Run	2	RW	0b	Manually starts the REI operation chosen in the *Mode* setting.
AutoLoadEnable	3	RW	0b	If set REI, reads efuse, decompresses and loads the redundancy information into the SRAMs and TCAMs when taken out of reset.
Reserved	31:4	RSV	0x0	Reserved.

11.25.2.40 REI_STAT

The REI status signals are directly synchronized to this register. Once a field has been asserted, it is possible to de-assert it by resetting the REI module or by either setting REI_CTRL.*AutoLoadEnable* = 0b (when using AutoLoad) or setting REI_CTRL.*Run* = 0b (when operating manually).

Address: 0x000000 + 0xC13
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ReiDonePass	0	vRO	0b	REI operation successful.
ReiDoneFail	1	vRO	0b	REI operation failed.
Reserved	31:2	RSV	0x0	Reserved.

11.25.2.41 GPIO_CFG

Address: 0x000000 + 0xC15
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Dir	15:0	RW	0x0	Defines the direction of each GPIO pin: 0b = Input 1b = Output
OpenDrain	31:16	RW	0x0	Defines if the pin is configured as open drain or normal output pin. This configuration has no effect if the pin is configured as an input

11.25.2.42 GPIO_DATA

Address: 0x000000 + 0xC16
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
data	15:0	RW	0x0	GPIO pin state. If the pin is configured as open drain: 0b = Causes the pin to be driven to ground. 1b = Causes the pin to not be driven. ***Note:*** Writing causes the value to be latched into a 16-bit latch, while reading returns the actual pin state.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.43 GPIO_IP

Address:	**0x000000 + 0xC17**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
detectHigh	15:0	vCW1	0x0	Indicates if the pin has changed its value: 0b = No change. 1b = Changed from 0b to 1b. Writing a 1b into this register on the corresponding GPIO pin clears the interrupt.
detectLow	31:16	vCW1	0x0	Indicates if the pin has changed its value: 0b = No change. 1b = Changed from 1b to 0b. Writing a 1b into this register on the corresponding GPIO pin clears the interrupt.

11.25.2.44 GPIO_IM

The interrupt mask register controls if the corresponding interrupt source in GPIO_IP is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address:	**0x000000 + 0xC18**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
detectHighMask	15:0	RW	0xFFFF	Indicates if the a transition from 0b to 1b should cause an interrupt or should be masked. 0b = Interrupt 1b = Mask
detectLowMask	31:16	RW	0xFFFF	Indicates if the a transition from 1b to 0b should cause an interrupt or should be masked. 0b = Interrupt 1b = Mask

11.25.2.45 I2C_CFG

Address:	**0x000000 + 0xC19**			
Atomicity:	**32**			
Reset Domains:	**MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Enable	0	RW	1b	Controls I^{2}C operation. 0b = I^{2}C does not answer to I^{2}C commands from other masters. 1b = I^{2}C answers to I^{2}C commands from other masters targeting the address. The default value is active.
Addr	7:1	vRW	0x40	Defines the I^{2}C address of the switch on a data bus. The default configuration is '1000000'.
Divider	19:8	RW	0x64	Defines the clock rate as a divider of the PCIE_REFCLK. • Use 52 decimal (0x34h) for 100 KHz • Use 10 decimal (0xAh) for 400 KHz ***Note:*** The default divider is 100 decimal (0x64h) and results in slower operation than the normal conventional operating clock frequency.

Address: **0x000000 + 0xC19**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
InterruptMask	20	RW	1b	Defines if the I^{2}C interrupt is masked or passed to the interrupt hierarchy. 0b = Passed 1b = Masked
DebounceFilterCountLimit	27:21	RW	0x14	Controls the debouncing filter circuit.
Reserved	31:28	RSV	0x0	Reserved.

11.25.2.46 I2C_DATA[0..2]

Address: **0x000000 + 0x1*i + 0xC1C**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRW	0x0	Bytes to be sent out or read from the I^{2}C bus depending on the command type: read, write or write-read. Bytes are sent or received in the following order: • I2C_DATA[0].*Data*[31:24] • I2C_DATA[0].*Data*[23:16] • I2C_DATA[0].*Data*[15:8] • I2C_DATA[0].*Data*[7:0] • I2C_DATA[1].*Data*[31:24] • etc...

11.25.2.47 I2C_CTRL

Address: **0x000000 + 0xC20**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Addr	7:0	RW	0x0	Device to address. Bit 0 is not used and is completed by the command selected.
Command	9:8	RW	00b	Command type. The commands are: 00b = Null. 01b = Write. 10b = Write_then_read. 11b = Read. ***Note:*** A new command starts execution only when this field is changed from 0b to one of the possible commands.
LengthW	13:10	RW	0x0	Specifies the number of bytes, 0..12, to write to the I^{2}C bus for a 'write' or 'write-read'. This field is not used for a 'read' command.
LengthR	17:14	RW	0x0	Specifies the number of bytes to receive, 1..12, for a 'read' or 'write-read'. Length of 0 is not valid for a 'read' or 'write-read' command. This field is not used and ignored for a 'write' command and can be set to 0 in this case.
LengthSent	21:18	vRO	0x0	Indicates the number of bytes sent before a NAK was received. If the targeted device has returned ACK on all bytes sent, this field is equal to the LengthW requested.

Address: 0x000000 + 0xC20
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
CommandCompleted	25:22	vRO	0xF	Indicates the command status. This field is automatically cleared when the command is started. The codes returned are: 0000b = Running — The command is still running or the command is null. 0001b = Terminated_normally — All bytes requested were transfered. 0010b = Terminated_prematurely — The targeted device returned a NAK on data. 0011b = No_device — The command terminated prematurely because there was no ACK at the address phase. 0100b = Timeout — The attached device asserted the clock for longer than 2^30 reference clock cycles. 0101b = Lost_arbitration — Arbitration lost to another I^{2}C master. 0110b = Waiting_for_bus — Currently executing. 1111b = Invalid — Default value when no status is reported. All other values are reserved.
InterruptPending	26	vCW1	0b	Defines if a command has completed or not. Writing a 1b clears the interrupt.
Reserved	31:27	RSV	0x0	Reserved.

11.25.2.48 MDIO_CFG

Address: 0x000000 + 0xC22
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Divider	11:0	RW	0x19	MDIO Clock Divider. The MDIO clock is equal to PCIE_REFCLK/2/DIVIDER. Default is 2 MHz.
Preamble	12	RW	1b	Defines if the preamble (32 cycles of 1s) is sent or not. 0b = Not sent 1b = Sent The default is to send the preamble. Faster access is possible by canceling transmission of preamble.
InterruptMask	13	RW	1b	Defines if the MDIO interrupt is masked or passed to the interrupt hierarchy. 0b = Passed 1b = Masked
Reserved	31:14	RSV	0x0	Reserved.

11.25.2.49 MDIO_DATA

Address:	0x000000 + 0xC23
Atomicity:	32
Reset Domains:	MASTER

Field Name	Bit(s)	Type	Default	Description
Data	15:0	vRW	0x0	Data register. For the command 'read', the content of this register is only updated if the command completed successfully. For the command 'write', the content of the register is sent to the device (MSB first) and does not change once the command completes.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.50 MDIO_CTRL

Address:	0x000000 + 0xC24
Atomicity:	32
Reset Domains:	MASTER

Field Name	Bit(s)	Type	Default	Description
Register	15:0	RW	0x0	Register to address.
Device	20:16	RW	0x0	Sub-device to address.
PhyAddress	25:21	RW	0x0	Physical device to address.
Command	27:26	RW	0x0	Command type. The commands are: 00b = Null 01b = Write — The address frame is always sent. 10b = Sequential-Read — The address frame is not sent and the read frame causes the next register to be read. 11b = Random-Read — The address is sent before reading. ***Note:*** A new command starts execution only when this field is changed from 0b to one of the possible commands.
DeviceType	28	RW	0b	Defines the device type. 0b = Clause 22 — The MDIO frame structure is for legacy devices and does not contain a 'device' field. The register address is only 5 bits. 1b = Clause 45 — The MDIO frame structure is for high-speed devices and contains a 'device' field. The register address is 16 bits.
Status	30:29	vRO	01b	Indicates the command status. This field is automatically cleared when the command is started. The codes returned are listed below: 00b = Running — The command still running. 01b = Terminated normally — All bytes requested were transfered. 10b = No device — The command terminated prematurely because there was no ACK at the address phase (only read or write-read may have this status) 11b = Reserved.
InterruptPending	31	vCW1	0b	Indicates if a command has been completed. Writing 1b clears the interrupt.

11.25.2.51 SPI_TX_DATA

Contains the data to send to attached SPI device.

Address:	**0x000000 + 0xC26**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Data is stored [*ShiftSize**8-1:0] and sent with MSB first.

11.25.2.52 SPI_RX_DATA

Contains the data received from attached SPI device.

Address:	**0x000000 + 0xC27**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	Data is stored [*ShiftSize**8-1:0] and sent with MSB first.

11.25.2.53 SPI_HEADER

Contains the header to send to the device. Typically include a one byte instruction and possibly a 3-byte address.

Address:	**0x000000 + 0xC28**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Data is stored [*HeaderSize**8-1:0] and sent with MSB first.

11.25.2.54 SPI_CTRL

Control access to SPI device. All values can be set simultaneously with the command.

Address:	**0x000000 + 0xC29**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Freq	9:0	RW	0x31	Defines the clock divider to be used. Actual speed is PCIE_REFCLK/(2*(1+Freq)). Default is 1 MHz.
Enable	10	RW	0b	Defines if the SPI controller is enabled or not. Command is ignored if the controller is not enabled.
Command	14:11	RW	0x0	Defines the command to execute. The command is a bit field with the following bits defined: Bit[0] = Send header or not. Bit[1] = Send turn-around-byte or not. Bit[2] = Shift data. Bit[3] = Release chip select. The command must be reset to 0b for another command to be executed.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0xC29** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
HeaderSize	16:15	RW	00b	Defines the number of bytes in the header. Zero means 4 bytes. The value could be set simultaneously with the command.
DataSize	18:17	RW	00b	Defines the number of bytes in the header. Zero means 4 bytes. The value could be set simultaneously with the command.
DataShiftMethod	20:19	RW	00b	Defines the access method for data. 00b = SINGLE 01b = DUAL 10b = QUAD 11b = Reserved The value could be set simultaneously with the command.
Busy	21	vRO	0b	Set while a command is executed. Cleared when the command is completed.
Selected	22	vRO	0b	Set if chip is currently selected.
DirIO2	23	RW	0b	Set direction for SPI_IO2. 0b = Input 1b = Output This register gets override when operating in or QUAD-pin mode.
DirIO3	24	RW	0b	Set direction for SPI_IO3. 0b = Input 1b = Output This register gets override when operating in QUAD-pin mode.
PinIO2	25	vRW	0b	Read or set pins SPI_IO2. Reading always returns the actual input pins. Writing sets the pin, but this might be ignored if the pin is an input or the device is operating in QUAD-pin mode.
PinIO3	26	vRW	0b	Read or set pins SPI_IO3. Reading always returns the actual input pins. Writing sets the pin, but this might be ignored if the pin is an input or the device is operating in QUAD-pin mode.
Reserved	31:27	RSV	0x0	Reserved.

11.25.2.55 LED_CFG

This register defines the behavior of the LED interface.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0xC2B** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LEDFreq	23:0	RW	0x1E848	Divider of PCIE_REFCLK to produce LED_CLK (default 8 KHz).
LEDEnable	24	RW	0b	Indicates if LED's are enabled or not.
Reserved	31:25	RSV	0x0	Reserved.

11.25.2.56 SCAN_DATA_IN

Writing to this register will automatically initiate a request to SCAN_TOP.

Address: 0x000000 + 0xC2D
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ScanData	24:0	RW	0x0	Data shifted into the scan chains.
ShiftIn	25	RW	0b	S2A valid bit value.
ShiftOut	26	RW	0b	A2S enable bit value.
UpdateNodes	27	RW	0b	If set, the values in the 4 ChanDft rails is updated.
Inject	28	RW	0b	Valid of the ChanDft inject rail.
Drain	29	RW	0b	Valid of the ChanDft drain rail.
Passthru	30	RW	0b	Valid of the ChanDft passthru rail.
Single	31	RW	0b	Valid of the ChanDft single rail.

11.25.2.57 CRM_DATA[0..2047][0..1]

Address: 0x000000 + 0x2*j + 0x1*i + 0x1000
Atomicity: 32
Reset Domains: COLD_MEM

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRW	0x0	Data storage for the commands. The usage depends on the command.

11.25.2.58 CRM_CTRL

Address: 0x000000 + 0x2000
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Run	0	vRW	0b	Start or stop the CRM. Stopping may requires some time as the current command as to complete
FirstCommandIndex	6:1	vRW	0x0	Index of first command of the sequence.
LastCommandIndex	12:7	vRW	0x0	Index of last command of the sequence
ContinuousRun	13	RW	0b	Defines if the execution sequence of a command. 0b = Executed only once 1b = Executed continuously
TickPrescale	18:14	RW	0x0	Defines (in power of 2) the prescaler to apply to the PCIE_REFCLK to produce the reference time for the CRM. Valid range is [0..16].
TimeoutPrescale	23:19	RW	0x0	Defines (in power of 2) the maximum time in PCIE_REFCLK clock period allowed for the CRM to complete a read/write request to any register. Valid range is [0..19]. An alarm is sent to the watchdog if this time is exceeded. A value of 0b disables the feature. Used as protection against soft error.
Reserved	31:24	RSV	0x0	Reserved.

11.25.2.59 CRM_STATUS

Address:	**0x000000 + 0x2001**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Running	0	vRO	0b	Indicates actual state of CRM. 0b = Stopped 1b = Running ***Note:*** Stopping the CRM (turning 'Run' bit to 0b) may be delayed until the current command being executed completes.
CommandIndex	6:1	vRW	0x0	If stopped, this is the index to the next command to execute. The value must be set to any command in the sequence before starting the CRM. If running, this is the index of the current command being executed and updated by the switch. It must not be changed while the CRM is still active.
Reserved	31:7	RSV	0x0	Reserved.

11.25.2.60 CRM_TIME

Address:	**0x000000 + 0x2002**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Tick	31:0	vCW1	0x0	Updated by switch to define time base. Can be reset to 0b by software.

11.25.2.61 CRM_SRAM_CTRL

Counter Rate Monitor memory error injection control and BIST status reporting register.

- [0] => CRM_DATA[..][0]
- [1] => CRM_DATA[..][1]
- [2] => CRM_REGISTER(LOW)
- [3] => CRM_REGISTER(HIGH)
- [4] => CRM_PERIOD(LOW)
- [5] => CRM_PERIOD(HIGH)
- [6] => CRM_PARAM

Address: 0x000000 + 0x2004
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits) For error injection. 00b = Normal write operations. 01b = Generate a correctable error. 10b = Generate a correctable error. 11b = Generate an uncorrectable error.
CErr[0..6]	20:14	vCW1	0b	(7 x 1-bit) Correctable errors.
UErr[0..6]	27:21	vCW1	0b	(7 x 1-bit) Uncorrectable errors.
BistDonePass[0..6]	34:28	RO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	RO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.25.2.62 CRM_IP

Counter Rate Monitor IP registers.

Address: 0x000000 + 0x2008
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
InterruptPending[0..63]	63:0	vCW1	0b	(64 x 1-bit) Bit set when the corresponding command has either completed the operation (memory copy or memory set) or has recorded new information.
SramErr[0..1]	65:64	vCW1	0b	(2 x 1-bit) Set when a memory error is detected in CRM. Bits correspond to memory error type: Bit 0 = Correctable Bit 1 = Uncorrectable See register CRM_SRAM_CTRL to determine source of error.
Reserved	127:66	RSV	0x0	Reserved.

11.25.2.63 CRM_IM

Counter Rate Monitor IM registers.

The interrupt mask register controls whether the corresponding interrupt source in CRM_IP is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0x000000 + 0x200C
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
InterruptMask[0..63]	63:0	RW	1b	(64 x 1-bit)

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x200C** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SramErr[0..1]	65:64	RW	1b	(2 x 1-bit)
Reserved	127:66	RSV	0x0	Reserved.

11.25.2.64 CRM_COMMAND[0..63]

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2*i + 0x2080** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Command	2:0	RW	000b	Defines the command to execute: 000b = Set 001b = Copy 010b = CountRateTooFast 011b = MonitorChange 100b = SaveMax 101b = CountRateTooSlow 110b = CountGreaterThan 111b = Checksum
DataIndex	13:3	RW	0x0	Pointer of the data section for this command. Not used for all commands.
Count	33:14	RW	0x0	Defines the number of registers walk through by this command. If the command needs a data section, the *Count* is limited to 2048 maximum (total size of data set). If the command does not need a data section (as by example for memory set), the limit is 1,048,575. The count is the number of registers, not the number of bytes or words.
Reserved	63:34	RSV	0x0	Reserved.

11.25.2.65 CRM_REGISTER[0..63]

Defines the registers location, size and strides for the corresponding command.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2*i + 0x2100** **32** **COLD_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BaseAddress	23:0	RW	0x0	First register address.
Size	25:24	RW	0x0	Defines register size: 00b = 32 01b = 64 10b = 96 11b = 128
BlockSize1Shift	29:26	RW	0x0	Defines register block size in power of 2. Block size is 1<<BlockSize1Shift.
Stride1Shift	33:30	RW	0x0	Defines stride to next block in power of 2. Stride is 1<<(Stride1Shift+1)
BlockSize2Shift	37:34	RW	0x0	Defines second level register block size in power of 2. Block size is 1<<BlockSize2Shift.

Address: 0x000000 + 0x2*i + 0x2100
Atomicity: 32
Reset Domains: COLD_MEM

Field Name	Bit(s)	Type	Default	Description
Stride2Shift	41:38	RW	0x0	Defines second level stride to next block in power of 2. Stride is 1<<(Stride2Shift+1)
Reserved	63:42	RSV	0x0	Reserved.

11.25.2.66 CRM_PERIOD[0..63]

Address: 0x000000 + 0x2*i + 0x2180
Atomicity: 32
Reset Domains: COLD_MEM

Field Name	Bit(s)	Type	Default	Description
Interval	31:0	RW	0x0	Defines time interval before executing this command. The timebase ('CRM tick') is a prescale PCIE_REFCLK (1 to 65536 in power of 2). A value of 0b is as fast as possible.
LastTick	63:32	vRW	0x0	Defines last time the command was executed.

11.25.2.67 CRM_PARAM[0..63]

Address: 0x000000 + 0x1*i + 0x2200
Atomicity: 32
Reset Domains: COLD_MEM

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Parameters for commands.

11.25.2.68 PLL_PCIE_CTRL

Address: 0x000000 + 0x2241
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Nreset	0	RW	1b	PLL reset enable. 0b = Places PLL in reset. 1b = Normal operation. Should be set 1b upon power up until reset occurs.
Enable	1	RW	1b	PLL loop control block enable. 0b = Enable. 1b = Normal operation.
Halt	2	RW	0b	CCB halt control. 0b = Normal functional operation. 1b = Gate the PLL clock output.
RefDiv	8:3	RW	0x1	Reference clock divider by any integer from 1 to 63 inclusive. Programming input may NOT be changed on the fly.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2241** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FbDiv4	9	RW	0b	Pre-scalar for feedback divider. 0b = Divide-by-2 1b = Divide-by-4 Programming input may NOT be changed on the fly.
FbDiv255	17:10	RW	0x19	PLL Feedback Divider main divisor. Divides clock fed into the PLL Control Block by any integer from 2 to 255 inclusive. Programming input may NOT be changed on the fly.
OutDiv	23:18	RW	0xA	PLL Output Divider divisor. Any integer from 2 to 63 inclusive is valid. Programming input may be changed on the fly.
OutMuxSel	26:24	RW	000b	PLL output clock mux select. Defaults to reference clock 000b = PCIE_REFCLK 001b = PLL_PCIE_CLK 010b = BURN_IN_CLK All other values are reserved.
Reserved	31:27	RSV	0x0	Reserved.

11.25.2.69 PLL_PCIE_STAT

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2242** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PllLocked	0	vRO	0b	Indicates if the PLL has locked or not.
PllFreqChange	1	vRO	0b	Toggles to indicate a change in the PLL output clock frequency.
MiscCtrl	9:2	RW	0x01	Connected to the PLL Spare pins that affect coarse calibration and on-the-fly output dividers. Bit[0] = Enables Fast Calibration Mode when set to 1b. Bits[1:3] = Additional spare pins. Bit[4] = Asynchronous load signal for PLL output divider. Bit[5] = Reserved. Must be 0b. Bits[6:7] = Sets the initial value for coarse thermal bits. Used to reduce the PLL calibration time.
Reserved	31:10	RSV	0x0	Reserved.

11.25.2.70 SBUS_PCIE_CFG

Configuration register for the SBUS controller.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2243** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SBUS_ControllerReset	0	RW	1b	Defines if SBUS controller is in reset. 0b = Not in reset. 1b = In reset.
RomEnable	1	RW	0b	Enables loading of the on-chip ROM onto the SBus during boot-up

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2243** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RomBusy	2	vRO	0b	Indicates if ROM loading is in progress. Requests to the SBus from the core are placed on hold until ROM loading is complete.
BistDonePass	3	vRO	0b	Indicates SBUS controller memory BIST status.
BistDoneFail	4	vRO	0b	Indicates SBUS controller memory BIST status.
Reserved	31:5	RSV	0x0	Reserved.

11.25.2.71 SBUS_PCIE_COMMAND

Access to serial bus registers.

Writing into this register creates a command (read or write) to be issued on the serial bus linking all SerDes, SBUS itself and SPICO micro-controller.

The command execution started only when *Execute* is changed from 0b to 1b. The command execution may take up to 4 µs to execute. The command is thus latched and the control is returned to the processor immediately while the command is being executed. The *Busy* field is automatically set when the command is started and remains asserted while a command is being executed. Any write to this register is ignored until the *Busy* is returned to 0b.

A new command starts only when the *Execute* bit is first reset to 0b and then set to 1b. *Register*, *Address*, *Op* and *Execute* can all be set jointly on the same register write. If command is a write, SBUS_PCIE_REQUEST must be been set before the *Execute* bit is set.

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2244** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Register	7:0	RW	0x0	Data register to access.
Address	15:8	RW	0x0	Receiver device address to access.
Op	23:16	RW	0x0	Operation.
Execute	24	RW	0b	Starts execution of command. The command is started only when this bit is changed from 0b to 1b. This field could be written at the same time as *Address*/*Register*/*Op* to start a new command.
Busy	25	RO	0b	Indicates if a command is being executed. Is set automatically when a command is started, and is cleared automatically when the command execution is completed.
ResultCode	28:26	RO	000b	Returns SBUS result code of last transaction.
Reserved	31:29	RSV	000b	Reserved.

11.25.2.72 SBUS_PCIE_REQUEST

Data sent with command. Data should only be changed while no command are being executed.

Address: 0x000000 + 0x2245
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Data to write.

11.25.2.73 SBUS_PCIE_RESPONSE

Data value returned from a read command. Data is not valid while a command is being executed.

Address: 0x000000 + 0x2246
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	Data to read.

11.25.2.74 SBUS_PCIE_SPICO_IN

Access to the SPICO inside the SBUS master.

Address: 0x000000 + 0x2247
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Interrupt	0	RW	0b	Interrupts SPICO controller.
GeneralPurposeIn	16:1	RW	0x0	General purpose 16-bit passed by host to SPICO.
Reserved	31:17	RSV	0x0	Reserved.

11.25.2.75 SBUS_PCIE_SPICO_OUT

Response from the SPICO inside the SBUS master.

Address: 0x000000 + 0x2248
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
GeneralPurposeOut	15:0	vRO	0x0	General purpose 16-bit passed by SPICO to host.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.76 SBUS_PCIE_IP

Interrupt register for SBUS.

Address:	**0x000000 + 0x2249**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
DataDetectHigh	15:0	vCW1	0x0	Detected transition from 0b to 1b on *GeneralPurposeOut* register.
DataDetectLow	31:16	vCW1	0x0	Detected transition from 1b to 0b on *GeneralPurposeOut* register.

11.25.2.77 SBUS_PCIE_IM

Interrupt mask register for SBUS.

Address:	**0x000000 + 0x224A**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
MaskDataDetectHigh	15:0	RW	0xFFFF	Mask detection of transition from 0b to 1b on *GeneralPurposeOut* register.
MaskDataDetectLow	31:16	RW	0xFFFF	Mask detection of transition from 1b to 0b on *GeneralPurposeOut* register.

11.25.2.78 SYSTIME_CFG

Configuration for the SYSTIME block (used for IEEE-1588 support).

Address:	**0x000000 + 0x224C**
Atomicity:	**32**
Reset Domains:	**COLD**

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
STEP	7:4	RW	0xA	Defines the STEP for each clock. Minimum is 2, maximum is 15.
CaptureHigh[0..3]	11:8	RW	0b	(4 x 1-bit) Capture time in SYSTIME_CAPTURE at low to high transition of GPIO[0..3]
CaptureLow[0..3]	15:12	RW	0b	(4 x 1-bit) Capture time in SYSTIME_CAPTURE at high to low transition of GPIO[0..3]
Reserved	23:16	RSV	0x0	Reserved.
Adjust	61:24	RW	0x0	Number of PLUS/MINUS adjustments per 2^48 cycles of SYSTIME refclock.
Reserved	62	RSV	0b	Reserved.
Dir	63	RW	0b	Direction of adjustments (+1 or -1).

11.25.2.79 SYSTIME

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x224E** **32** **COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CurrentTime	63:0	vRO	0x0	

11.25.2.80 SYSTIME0

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2250** **32** **COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Time0	63:0	RW	0x0	

11.25.2.81 SYSTIME_PULSE[0..1]

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x1*i + 0x2252** **32** **COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Period	31:0	RW	0x0	Counts the number of SYSTIME ticks before generating a pulse. The pulse out is always 8 SYSTIME refclock cycles. This register should be configured either to 0, or to a value at least 16*SYSTIME_CFG.*STEP*. A configuration of 0 causes the pulse output to be held constant low.

11.25.2.82 SYSTIME_CAPTURE[0..3]

Captured time on transition of GPIO[0..3], the transition captured depends on setting of SYTIME_CFG.*CaptureHigh*/*CaptureLow* (none, high to low, low to high or both).

Address: **Atomicity:** **Reset Domains:**	**0x000000 + 0x2*i + 0x2258** **64** **COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SampleTime	63:0	vRO	0x0	Time at transition.

11.25.2.83 PCIE_XPLL_CTRL

Shared PCIE_XREFCLK PLLs configuration.

Address: 0x00000 + 0x3000
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
RefDiv	5:0	RW	0x1	Reference clock divider by any integer from 1 to 63 inclusive. Programming input may NOT be changed on the fly.
FbDiv4	6	RW	0b	Pre-scalar for feedback divider. 0b = Divide-by-2 1b = Divide-by-4 Programming input may NOT be changed on the fly.
FbDiv255	14:7	RW	0x19	PLL Feedback Divider main divisor. Divides clock fed into the PLL control block by any integer from 2 to 255 inclusive. Programming input may NOT be changed on the fly.
OutDiv	20:15	RW	0xA	PLL Output Divider divisor. Any integer from 2 to 63 inclusive is valid. Programming input may be changed on the fly.
MiscCtrl	28:21	RW	0x01	Connected to the PLL spare pins that affect coarse calibration and on-the-fly output dividers. Bit[0] = Enables Fast Calibration Mode when set to 1. Bit[1:3] = Additional spare pins. Bit[4] = Asynchronous load signal for PLL output divider. Bit[5] = Reserved, must be 0. Bit[6:7] = Sets the initial value for coarse thermal bits. Used to reduce the PLL calibration time
Reserved	31:29	RSV	0x0	Reserved.

11.25.2.84 PCIE_CLK_CTRL

Per PCIE_XREFCLK PLLs control registers.

Address: 0x00000 + 0x3001
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Nreset[0..3]	3:0	RW	0x0	(4 x 1-bit) 0b = Places PLL in reset, 1b = Normal functional operation. Should be set 1b upon power up until reset occurs.
Enable[0..3]	7:4	RW	0x1	(4 x 1-bit) PLL loop control block enable. Set to 1b for normal functional operation.
Halt[0..3]	11:8	RW	0x0	(4 x 1-bit) CCB halt control. 0b = Gates the PLL clock output. 1b = Normal functional operation.

Address: **0x00000 + 0x3001**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
OutMuxSel[0..3]	19:12	RW	0x0	(4 x 2-bits) PLL output clock mux select. Defaults to PLL_PCIE mux output. 0 = PLL_PCIE_MUX_OUT — Same as clock source selected by PLL_PCIE_CTRL.*OutMuxSel*. 1 = XPLL_PCIE_CLK — XPLL output clock.
Mode[0..3]	31:20	RW	0x0	(4 x 3-bits) Define the control mode for selecting reference clock to XPLL. 0 = PCIE_REFCLK — Always select PCIE_REFCLK, reset and XREFCLK validity are ignored. 1 = PCIE_XREFCLK — Always select PCIE_XREFCLK, reset and XREFCLK validity are ignored. 2 = CHECK_VALID — Use PCIE_XREFCLK if reset de-asserted and if the clock is valid. Otherwise use PCIE_REFCLK. 3 = CHECK_RESET — Use PCIE_XREFCLK if reset de-asserted. Otherwise, use PCIE_REFCLK. Ignore PCIE_XREFCLK validity.

11.25.2.85 PCIE_CLK_CTRL2

Per PCIE_XREFCLK PLLs configuration registers (continued).

Address:	**0x00000 + 0x3002**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
XclkTerm[0..3]	3:0	RW	0x0	(4 x 1-bit) Termination setting for each PCIE_XREFCLK pin. Set to 1b to enable on-chip termination.
ClkObs	7:4	RW	0x0	Mux select value for the output signal to the CLKOUT_C clock observation pad. If LSM_CLKOBS_CTRL.*SelC* is set to 14, the following outputs are selected: 0 = PCIE_XREFCLK[0] 1 = PCIE_XREFCLK[1] 2 = PCIE_XREFCLK[2] 3 = PCIE_XREFCLK[3] 4 = CLKMUX_OUT[0] 5 = CLKMUX_OUT[1] 6 = CLKMUX_OUT[2] 7 = CLKMUX_OUT[3] 8 = XPLL[0] clock output 9 = XPLL[1] clock output 10 = XPLL[2] clock output 11 = XPLL[3] clock output 12 = xlock[0] 13 = xlock[1] 14 = xlock[2] 15 = xlock[3] If LSM_CLKOBS_CTRL.SelC is set to 15, the following outputs are selected: 0 = xclkval[0] 1 = xclkval[1] 2 = xclkval[2] 3 = xclkval[3] 4 = XPLL_CCB_OUT[0] 5 = XPLL_CCB_OUT[1] 6 = XPLL_CCB_OUT[2] 7 = XPLL_CCB_OUT[3]
Reserved	31:8	RSV	0x0	Reserved.

11.25.2.86 PCIE_CLKMON_RATIO_CFG

Clock monitor configuration register.

Address:	**0x00000 + 0x3003**
Atomicity:	**32**
Reset Domains:	**MASTER**

Field Name	Bit(s)	Type	Default	Description
Denom	15:0	RW	0x5	Denominator for clock ratio between reference clock and PCIE_PLL output. Supported values are 5, 999, and 1999.
Num	31:16	RW	0x1	Numerator for clock ratio between reference clock and PCIE_PLL output. Supported values are 1, 200, and 400.

11.25.2.87 PCIE_CLKMON_TOLERANCE_CFG

Clock monitor configuration register.

Address: 0x00000 + 0x3004
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
NegTol	15:0	RW	0xA	Largest allowed negative deviation of clock accumulator. The register is interpreted as a positive number.
PosTol	31:16	RW	0xF	Largest allowed positive deviation of clock accumulator.

11.25.2.88 PCIE_CLKMON_DEADLINES_CFG

Clock monitor configuration register.

Address: 0x00000 + 0x3005
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Sustain	15:0	RW	0x3E8	Time period in which to check tolerance Expressed in terms of 2 ns clock cycles. Setting to 0 disables checking long term tolerance as a validity condition.
SilentLimit	31:16	RW	0xA	The maximum amount of time before declaring the PCIE_REFCLK silent. Expressed in terms of 2 ns clock cycles. Setting to 0 disables checking short term tolerance as a validity condition.

11.25.2.89 PCIE_CLK_STAT

Per PCIE_XREFCLK PLLs status.

Address: 0x00000 + 0x3006
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
PllLocked[0..3]	3:0	vRO	0x0	(4 x 1-bit) Indicates if the XPLLs have locked or not. ***Note:*** This status reflects the XPLL state, not the PCIE_XREFCLK clocks validity. See PCIE_CLK_CTRL.*Mode* (Section 11.25.2.84) to determine what input is presented to XPLL. If PCIE_XREFCLK is outside of bound, the XPLL might still lock. Use *XRefclkValid* to check if clock is valid.
PllFreqChange[0..3]	7:4	vRO	0x0	(4 x 1-bit) Toggles to indicate a change in the XPLL outputs clock frequency.
RefclkSel[0..3]	11:8	vRO	0x0	(4 x 1-bit) Indicates which clock is selected on XCLKMUX.
XRefclkValid[0..3]	15:12	vRO	0x0	(4 x 1-bit) Indicates which XREFCLK is valid.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.90 PCIE_CLK_IP

Interrupt pending for PCIE_XREFCLK PLLs.

Address: **0x00000 + 0x3007**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
XRefClkValidChangeHigh[0..3]	3:0	vCW1	0x0	(4 x 1-bit) Detected transition from 0 to 1.
XRefClkValidChangeLow[0..3]	7:4	vCW1	0x0	(4 x 1-bit) Detected transition from 1 to 0.
XPllLockChangeHigh[0..3]	11:8	vCW1	0x0	(4 x 1-bit) Detected transition from 0 to 1.
XPllLockChangeLow[0..3]	15:12	vCW1	0x0	(4 x 1-bit) Detected transition from 1 to 0.
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.91 PCIE_CLK_IM

Interrupt mask for PCIE_XREFCLK PLLs.

Address: **0x00000 + 0x3008**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
XRefClkValidChangeHigh[0..3]	3:0	RW	0x1	(4 x 1-bit)
XRefClkValidChangeLow[0..3]	7:4	RW	0x1	(4 x 1-bit)
XPllLockChangeHigh[0..3]	11:8	RW	0x1	(4 x 1-bit)
XPllLockChangeLow[0..3]	15:12	RW	0x1	(4 x 1-bit)
Reserved	31:16	RSV	0x0	Reserved.

11.25.2.92 PCIE_WARM_RESET_DELAY

Delay between xrefclk selection and PCIe warm reset de-assertion.

Address: **0x00000 + 0x3009**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
DelayCount	15:0	RW	0x64	Each increment is 100 PCIE_REFCLK cycles (nominally 1 µs). This delay is only enforced when the corresponding PCIE_CLK_CTRL.*Mode*[i] is set to 2 or 3. The delay setting is shared among all PCIe warm reset signals.
Reserved	31:16	RSV	0x0	Reserved.

11.26 PORTS_MGMT Registers Description

11.26.1 PORTS_MGMT Map

Table 11-38 Ports Management Registers (reserved 16 KW) Map

Name	Address (Base = 0x0E8000)	Atomicity	Brief Description
PLL_EPL_CTRL	Base + 0x0	32	PLL configuration register.
PLL_EPL_STAT	Base + 0x1	32	PLL status register.
PLL_FABRIC_CTRL	Base + 0x2	32	PLL configuration register.
PLL_FABRIC_STAT	Base + 0x3	32	PLL status register.
PLL_FABRIC_LOCK	Base + 0x4	32	PLL lock register.
SBUS_EPL_CFG	Base + 0x5	32	SBUS Controller configuration.
SBUS_EPL_COMMAND	Base + 0x6	32	SBUS Command Execution.
SBUS_EPL_REQUEST	Base + 0x7	32	SBUS Data Write Request.
SBUS_EPL_RESPONSE	Base + 0x8	32	SBUS Data Read Response.
SBUS_EPL_SPICO_IN	Base + 0x9	32	
SBUS_EPL_SPICO_OUT	Base + 0xA	32	
SBUS_EPL_IP	Base + 0xB	32	
SBUS_EPL_IM	Base + 0xC	32	
ETHCLK_CFG[0..1]	Base + 0x1*i + 0xE	32	
ETHCLK_RATIO[0..1]	Base + 0x1*i + 0x10	32	
PM_CLKOBS_CTRL	Base + 0x12	32	

11.26.2 PORTS_MGMT Registers

11.26.2.1 PLL_EPL_CTRL

Address: Atomicity: Reset Domains:	0x0E8000 + 0x0 32 MASTER			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Nreset	0	RW	1b	PLL reset enable. 0b = Places PLL in reset. 1b = Normal operation. Should be set 1b upon power up until reset occurs.
Enable	1	RW	1b	PLL loop control block enable. 0b = Enable. 1b = Normal operation.
Halt	2	RW	0b	CCB halt control. 0b = Normal functional operation. 1b = Gate the PLL clock output.

Address: **Atomicity:** **Reset Domains:**	**0x0E8000 + 0x0** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RefDiv	8:3	RW	0x3	Reference clock divider by any integer from 1 to 63 inclusive. Programming input may NOT be changed on the fly.
FbDiv4	9	RW	0b	Pre-scalar for feedback divider. 0b = Divide-by-2 1b = Divide-by-4 Programming input may NOT be changed on the fly.
FbDiv255	17:10	RW	0x2E	PLL Feedback Divider main divisor. Divides clock fed into the PLL Control Block by any integer from 2 to 255 inclusive. Programming input may NOT be changed on the fly.
OutDiv	23:18	RW	0x6	PLL Output Divider divisor. Any integer from 2 to 63 inclusive is valid. Programming input may be changed on the fly. Default values for *RefDiv*,*FbDiv4*,*FbDiv255*,*OutDiv* generates 798.6 MHz.
OutMuxSel	26:24	RW	000b	PLL output clock mux select. Defaults to reference clock 000b = ETH_REFCLK 001b = PLL_EPL_CLK 010b = BURN_IN_CLK All other values are reserved.
Reserved	31:27	RSV	0x0	Reserved.

11.26.2.2 PLL_EPL_STAT

Address: **Atomicity:** **Reset Domains:**	**0x0E8000 + 0x1** **32** **MASTER**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PllLocked	0	vRO	0b	Indicates if the PLL has locked or not.
PllFreqChange	1	vRO	0b	Toggles to indicate a change in the PLL output clock frequency.
MiscCtrl	9:2	RW	0x01	Connected to the PLL Spare pins that affect coarse calibration and on-the-fly output dividers. Bit[0] = Enables Fast Calibration Mode when set to 1b. Bits[1:3] = Additional spare pins. Bit[4] = Asynchronous load signal for PLL output divider. Bit[5] = Reserved. Must be 0b. Bits[6:7] = Sets the initial value for coarse thermal bits. Used to reduce the PLL calibration time.
Reserved	31:10	RSV	0x0	Reserved.

11.26.2.3 PLL_FABRIC_CTRL

Address: **0x0E8000 + 0x2**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
Nreset	0	RW	1b	PLL reset enable. 0b = Places PLL in reset. 1b = Normal operation. Should be set 1b upon power up until reset occurs.
Enable	1	RW	1b	PLL loop control block enable. 0b = Enable. 1b = Normal operation.
Halt	2	RW	0b	CCB halt control. 0b = Normal functional operation. 1b = Gate the PLL clock output.
RefDiv	8:3	RW	0x3	Reference clock divider by any integer from 1 to 63 inclusive. Programming input may NOT be changed on the fly.
FbDiv4	9	RW	0b	Pre-scalar for feedback divider. 0b = Divide-by-2 1b = Divide-by-4 Programming input may NOT be changed on the fly.
FbDiv255	17:10	RW	0x2F	PLL Feedback Divider main divisor. Divides clock fed into the PLL Control Block by any integer from 2 to 255 inclusive. Programming input may NOT be changed on the fly.
OutDiv	23:18	RW	0x7	PLL Output Divider divisor. Any integer from 2 to 63 inclusive is valid. Programming input may be changed on the fly. Default values for *RefDiv*,*FbDiv4*,*FbDiv255*,*OutDiv* generates 699.4 MHz.
OutMuxSel	26:24	RW	000b	PLL output clock mux select. Defaults to reference clock 000b = ETH_REFCLK 001b = PLL_FABRIC_CLK 010b = BURN_IN_CLK All other values are reserved.
Reserved	31:27	RSV	0x0	Reserved.

11.26.2.4 PLL_FABRIC_STAT

Address: **0x0E8000 + 0x3**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
PllLocked	0	vRO	0b	Indicates if the PLL has locked or not.
PllFreqChange	1	vRO	0b	Toggles to indicate a change in the PLL output clock frequency.

Address: **0x0E8000 + 0x3**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
MiscCtrl	9:2	RW	0x01	Connected to the PLL Spare pins that affect coarse calibration and on-the-fly output dividers. Bit[0] = Enables Fast Calibration Mode when set to 1b. Bits[1:3] = Additional spare pins. Bit[4] = Asynchronous load signal for PLL output divider. Bit[5] = Asynchronous load signal for DLL output divider. Bits[6:7] = Sets the initial value for coarse thermal bits. Used to reduce the PLL calibration time.
Reserved	31:10	RSV	0x0	Reserved.

11.26.2.5 PLL_FABRIC_LOCK

Address: **0x0E8000 + 0x4**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
FeatureCode	3:0	SW1	0x0	Some SKUs support a restricted set of frequencies. Only writes with higher values than the existing *FeatureCode* take effect. Options are: 0000b = FULL — All frequencies supported. 0001b = LIMITED0 — Restricted to 600, 500, 400, 300 MHz. 0010b = LIMITED1 — Restricted to 500, 400, 300 MHz. 0011b = LIMITED2 — Restricted to 400, 300 MHz. 0100b = LIMITED3 — Restricted to 300 MHz. All other values are reserved.
FreqSel	7:4	RW	0x0	Selects which frequency to use for the fabric: 0000b = USE_CTRL — Use PLL_FABRIC_CTRL's values to set frequency, only supported when 0001b = F600 — 600 MHz if supported. 0010b = F500 — 500 MHz if supported. 0011b = F400 — 400 MHz if supported. 0100b = F300 — 300 MHz if supported. All other values are reserved.
Reserved	31:8	RSV	0x0	Reserved.

11.26.2.6 SBUS_EPL_CFG

Configuration register for the SBUS controller.

Address: **0x0E8000 + 0x5**
Atomicity: **32**
Reset Domains: **MASTER**

Field Name	Bit(s)	Type	Default	Description
SBUS_ControllerReset	0	RW	1b	Defines if SBUS controller is in reset. 0b = Not in reset. 1b = In reset.
RomEnable	1	RW	0b	Enables loading of the on-chip ROM onto the SBus during boot-up.

Address: 0x0E8000 + 0x5 **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RomBusy	2	vRO	0b	Indicates if ROM loading is in progress. Requests to the SBus from the core are placed on hold until ROM loading is complete.
BistDonePass	3	vRO	0b	Indicates SBUS controller memory BIST status.
BistDoneFail	4	vRO	0b	Indicates SBUS controller memory BIST status.
Reserved	31:5	RSV	0x0	Reserved.

11.26.2.7 SBUS_EPL_COMMAND

Access to serial bus registers.

Writing into this register creates a command (read or write) to be issued on the serial bus linking all SerDes, SBUS itself and SPICO micro-controller.

The command execution starts only when *Execute* is changed from 0b to 1b. The command execution may take up to 4 µs to execute. The command is thus latched and the control is returned to the processor immediately while the command is being executed. The *Busy* field is automatically set when the command is started and remains asserted while a command is being executed. Any write to this register is ignored until the *Busy* is returned to 0b.

A new command starts only when the *Execute* bit is first reset to 0b and then set to 1b. *Register*, *Address*, *Op* and *Execute* can all be set jointly on the same register write. If command is a write, SBUS_EPL_REQUEST must be been set before the *Execute* bit is set.

Address: 0x0E8000 + 0x6 **Atomicity:** 32 **Reset Domains:** MASTER				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Register	7:0	RW	0x0	Data register to access.
Address	15:8	RW	0x0	Receiver device address to access.
Op	23:16	RW	0x0	Operation.
Execute	24	RW	0b	Starts execution of command. The command is started only when this is bit is changed from 0b to 1b. This field could be written at the same time as *Address*/*Register*/*Op* to start a new command.
Busy	25	vRO	0b	Indicates if a command is being executed. Set automatically when a command is started, and is cleared automatically when the command execution is completed.
ResultCode	28:26	vRO	000b	Returns SBUS result code of last transaction.
Reserved	31:29	RSV	000b	Reserved.

11.26.2.8 SBUS_EPL_REQUEST

Data sent with command. Data should only be changed while no command are being executed.

Address: 0x0E8000 + 0x7
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Data to write.

11.26.2.9 SBUS_EPL_RESPONSE

Data value returned from a read command. Data is not valid while a command is being executed.

Address: 0x0E8000 + 0x8
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Data	31:0	vRO	0x0	Date to read.

11.26.2.10 SBUS_EPL_SPICO_IN

Access to the SPICO inside the SBUS master.

Address: 0x0E8000 + 0x9
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Interrupt	0	RW	0b	Interrupts SPICO controller.
GeneralPurposeIn	16:1	RW	0x0	General purpose 16-bit passed by host to SPICO.
Reserved	31:17	RSV	0x0	Reserved.

11.26.2.11 SBUS_EPL_SPICO_OUT

Response from the SPICO inside the SBUS master.

Address: 0x0E8000 + 0xA
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
GeneralPurposeOut	15:0	vRO	0x0	General purpose 16-bit passed by SPICO to host.
Reserved	31:16	RSV	0x0	Reserved.

11.26.2.12 SBUS_EPL_IP

Interrupt register for SBUS.

Address: 0x0E8000 + 0xB

Atomicity: 32

Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
DataDetectHigh	15:0	vCW1	0x0	Detected transition from 0b to 1b on *GeneralPurposeOut* register.
DataDetectLow	31:16	vCW1	0x0	Detected transition from 1b to 0b on *GeneralPurposeOut* register.

11.26.2.13 SBUS_EPL_IM

Interrupt mask register for SBUS.

Address: 0x0E8000 + 0xC

Atomicity: 32

Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
MaskDataDetectHigh	15:0	RW	0xFFFF	Mask detection of transition from 0b to 1b on *GeneralPurposeOut* register.
MaskDataDetectLow	31:16	RW	0xFFFF	Mask detection of transition from 1b to 0b on *GeneralPurposeOut* register.

11.26.2.14 ETHCLK_CFG[0..1]

The divider definition for each recovered clock. The output clock is equal to:

outputClk = recoveredClock x (M/N) / Divider

Address: 0x0E8000 + 0x1*i + 0xE

Atomicity: 32

Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
Divider	15:0	RW	0x0	A divider of 0 is undefined.
EnableDivider	16	RW	0b	Setting this bit enables the divider. To be toggled 0->1 after initializing the counters in the divider logic and configuring the muxes in the Ethernet Port Logic. The divider should be disabled when known changes are applied to the divider itself or when the reference clock source is changed.
EnableClockOut	17	RW	0b	Defines the state of the output clock pin. If the output is not enabled, the pin is driven with a constant 0b.
Reserved	31:18	RSV	0x0	Reserved.

11.26.2.15 ETHCLK_RATIO[0..1]

The ratio definition for each recovered clock. M must be smaller than N.

Address: 0x0E8000 + 0x1*i + 0x10
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
M	11:0	RW	0x0	Multiplier.
N	23:12	RW	0x0	Divider.
Reserved	31:24	RSV	0x0	Reserved.

11.26.2.16 PM_CLKOBS_CTRL

Ctrl register for the clock observation pads in PM_PADS.

Address: 0x0E8000 + 0x12
Atomicity: 32
Reset Domains: MASTER

Field Name	Bit(s)	Type	Default	Description
SelA	3:0	RW	0x0	Clock mux selection for clkobs pad CLKOUT_A: 0000b = SERDES — SerDes recovered clock. 0001b = SERDES_DIV — SerDes recovered clock with ratio and divider applied. 0010b = EPL_PLL_DIV8 — Divide-by-8 PLL output. All other values are reserved.
DivA	5:4	RW	00b	Additional output clock divider: 00b = No division 01b = Divide-by-2 10b = Divide-by-3 11b = Divide-by-8
SelB	9:6	RW	0x0	Clock mux selection for clkobs pad CLKOUT_B: 0000b = SERDES — SerDes recovered clock 0001b = SERDES_DIV — SerDes recovered clock with ratio and divider applied 0010b = FABRIC_PLL_DIV8 — Divide-by-8 PLL output All other values are reserved.
DivB	11:10	RW	00b	Additional output clock divider: 00b = No division 01b = Divide-by-2 10b = Divide-by-3 11b = Divide-by-8
Reserved	31:12	RSV	0x0	Reserved.

11.27 PCIE_PF Registers Description

11.27.1 PCIE_PF Map

Table 11-39 PCI Express Registers - Physical Function Map

Name	Address (Base = 0x100000)	Atomicity	Brief Description
PCIE_CTRL	Base + 0x0	32	Device Control Register.
PCIE_CTRL_EXT	Base + 0x1	32	Extended Device Control Register.
PCIE_EXVET	Base + 0x2	32	Extended Ethernet Type Support.
PCIE_GCR	Base + 0x3	32	PCIe Control Register.
PCIE_FACTPS	Base + 0x4	32	Function Active and Power State to Manageability.
PCIE_GCR_EXT	Base + 0x5	32	PCIe Control Extended Register.
PCIE_EICR	Base + 0x6	32	Extended Interrupt Cause Register.
PCIE_EIMR	Base + 0x7	32	Extended Interrupt Mask Register.
PCIE_PCA_FAULT	Base + 0x8	32	PCI Core Access Fault.
PCIE_THI_FAULT	Base + 0x10	32	Transmitter Host Interface Fault. Not implemented. Reserved for future use.
PCIE_FUM_FAULT	Base + 0x1C	32	Function Manager Fault.
PCIE_MAXHOLDQ[0..7]	Base + 0x1*i + 0x20	32	
PCIE_SM_AREA	Base + 0x28	32	Switch Manager Area.
PCIE_DGLORTMAP[0..7]	Base + 0x1*i + 0x30	32	GLORT mapping register.
PCIE_DGLORTDEC[0..7]	Base + 0x1*i + 0x38	32	GLORT decoding.
PCIE_TUNNEL_CFG	Base + 0x40	32	Tunnel configuration.
PCIE_SWPRI_MAP[0..15]	Base + 0x1*i + 0x50	32	Switch Priority Map.
PCIE_RSSRK[0..64][0..9]	Base + 0x10*j + 0x1*i + 0x800	32	RSS Random Key Register.
PCIE_RETA[0..64][0..31]	Base + 0x20*j + 0x1*i + 0x1000	32	Redirection Table.
PCIE_TC_CREDIT[0..63]	Base + 0x1*i + 0x2000	32	Traffic Class Credit.
PCIE_TC_MAXCREDIT[0..63]	Base + 0x1*i + 0x2040	32	Traffic Class Max Credit.
PCIE_TC_RATE[0..63]	Base + 0x1*i + 0x2080	32	Traffic Class Rate.
PCIE_TC_RATE_STATUS	Base + 0x20C0	32	Traffic Class Status.
PCIE_PAUSE	Base + 0x20C2	32	Pause Status Register
PCIE_DMA_CTRL	Base + 0x20C3	32	DMA Control Register.
PCIE_DMA_CTRL2	Base + 0x20C4	32	DMA Control 2 Register.
PCIE_DTXTCPFLGL	Base + 0x20C5	32	DMA Tx TCP Flags Control Low.
PCIE_DTXTCPFLGH	Base + 0x20C6	32	DMA Tx TCP Flags Control High.
PCIE_TPH_CTRL	Base + 0x20C7	32	TPH Control Register.
PCIE_MRQC[0..64]	Base + 0x1*i + 0x2100	32	Multiple Receive Queues Command Register.

Table 11-39 PCI Express Registers - Physical Function Map [Continued]

Name	Address (Base = 0x100000)	Atomicity	Brief Description
PCIE_TQMAP[0..2047]	Base + 0x1*i + 0x2800	32	Transmit Queue Mapping.
PCIE_RQMAP[0..2047]	Base + 0x1*i + 0x3000	32	Receive Queue Mapping.
PCIE_STATS_TIMEOUT	Base + 0x3800	32	Count Completion Timeouts.
PCIE_STATS_UR	Base + 0x3801	32	Count Unsupported Requests.
PCIE_STATS_CA	Base + 0x3802	32	Count Completer Aborts.
PCIE_STATS_UM	Base + 0x3803	32	Count Unsupported Messages Received.
PCIE_STATS_XEC	Base + 0x3804	32	Count Packet with Checksum Errors.
PCIE_STATS_VLAN_DROP	Base + 0x3805	32	Count VLAN Membership Drops.
PCIE_STATS_LOOPBACK_DROP	Base + 0x3806	32	Count Loopback Suppress Drops.
PCIE_STATS_NODESC_DROP	Base + 0x3807	32	Count No Descriptors Drop.
PCIE_RRTIME_CFG	Base + 0x3808	32	Read Response Time Configuration.
PCIE_RRTIME_LIMIT[0..2]	Base + 0x1*i + 0x380C	32	Read Response Time Limit.
PCIE_RRTIME_COUNT[0..3]	Base + 0x1*i + 0x3810	32	Read Response Time Event Count.
PCIE_SYSTIME	Base + 0x3814	32	System Time.
PCIE_SYSTIME0	Base + 0x3816	32	System Time Initial Value.
PCIE_SYSTIME_CFG	Base + 0x3818	32	System Time Configuration.
PCIE_PFVFBME	Base + 0x381A	32	BME bit state of the VFs.
PCIE_PHYADDR	Base + 0x381C	32	Physical Address Space Register.
PCIE_RDBAL[0..255]	Base + 0x40*i + 0x4000	32	Receive Descriptor Base Address Low.
PCIE_RDBAH[0..255]	Base + 0x40*i + 0x4001	32	Receive Descriptor Base Address High.
PCIE_RDLEN[0..255]	Base + 0x40*i + 0x4002	32	Receive Descriptor Length.
PCIE_TPH_RXCTRL[0..255]	Base + 0x40*i + 0x4003	32	Rx TPH Control Register.
PCIE_RDH[0..255]	Base + 0x40*i + 0x4004	32	Receive Descriptor Head.
PCIE_RDT[0..255]	Base + 0x40*i + 0x4005	32	Receive Descriptor Tail.
PCIE_RXQCTL[0..255]	Base + 0x40*i + 0x4006	32	Receive Queue Control.
PCIE_RXDCTL[0..255]	Base + 0x40*i + 0x4007	32	Receive Descriptor Control.
PCIE_RXINT[0..255]	Base + 0x40*i + 0x4008	32	Receive Interrupt Register.
PCIE_SRRCTL[0..255]	Base + 0x40*i + 0x4009	32	Split Receive Control Registers.
PCIE_QPRC[0..255]	Base + 0x40*i + 0x400A	32	Queue Packets Received Count.
PCIE_QPRDC[0..255]	Base + 0x40*i + 0x400B	32	Queue Packets Received Drop Count.
PCIE_QBRC_L[0..255]	Base + 0x40*i + 0x400C	32	Queue Bytes Received Count Low.
PCIE_QBRC_H[0..255]	Base + 0x40*i + 0x400D	32	Queue Bytes Received Count High.
PCIE_RX_SGLORT[0..255]	Base + 0x40*i + 0x400E	32	Source GLORT for each Q.
PCIE_TDBAL[0..255]	Base + 0x40*i + 0x8000	32	Transmit Descriptor Base Address Low.
PCIE_TDBAH[0..255]	Base + 0x40*i + 0x8001	32	Transmit Descriptor Base Address High.
PCIE_TDLEN[0..255]	Base + 0x40*i + 0x8002	32	Transmit Descriptor Length.
PCIE_TPH_TXCTRL[0..255]	Base + 0x40*i + 0x8003	32	Tx TPH Control Registers.

Table 11-39 PCI Express Registers - Physical Function Map [Continued]

Name	Address (Base = 0x100000)	Atomicity	Brief Description
PCIE_TDH[0..255]	Base + 0x40*i + 0x8004	32	Transmit Descriptor Head.
PCIE_TDT[0..255]	Base + 0x40*i + 0x8005	32	Transmit Descriptor Tail.
PCIE_TXDCTL[0..255]	Base + 0x40*i + 0x8006	32	Transmit Descriptor Control.
PCIE_TXQCTL[0..255]	Base + 0x40*i + 0x8007	32	Transmit Queue Control
PCIE_TXINT[0..255]	Base + 0x40*i + 0x8008	32	Transmit Interrupt Register.
PCIE_QPTC[0..255]	Base + 0x40*i + 0x8009	32	Queue Packets Transmitted Count.
PCIE_QBTC_L[0..255]	Base + 0x40*i + 0x800A	32	Queue Bytes Transmitted Count Low.
PCIE_QBTC_H[0..255]	Base + 0x40*i + 0x800B	32	Queue Bytes Transmitted Count High.
PCIE_TQDLOC[0..255]	Base + 0x40*i + 0x800C	32	Tx Descriptors Location in Tx Cache.
PCIE_TX_SGLORT[0..255]	Base + 0x40*i + 0x800D	32	Source GLORT for each Q.
PCIE_PFVTCTL[0..255]	Base + 0x40*i + 0x800E	32	VT Control Register
PCIE_TX_DESC[0..255][0..255][0..3]	Base + 0x400*k + 0x4*j + 0x1*i + 0x40000	32	Transmit Descriptor Cache.
PCIE_PBACL[0..71]	Base + 0x1*i + 0x10000	32	Pending Bit Array Table.
PCIE_INT_MAP[0..15]	Base + 0x1*i + 0x10080	32	Interrupt Mapping Register.
PCIE_MSIX_VECTOR[0..767]	Base + 0x4*i + 0x11000	32	MSI-X Vector Table.
PCIE_INT_CTRL	Base + 0x4*i + 0x12000	32	Interrupt Control Register.
PCIE_ITR[0..767]	Base + 0x1*i + 0x12400	32	Interrupt Timers.
PCIE_ITR2[0..767]	Base + 0x2*i + 0x12800	32	Interrupt Timers Config 2.
PCIE_IP	Base + 0x13000	32	Interrupt from PEP Register.
PCIE_IM	Base + 0x13001	32	Interrupt from PEP Mask to Switch Manager.
PCIE_IB	Base + 0x13002	32	Interrupt from PEP Mask to Boot State Machine.
PCIE_SRAM_IP	Base + 0x13003	32	SRAM Interrupt from PEP.
PCIE_SRAM_IM	Base + 0x13004	32	SRAM Interrupt from PEP Mask.
PCIE_VLAN_TABLE[0..64][0..127]	Base + 0x80*j + 0x1*i + 0x14000	32	VLAN Table.
PCIE_MBMEM[0..2047]	Base + 0x1*i + 0x18000	32	Mailboxes Memory.
PCIE_MBX[0..63]	Base + 0x1*i + 0x18800	32	VF Mailboxes Control.
PCIE_MBICR[0..1]	Base + 0x1*i + 0x18840	32	VF Mailboxes Interrupt Causes Register.
PCIE_GMBX	Base + 0x18842	32	Global Mailbox Control.
PCIE_PFVFLRE	Base + 0x18844	32	PF VFLR Events Indication.
PCIE_PFVFLREC	Base + 0x18846	32	PF VFLR Events Clear.
PCIE_TEST_CFG0	Base + 0x18849	32	PCIe TEST Configuration Register.
PCIE_INT_SRAM_CTRL	Base + 0x18850	32	Interrupt SRAM Control Register.
PCIE_FUM_SRAM_CTRL	Base + 0x18852	32	FUM SRAM Control Register.
PCIE_PCA_SRAM_CTRL	Base + 0x18854	32	PCA SRAM Control Register.
PCIE_PP_SRAM_CTRL	Base + 0x18858	32	PP SRAM Control Register.
PCIE_PCW_SRAM_CTRL	Base + 0x1885C	32	PCW SRAM Control Register.

Table 11-39 PCI Express Registers - Physical Function Map [Continued]

Name	Address (Base = 0x100000)	Atomicity	Brief Description
PCIE_RHI_SRAM_CTRL1	Base + 0x18872	32	RHI SRAM Control1 Register.
PCIE_RHI_SRAM_CTRL2	Base + 0x18860	32	RHI SRAM Control2 Register.
PCIE_THI_SRAM_CTRL1	Base + 0x18864	32	THI SRAM Control1 Register.
PCIE_THI_SRAM_CTRL2	Base + 0x18868	32	THI SRAM Control2 Register.
PCIE_FPGA_REVID	Base + 0x1886A	32	PEP FPGA Revision ID Register.
PCIE_TIMEOUT_CFG	Base + 0x1886B	32	PCIe Timeouts Configuration Register.
PCIE_LVMMC	Base + 0x18880	32	PCIe Last VM Misbehavior Cause Register.
PCIE_LVMMI	Base + 0x18881	32	PCIe Last VM Misbehavior Indexes Register.
PCIE_HOST_MISC	Base + 0x19000	32	PCIe Host Miscellaneous Register.
PCIE_HOST_LANE_CTRL	Base + 0x19001	32	PCIe Host Lane Control Register.
PCIE_SERDES_CTRL[0..7]	Base + 0x2*i + 0x19010	32	SerDes Control Register.

11.27.2 PCIE_PF Registers

11.27.2.1 PCIE_CTRL

Address: 0x100000 + 0x0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
LTSSM_ENABLE	0	RW	0b	When low, holds the LTSSM in the Detect state until any necessary configuration has completed. This bit is cleared only by PCIE reset events, and is never cleared by host or SW/BSM.
REQ_RETRY_EN	1	RW	0b	When asserted, incoming configuration requests are completed with a configuration request retry status.
BAR4_Allowed	2	CW1	1b	This register bit is set after a COLD_RESET, and can be cleared to block future access to BAR4.
Reserved	3	RSV	0b	Reserved.
RxLaneflipEn	4	RW	0b	Enable the physical RX lane flip so that a physical layout problem of lane reversed can be fixed. 0b = Indicates that core currently requires the LSB lane (i.e lane0) to be physically presented. 1b = Enables the flip of the MSB lane to lane0. The field is set to the proper value by NVM/BSM. ***Note:*** The lane flip feature is not supported in the FM10000. The bit must always be set to 0b.
TxLaneflipEn	5	RW	0b	Enable the physical TX lane flip so that a physical layout problem of lane reversed can be fixed. 0b = Indicates that core currently requires the LSB lane (i.e lane0) to be physically presented. 1b = Enables the flip of the MSB lane to lane0. The field is set to the proper value by NVM/BSM. ***Note:*** The lane flip feature is not supported in the FM10000. The bit must always be set to 0b.
Reserved	31:6	RSV	0x0	Reserved.

11.27.2.2 PCIE_CTRL_EXT

Address: 0x100000 + 0x1
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
NS_DIS	0	RW	0b	No Snoop Disable. This bit is non-functional in the FM10000. It remain set to 0b.
RO_DIS	1	RW	0b	Relaxed Ordering Disable. 0b = When this bit is cleared and the Enable Relaxed Ordering bit in the Device Control register is set, the device requests relaxed ordering transactions per queues as configured in the TPH_RXCTRL[n] and TPH_TXCTRL[n] registers. 1b = The device does not request any relaxed ordering transactions.
SwitchLoopback	2	RW	1b	Enables a switch loopback. 0b = Normal operation. 1b = Packets sent to the switch are looped back to PEP, and packets sent by the switch are looped back to switch. This field resets on COLD_RESET events. This field may not be written while traffic is flowing.
Reserved	31:3	RSV	0x0	Reserved.

11.27.2.3 PCIE_EXVET

Address: 0x100000 + 0x2
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
EthernetType	15:0	RW	0x8100	Extra Ethernet type. TX parser jumps over if detected, RX parser jumps over and set VEXT2 in the RX descriptor.
TagSize	17:16	RW	00b	Size in 16-bit quanta. Sizes supported are 1,2,3. Setting to 0b disables detection of this type.
AfterVLAN	18	RW	0b	Defines if the extended tag is expected before or after the VLAN (8100) tag. 0b = Before the VLAN tag. 1b = After the VLAN tag.
Reserved	31:19	RSV	0x0	Reserved.

11.27.2.4 PCIE_GCR

Address: 0x100000 + 0x3
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Reserved1	8:0	RO	0x4	Reserved.
Reserved2	17:9	RO	0x0	Reserved.
PCIeCapabilityVersion	18	RO	1b	Read only field reporting supported PCIe capability version. 0b = Capability version: 0x1. 1b = Capability version: 0x2.
Reserved	31:19	RSV	0x0	Reserved.

11.27.2.5 PCIE_FACTPS

Register for use by the device firmware for configuration.

Address: 0x100000 + 0x4
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Func0PowerState	1:0	RO	00b	Power state indication of function 0. 00b = DR 01b = D0u 10b = D0a 11b = D3
Reserved	2	RSV	0b	Reserved.
Func0Aux_En	3	RO	0b	Function 0 Auxiliary (AUX) Power PM Enable bit shadow from the configuration space.
Reserved	31:4	RSV	0x0	Reserved.

11.27.2.6 PCIE_GCR_EXT

Address: 0x100000 + 0x5
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
APBACD	0	RW	0b	Auto PBA Clear Disable. 0b = Any active PBA entry is cleared on the falling edge of the appropriate interrupt request to the PCIe block. The appropriate interrupt request is cleared when software sets the associated interrupt mask bit in the EIMR (re-enabling the interrupt) or by direct write to clear to the PBA. 1b = Software can clear the PBA only by direct write to clear access to the PBA bit. ***Note:*** ** NOT IMPLEMENTED in the FM10000. The PBA is set when an interrupt is detected, and cleared when the MSI-X is sent. **
Reserved	31:1	RSV	0x0	Reserved.

11.27.2.7 PCIE_EICR

Address: 0x100000 + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
PCA_Fault	0	RO	0b	Reports PCIE_PCA_FAULT.*Valid.*
Reserved	4:1	RSV	0x0	Reserved.
FUM_Fault	5	RO	0b	Reports PCIE_FUM_FAULT.*Valid.*
MailBox	6	CW1	0b	Reports a mailbox interrupt. Read PCIE_MBICR[0..1] to determine the sending VF and PCIE_GMBX to see if SM is the sender. Write 1b to clear.
SwitchReady	7	CW1	0b	Reports a switch transition from not ready to ready. Write 1b to clear.
SwitchNotReady	8	CW1	0b	Reports a switch transition from ready to not ready. Write 1b to clear.

Address: **Atomicity:** **Reset Domains:**	**0x100000 + 0x6** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SwitchInterrupt	9	RO	0b	Interrupt from switch active.
SramError	10	RO	0b	Indicates a correctable or an uncorrectable ECC/parity error occurred on an SRAM of the PEP. Read PCIE_SRAM_IP register to retrieve the concerned block. The bit is kept asserted until all asserted bits in PCIE_SRAM_IP are cleared by the appropriate reset or by CW1. This interrupt is relevant only for the PEP that hosts the SM, and should otherwise be masked.
VFLR	11	RO	0b	Reports a VFLR event occurred. Read PCIE_PFVFLRE to retrieve the VF.
MaxHoldTime	12	RO	0b	Indicates the max hold timer has timed out for a receive queue. It means a receive queue that was configured with *DropOnEmpty*=0 waited for available Rx descriptors for a longer time than PCIE_DMA_CTRL.*MaxHoldTime*. The device has set *DropOnEmpty* to 1 for that queue in return. Read PCIE_MAXHOLDQ[0..7] registers to retrieve the concerned queue(s). The bit is kept asserted until all asserted bits in PCIE_MAXHOLDQ are cleared by CW1.
Reserved	31:13	RSV	0x0	Reserved.

11.27.2.8 PCIE_EIMR

Address: **Atomicity:** **Reset Domains:**	**0x100000 + 0x7** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Mask[0..15]	31:0	RW	01b	(16 x 2-bits) Read as 1 if the corresponding interrupt is masked. The last 2 items are reserved. 00b or 11b = Do not change the mode. 01b = Disable the corresponding interrupt cause (set the mask). 10b = Enable the corresponding interrupt cause (clear the mask)

11.27.2.9 PCIE_PCA_FAULT

Address: **Atomicity:** **Reset Domains:**	**0x100000 + 0x8** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Address	63:0	RO	0x0	The address at the time the fault was detected.
Reserved	65:64	RSV	0x0	Reserved.
FaultType	103:96	RO	0x0	00000000b = PCA_NO_FAULT 00000001b = PCA_UNMAPPED_ADDR 00000010b = PCA_BAD_QACCESS_PF 00000011b = PCA_BAD_QACCESS_VF 00000100b = PCA_MALICIOUS_REQ 00000101b = PCA_POISONED_TLP 00000110b = PCA_TLP_ABORT All other values are reserved.

Address: **0x100000 + 0x8**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**

Field Name	Bit(s)	Type	Default	Description
VF	109:104	RO	0x0	Indicate VF that made the fault.
PF	110	RO	0b	Set by hardware to 1b to indicate fault was made by PF.
Valid	111	CW1	0b	Set by hardware to 1b when a fault is logged. No further fault reported for this sub-unit until this bit is cleared.
Reserved	127:112	RSV	0x0	Reserved.

11.27.2.10 PCIE_THI_FAULT

This register reports Tx descriptor fault events where a Tx queue is disabled by the device and remapped to the PF further to a non-consistent contents in a Tx descriptor.

Address: **0x100000 + 0x10**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
Address	63:0	RO	0x0	The Tx queue index on which the fault occurred.
Reserved	95:64	RO	0x0	Reserved.
FaultType	103:96	RO	0x0	Fault type values reported: 0b = THI_NO_FAULT 1b = THI_MAL_DIS_Q_FAULT
VF	109:104	RO	0x0	Indicates the VF that made the fault.
PF	110	RO	0b	Set by hardware to 1b to indicate fault was made by PF.
Valid	111	CW1	0b	Set by hardware to 1b when a fault is logged. No further fault reported for this sub-unit until this bit is cleared.
Reserved	127:112	RSV	0x0	Reserved.

11.27.2.11 PCIE_FUM_FAULT

Address: **0x100000 + 0x1C**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Address	63:0	RO	0x0	The double-word address at the time the fault was detected. Only the lower 22 bits in this field are valid. The remaining bits are reserved, and always set to 0b.
Reserved	95:64	RSV	0x0	Reserved.

Address: 0x100000 + 0x1C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
FaultType	103:96	RO	0x0	00000000b = FUM_NO_FAULT 00000001b = FUM_UNMAPPED_ADDR 00000010b = Reserved 00000011b = FUM_BAD_VF_QACCESS 00000100b = FUM_ADD_DECODE_ERR 00000101b = FUM_RO_ERROR 00000110b = FUM_QPRC_CRC_ERROR 00000111b = FUM_CSR_TIMEOUT 00001000b = FUM_INVALID_TYPE 00001001b = FUM_INVALID_LENGTH 00001010b = FUM_INVALID_BE 00001011b = FUM_INVALID_ALIGN All other values are reserved.
VF	109:104	RO	0x0	Indicate VF that made the fault.
PF	110	RO	0b	Set by hardware to 1b to indicate fault was made by PF.
Valid	111	CW1	0b	Set by hardware to 1b when a fault is logged. No further fault reported for this sub-unit until this bit is cleared.
Reserved	127:112	RSV	0x0	Reserved.

11.27.2.12 PCIE_MAXHOLDQ[0..7]

Indicates the receive queue(s) for which the max hold timer has timed out.

Address: 0x100000 + 0x1*i + 0x20
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
MaxHoldQ[0..31]	31:0	CW1	0b	(32 x 1-bit) Each bit 'n' in register 'i' represents receive queue index 32*i+n.

11.27.2.13 PCIE_SM_AREA

Address: 0x100000 + 0x28
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SerialNumber	63:0	RW	0x0	Serial Number copied by SM at boot time from PCIe config space (registers PCIE_SPD_NUMBER_L/H). The *SerialNumber* is encoded big endian. Bits[7:0] = EUI[7] Bits[15:8] = EUI[6] Bits[23:16] = EUI[5] Bits[31:24] = EUI[4] Bits[39:40] = EUI[3] Bits[47:40] = EUI[2] Bits[55:48] = EUI[1] Bits[63:56] = EUI[0]

11.27.2.14 PCIE_DGLORTMAP[0..7]

The register set is used for comparing incoming DGLORT from fabric to known patterns. The patterns are in form a *VALUE*/*MASK*. The DGLORT is compared to all patterns, and the highest matching pattern defines how the DGLORT is interpreted.

Address:	**0x100000 + 0x1*i + 0x30**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Value	15:0	RW	0x0	Value to compare.
Mask	31:16	RW	0x0	Selects which bits to compare. • Setting *Mask* to 0xFFFF means all bits are significant for comparison to *Value*. • Setting *Mask* to 0x0000 and *Value* to 0x0000 is always match. • Setting *Mask* to 0x0000 and *Value* to other than 0x0000 never matches.

11.27.2.15 PCIE_DGLORTDEC[0..7]

This register defines how to decode the DGLORT, the index 'n' selected correspond to the highest matching DGLORT in PCIE_GLORTMAP.

The DGLORT contains the Queue Number and the VF number. The size of the queue number and the size of the VF number is variable.

Address:	**0x100000 + 0x1*i + 0x38**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Qlength	3:0	RW	0x0	Defines the length in bits for the Queue number encoded in DGLORT. A value of 0 means the queue # is not present in the DGLORT. Maximum width is 8 bits.
VSIlength	6:4	RW	000b	Defines the length in bits for the VSI number encoded in DGLORT, relative to VSI Base. A value of 0 means the *VSIbase* is used as the VSI number.
VSIbase	13:7	RW	0x0	The VSI base number.
PClength	15:14	RW	00b	Defines how many bits of Priority Class are included in the queue number computation. A value of 0 means that VPRI has no influence in the queue selection.
Qbase	23:16	RW	0x0	The first queue of a block of queues for this decoder.
RSSlength	26:24	RW	000b	The number of bits used out of the RETA table value. A value of 0 means the hash key is not computed and the RETA indirect table is not used.
InnerRSS	27	RW	0b	Use the inner header for RSS (if present). 0b = Always use the outer header regardless if there is a recognized tunnel. 1b = Use the inner header if a tunnel is detected.
Reserved	31:28	RSV	0x0	Reserved.

11.27.2.16 PCIE_TUNNEL_CFG

Address:	**0x100000 + 0x40**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VXLAN_PORT	15:0	RW	0x0	Defines the L4 port used for VXLAN. Setting to 0b disables VXLAN detection.
Protocol_Type	31:16	RW	0x0	Defines the encapsulation ID for NVGRE and the Next Protocol type supported inside NGE headers. Setting to 0 disables GRE detection. Next Protocol type supported is for Ethernet, which uses the value of 0x6558.
NGE_PORT	47:32	RW	0x0	Defines the L4 port used for NGE. Setting to 0b disable NGE detection. UDP port defined for NGE (a.k.a. GENEVE) is 6081 (decimal).
Reserved	63:48	RSV	0x0	Reserved.

11.27.2.17 PCIE_SWPRI_MAP[0..15]

Address:	**0x100000 + 0x1*i + 0x50**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
pc	2:0	RW	000b	Defines the priority class for queue selection from the FTAG SWPRI.
Reserved	31:3	RSV	0x0	Reserved.

11.27.2.18 PCIE_RSSRK[0..64][0..9]

Index 1..64 is for VF, index 0 is for PF/VMDq.

Address:	**0x100000 + 0x10*j + 0x1*i + 0x800**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
K0	7:0	RW	0x0	RSS Key Byte [4*n+0] of the RSS random key, for each register [n].
K1	15:8	RW	0x0	RSS Key Byte [4*n+1] of the RSS random key, for each register [n].
K2	23:16	RW	0x0	RSS Key Byte [4*n+2] of the RSS random key, for each register [n].
K3	31:24	RW	0x0	RSS Key Byte [4*n+3] of the RSS random key, for each register [n].

11.27.2.19 PCIE_RETA[0..64][0..31]

Index 1..64 is for VF, index 0 is for PF. The contents of the redirection table are not defined following reset of the Memory Configuration registers. System software must initialize the table prior to enabling multiple receive queues. It can also update the redirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

Address: 0x100000 + 0x20*j + 0x1*i + 0x1000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Entry0	7:0	RW	0x0	Entry0 defines the RSS output index for hash value [4*n+0]. While [n] is the register index, equals to 0...31.
Entry1	15:8	RW	0x0	Entry1 defines the RSS output index for hash value [4*n+1]. While [n] is the register index, equals to 0...31.
Entry2	23:16	RW	0x0	Entry2 defines the RSS output index for hash value [4*n+2]. While [n] is the register index, equals to 0...31.
Entry3	31:24	RW	0x0	Entry3 defines the RSS output index for hash value [4*n+3]. While [n] is the register index, equals to 0...31.

11.27.2.20 PCIE_TC_CREDIT[0..63]

Implements a token bucket for rate control of traffic shaping group “n”.

Address: 0x100000 + 0x1*i + 0x2000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Credit	20:0	RW	0xFFFF	Current credit in bytes available for this traffic class (Signed number). Packets are not pulled from memory when the credit is negative. Credits are subtracted when memory reads are issued to host. Credits are added periodically (each Interval*512ns period) (once per Interval refresh periods) by the Quanta value and saturates saturate at MaxCredit.
Reserved	23:21	RSV	000b	Reserved.
IntervalLeft	31:24	RW	0x1	IntervalLeft counter counts down from Interval once every refresh period. Credits are added when IntervalLeft reaches zero.

11.27.2.21 PCIE_TC_MAXCREDIT[0..63]

Implements a token bucket for rate control of traffic shaping group “n”.

Address: 0x100000 + 0x1*i + 0x2040
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
MaxCredit	19:0	RW	0xFFFF	The maximum credit available for a given queue (Unsigned number).
Reserved	31:20	RSV	0x0	Reserved.

11.27.2.22 PCIE_TC_RATE[0..63]

Implements a token bucket for rate control of traffic shaping group “n”.

Address: 0x100000 + 0x1*i + 0x2080
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Quanta	15:0	RW	0xFFFF	The additional credits allowed per time interval.
Interval	23:16	RW	0x1	The time interval to give credits in units of refresh periods. A refresh period is 512 ns in gen3 mode, 1024 ns in gen2 mode, or 2048 ns in gen1 mode. An Interval value of 0b is treated the same as 1b.
Reserved	31:24	RSV	0x0	Reserved.

11.27.2.23 PCIE_TC_RATE_STATUS

Address: 0x100000 + 0x20C0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
Allowed[0..63]	63:0	RO	0b	(64 x 1-bit) Reflects if a traffic class currently has non-negative Credit, allowing packets to be transmitted.

11.27.2.24 PCIE_PAUSE

Address: 0x100000 + 0x20C2
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TxPause[0..7]	7:0	RO	0b	(8 x 1-bit) Indicates if a Tx Priority Class is currently paused or not.
RxPause[0..7]	15:8	RO	0b	(8 x 1-bit) Indicates if an Rx Priority Class is currently paused or not. ** OBSOLETE, NOT IMPLEMENTED **
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.25 PCIE_DMA_CTRL

Excepted to the *DataPathReset* bit, all the RW fields of this register can be changed only once all the Rx and Tx queues are disabled.

Address: 0x100000 + 0x20C3
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TxEnable	0	RW	0b	Defines if TX DMA is active or not. Setting this bit to 0b causes the TX DMA controller to stop gracefully (i.e., not issue further PCIe requests). Once *TxEnable* is reasserted, Tx queues must be re-initialized to resume operation.
TxHostPending	1	RO	0b	Indicates if TX DMA has pending host requests.
TxData	2	RO	0b	Indicates if TX DMA has pending data waiting to be transmitted to Ethernet fabric.
TxActive	3	RO	0b	Indicates if TX DMA is active (i.e., there is some pending work in Tx).
RxEnable	4	RW	0b	Defines if RX DMA is active or not. Setting this bit to 0b causes the RX DMA controller to stop gracefully (i.e., the controller stops issuing further PCIe requests). Once *RxEnable* is reasserted, Rx queues must be re-initialized to resume operation.
RxHostPending	5	RO	0b	Indicates if RX DMA has pending host requests.
RxData	6	RO	0b	Indicates if RX DMA has pending data being processed. This bit is cleared if there is no packet being processed at that time.
RxActive	7	RO	0b	Indicates if RX DMA is active (i.e., there is some pending work in Rx).
RxDescSize	8	RW	1b	Defines the Rx Descriptor size: 0b = 16-byte descriptors. 1b = 32-byte descriptors. Must be set to 1b. Only 32-byte descriptors are supported.
MinMSS	22:9	RW	0x40	Minimum MSS allowed. If a TSO session request an MSS lower than this limit, the queue is shut down and an event is posted.
MaxHoldTime	27:23	RW	0x0	Defines log2() of maximum waiting time for descriptors to become available. The maximum time is 2^^31 * PCLK.
Reserved	28	RSV	0b	Reserved.
DataPathReset	29	RW	0b	Resets the PEP internal data path. The bit is cleared by hardware.
MaxNumOfQs	31:30	RW	00b	Defines the maximum number of queues that are enabled. It induces the maximum number of Rx descriptors that can be stored per queue in the internal Rx descriptor cache. The same configuration is made in all TQDLOC[Q] registers for the number of Tx descriptors stored internally per queue. An undefined PEP behavior may result in case PF enables more queues than the upper limit specified by this field. 00b = Maximum of 256 queues (i.e., internal storage capacity for up to 32 Rx descriptors per queue). 01b = Maximum of 128 queues (i.e., internal storage capacity for up to 64 Rx descriptors per queue). 10b = Maximum of 64 queues (i.e., internal storage capacity for up to 128 Rx descriptors per queue). 11b = Reserved.

11.27.2.26 PCIE_DMA_CTRL2

Address:	**0x100000 + 0x20C4**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
Reserved	4:0	RSV	0x0	Reserved.
TxFrameSpacing	12:5	RW	0x0	Defines the time between the starting frame transmit to the switch and the next start frame. The time is defined in 2 ns increments for GEN3, 4 ns for GEN2 and 8 ns for GEN1. Setting this value to anything smaller than 10 is equivalent to sending frames back to back with no space in between, as the time would be smaller than the minimum packet size (72 bytes). This feature is not supported, so the field is set to 0b.
SwitchReady	13	RO	1b	Indicates if the switch is ready or not.
RxDescReadPrio	16:14	RW	000b	Controls the relative priority of Rx descriptor read w.r.t. Tx descriptor and Tx data read processes. Each process is mapped to a unique priority value: 000b = Highest priority, 001b = Round-robin at medium priority. 010b = Round-robin at medium priority. 011b = Round-robin at medium priority. 100b = Lowest priority All other values are reserved. The driver never sets the same value in two or more DMA_CTRL2.*xxxReadPrio* fields. It is recommended to use only the default setting.
TxDescReadPrio	19:17	RW	001b	Controls the relative priority of Tx descriptor read w.r.t. Rx descriptor and Tx data read processes. Each process is mapped to a unique priority value: 000b = Highest priority, 001b = Round-robin at medium priority. 010b = Round-robin at medium priority. 011b = Round-robin at medium priority. 100b = Lowest priority All other values are reserved. The driver never sets the same value in two or more DMA_CTRL2.*xxxReadPrio* fields. It is recommended to use only the default setting.
TxDataReadPrio	22:20	RW	100b	Controls the relative priority of Tx data read w.r.t. Rx descriptor and Tx descriptor read processes. Each process is mapped to a unique priority value: 000b = Highest priority, 001b = Round-robin at medium priority. 010b = Round-robin at medium priority. 011b = Round-robin at medium priority. 100b = Lowest priority All other values are reserved. The driver never sets the same value in two or more DMA_CTRL2.*xxxReadPrio* fields. It is recommended to use only the default setting.
Reserved	31:23	RSV	0x0	Reserved.

11.27.2.27 PCIE_DTXTCPFLGL

This register holds the mask bits for the TCP flags in Tx segmentation.

Address: 0x100000 + 0x20C5
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TCP_flg_first_seg	11:0	RW	0xFF6	TCP Flags First Segment. Bits that make AND operation with the TCP flags at TCP header in the first segment.
Reserved	15:12	RSV	0x0	Reserved.
TCP_Flg_mid_seg	27:16	RW	0xFF6	TCP Flags Middle Segments. The low bits that make AND operation with the TCP flags at TCP header in the middle segments.
Reserved	31:28	RSV	0x0	Reserved.

11.27.2.28 PCIE_DTXTCPFLGH

This register holds the mask bits for the TCP flags in Tx segmentation.

Address: 0x100000 + 0x20C6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TCP_Flg_lst_seg	11:0	RW	0xF7F	TCP Flags Last Segment. Bits that make AND operation with the TCP flags at TCP header in the last segment.
Reserved	31:12	RSV	0x0	Reserved.

11.27.2.29 PCIE_TPH_CTRL

Address: 0x100000 + 0x20C7
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
DisableReadHint	7	RO	1b	Disable Read Hint: 0b = The FM10000 uses the RXCTL.*CPUID* field for receive PCIe read transactions with TPH bit set and the TXCTL.*CPUID* for transmit PCIe read transactions. 1b = A CPUID value of zero is used in receive and transmit PCIe read transactions. ***Note:*** This field is not operational in the FM10000, and has been fixed to 1b.
DescPH	9:8	RW	00b	Descriptor PH. Defines the PH field used when a TPH hint is given for descriptor associated traffic (descriptor fetch, descriptor write back).
DataPH	11:10	RW	10b	RW Data PH. Defines the PH field used when a TPH hint is given for data associated traffic (Tx data read, Rx data write).
Reserved	31:12	RSV	0x0	Reserved.

11.27.2.30 PCIE_MRQC[0..64]

Queue assignment control. Index 1..64 is for VF, index 0 is for PF. Each bit, when set, enables a specific field selection to be used by the hash function. Several bits can be set at the same time. The bit PCIE_DGLORTDEC[x].*InnerRSS* controls if fields are from outer or inner IP header (if present).

Address: 0x100000 + 0x1*i + 0x2100
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TcpIPv4	0	RW	0b	Enable TcpIPv4 hash function.
IPv4	1	RW	0b	Enable IPv4 hash function.
Reserved	3:2	RSV	00b	Reserved.
IPv6	4	RW	0b	Enable IPv6 hash function.
TcpIPv6	5	RW	0b	Enable TcpIPv6 hash function.
UdpIPV4	6	RW	0b	Enable UdpIPv4 hash function.
UdpIPV6	7	RW	0b	Enable UdpIPv6 hash function.
Reserved	31:8	RSV	0x0	Reserved.

11.27.2.31 PCIE_TQMAP[0..2047]

Address: 0x100000 + 0x1*i + 0x2800
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
PhysicalQueue	7:0	RW	0x0	Defines the physical queue number for a virtual queue on RX. If NumVFs is greater than 8, the index is (vf*32 + q&31). Otherwise, the index is (vf*256 + q).
Reserved	31:8	RSV	0x0	Reserved.

11.27.2.32 PCIE_RQMAP[0..2047]

Address: 0x100000 + 0x1*i + 0x3000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
PhysicalQueue	7:0	RW	0x0	Defines the physical queue number for a virtual queue on TX. If NumVFs is greater than 8, the index is (vf*32 + q&31). Otherwise, the index is (vf*256 + q).
Reserved	31:8	RSV	0x0	Reserved.

11.27.2.33 PCIE_STATS_TIMEOUT

Address: 0x100000 + 0x3800
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Timeout	31:0	RO	0x0	Number of PEP requests to PCIe which ended as completion Timeout.

11.27.2.34 PCIE_STATS_UR

Address: 0x100000 + 0x3801
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Timeout	31:0	RO	0x0	Number of PEP requests to PCIe which completed as Unsupported Request (UR).

11.27.2.35 PCIE_STATS_CA

Address: 0x100000 + 0x3802
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Timeout	31:0	RO	0x0	Number of PEP requests to PCIe which ended as Completer Abort (CA).

11.27.2.36 PCIE_STATS_UM

Address: 0x100000 + 0x3803
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Timeout	31:0	RO	0x0	Number of PCIe messages received from host which are not supported by the PEP and that were silently dropped.

11.27.2.37 PCIE_STATS_XEC

Address: 0x100000 + 0x3804
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
XEC	31:0	RO	0x0	Number of packets received with either IPv4, TCP, UDP or SCTP XSUM errors. Counter not incremented when a packet has any MAC error (CRC, length, under-size, over-size, byte error or symbol error).

11.27.2.38 PCIE_STATS_VLAN_DROP

Address: 0x100000 + 0x3805
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Drop	31:0	RO	0x0	Count the number of packets dropped due to VLAN membership.

11.27.2.39 PCIE_STATS_LOOPBACK_DROP

Address: 0x100000 + 0x3806
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Drop	31:0	RO	0x0	Count the number of packets dropped due to loopback suppress.

11.27.2.40 PCIE_STATS_NODESC_DROP

Address: 0x100000 + 0x3807
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Drop	31:0	RO	0x0	Count the number of packets dropped due to lack of descriptors.

11.27.2.41 PCIE_RRTIME_CFG

Defines the read response time measurement filter.

Address: 0x100000 + 0x3808
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
UnitID	7:0	RW	0x0	Define ID to watch.
FilterType	9:8	RW	00b	Defines what is being monitored: 00b = Do not filter, all read completion measured. 01b = Only tracks read requests for PF. 10b = Only tracks read requests for VF defined in *UnitID*. 11b = Only tracks read requests for queue defined in *UnitID*.

TransactionType	12:10	RW	000b	Defines what type of read request is monitored:
				Bit[0] = Watch RX descriptor read requests.
				Bit[1] = Watch TX descriptor read requests.
				Bit[2] = Watch TX packet read requests.

ResetCounts	13	WO	0b	Request to clear all RRTIME_COUNT registers. The bit is self-cleared by the device once the counters have been reset.
Reserved	31:14	RSV	0x0	Reserved.

11.27.2.42 PCIE_RRTIME_LIMIT[0..2]

Address: 0x100000 + 0x1*i + 0x380C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
TimeLimit	15:0	RW	0x0	Defines the time limit for counting purpose.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.43 PCIE_RRTIME_COUNT[0..3]

Tracks event count for each bin:

- RRTIME_COUNT[0] : count if time <= RRTIME_LIMIT[0]
- RRTIME_COUNT[1] : count if time <= RRTIME_LIMIT[1]
- RRTIME_COUNT[2] : count if time <= RRTIME_LIMIT[2]
- RRTIME_COUNT[3] : count if time > RRTIME_LIMIT[2]

Address: 0x100000 + 0x1*i + 0x3810
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
CountEvent	31:0	RO	0x0	Count the events on that bin.

11.27.2.44 PCIE_SYSTIME

Time reference. The time unit is system implementation depended but should normally represent nanoseconds.

Address: 0x100000 + 0x3814
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
CurrentTime	63:0	RO	0x0	System Time.

11.27.2.45 PCIE_SYSTIME0

Address: 0x100000 + 0x3816
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
Time0	63:0	RW	0x0	The content of this register is transfered to PCIE_SYSTIME at time 0 reset from management.

11.27.2.46 PCIE_SYSTIME_CFG

Address:	**0x100000 + 0x3818**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Step	3:0	RW	0xA	The number of time unit per tick. At least 2, values or 1 or 0 are not supported.
Reserved	31:4	RSV	0x0	Reserved.

11.27.2.47 PCIE_PFVFBME

Address:	**0x100000 + 0x381A**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VFBME	63:0	RO	0x0	Bit [n] reflects the current setting of the BME bit in the VF[n] configuration space.

11.27.2.48 PCIE_PHYADDR

Address:	**0x100000 + 0x381C**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PhyAddrSpace	31:0	RW	0x0000FFFF	Bitmask that defines the size of the host physical address space. Out of range accesses are not sent to host and are terminated internally as Unsupported Requests(UR). Default is a 48-bits address space. 00000000000000000000000000000000b = 32-bit address space. 00000000000000000000000000000001b = 33-bit address space. 00000000000000000000000000000011b = 34-bit address space. . . . 00000000000000001111111111111111b = 48-bit address space. . . . 01111111111111111111111111111111b = 63-bit address space. 11111111111111111111111111111111b = 64-bit address space.

11.27.2.49 PCIE_RDBAL[0..255]

Address:	**0x100000 + 0x40*i + 0x4000**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
RDBAL	31:7	RW	0x0	Receive Descriptor Base Address Low. This register contains the lower bits of the 64-bit descriptor base address. The lower 7 bits are always ignored. The receive descriptor base address must point to a 128 byte-aligned block of data.

11.27.2.50 PCIE_RDBAH[0..255]

Address:	**0x100000 + 0x40*i + 0x4001**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
RDBAH	31:0	RW	0x0	Receive Descriptor Base Address [63:32]. This register contains the upper 32 bits of the 64-bit descriptor base address.

11.27.2.51 PCIE_RDLEN[0..255]

Address:	**0x100000 + 0x40*i + 0x4002**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
LEN	19:7	RW	0x20	Descriptor Ring Length. This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. It is expressed in units of 128 bytes. Minimum supported value for this field is 32 (128 descriptors of 32-bytes each). ***Note:*** Validated lengths up to 512 KB (32768 descriptors of 16 bytes or 16 K descriptors of 32 bytes).
Reserved	31:20	RSV	0x0	Reserved.

11.27.2.52 PCIE_TPH_RXCTRL[0..255]

Address:	**0x100000 + 0x40*i + 0x4003**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved1	4:0	RO	0x0	Reserved.
RxDescriptorTPHEN	5	RW	0b	Descriptor TPH EN. 0b = Hardware does not enable TPH for descriptor write-backs. 1b = Hardware enables TPH for all Rx descriptors written back into memory.
RxHeaderTPHEN	6	RW	0b	Rx Header TPH EN. 0b = Hardware does not enable TPH for Rx Headers. 1b = Hardware enables TPH for all received header buffers.

Address:	**0x100000 + 0x40*i + 0x4003**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
RxPayloadTPHEN	7	RW	0b	Payload TPH EN. 0b = Hardware does not enable TPH for Ethernet payloads. 1b = Hardware enables TPH for all Ethernet payloads written into memory. By default this bit is cleared (0b).
Reserved2	8	RO	0b	Reserved.
RXdescReadROEn	9	RW	1b	Rx Descriptor Read Relax Order Enable.
Reserved3	10	RO	0b	Reserved.
RXdescWBROen	11	RW	0b	Rx Descriptor Write Back Relax Order Enable. This bit must be 0b to enable correct functionality of the descriptors write back.
Reserved4	12	RO	0b	Reserved.
RXdataWriteROEn	13	RW	1b	Rx data Write Relax Order Enable.
Reserved5	14	RO	0b	Reserved.
RxRepHeaderROEn	15	RW	1b	Rx Split Header Relax Order Enable.
Reserved6	23:16	RO	0x0	Reserved.
CPUID	31:24	RO	0x0	Physical ID for TPH operated in DeviceSpecific mode. • Legacy TPH capable platforms — The device driver, upon discovery of the physical CPU ID and CPU bus ID, programs the *CPUID* field with the physical CPU and bus ID associated with this Rx queue. • TPH 1.0 capable platforms — The device driver programs a value, based on the relevant APIC ID, associated with this Rx queue. ***Note:*** This field is not operational in the FM10000, and has been made RO.

11.27.2.53 PCIE_RDH[0..255]

Address:	**0x100000 + 0x40*i + 0x4004**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
RDH	15:0	RO	0x0	Receive Descriptor Head. Holds the head pointer for the receive descriptor buffer in descriptor units. The *RDH* is controlled by hardware.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.54 PCIE_RDT[0..255]

This register contains the tail pointer for the receive descriptor buffer in descriptor units. Software writes the tail register to add receive descriptors to the hardware free list for the ring.

The tail pointer should be set to one descriptor beyond the last empty descriptor in host descriptor ring.

Address:	**0x100000 + 0x40*i + 0x4005**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
RDT	15:0	RW	0x0	Receive Descriptor Tail.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.55 PCIE_RXQCTL[0..255]

Address: 0x100000 + 0x40*i + 0x4006
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
ENABLE	0	RW	0b	Receive Queue Enable. When set, the *ENABLE* bit enables the operation of the specific receive queue.
Reserved	1	RSV	0b	Reserved.
VF	7:2	RW	0x0	Indicates to which VF this queue belongs. This register is RW when accessed from PF and RO when accessed from VF. Driver is allowed to change this field only while the *ENABLE* bit is clear.
OwnedByVF	8	RW	0b	Indicates this queue belongs to a Virtual Function. It overrides the *VF* field if cleared. This register is RW when accessed from PF and RO when accessed from VF. Driver is allowed to change this field only while the *ENABLE* bit is clear.
Reserved	31:9	RSV	0x0	Reserved.

11.27.2.56 PCIE_RXDCTL[0..255]

Address: 0x100000 + 0x40*i + 0x4007
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
MaxTime	7:0	RW	0x0	Max time before writing back descriptors. Time is in 512 * PCLK periods. A value of 0 means no delay. Only used on end of packet descriptors. For non-end of packet descriptors, the descriptors are held until a cache line is full and then written. This field can be modified only while the queue is disabled.
WriteBackImm	8	RW	0b	Defines if descriptor is written back immediately regardless how full the 64-byte cache line is. This field can be modified only while the queue is disabled.
DropOnEmpty	9	RW	0b	Defines if frames are dropped or held when a queue is empty. 0b = HOLD — Hold onto the frame if queue has no descriptors until descriptors become available. 1b = DROP — Drop the frame if queue has no descriptors.
WriteRSSHash	10	RW	0b	Defines, for 16-byte descriptor, if the RSS-HASH field or DGLORT/SGLORT is written to the descriptor. For 32-byte descriptors, both fields get written. 0b = DGLORT/SGLORT — Write DGLORT/SGLORT to descriptor. 1b = RSS_HASH — Write RSS Hash (32 bits) to descriptor. ** OBSOLETE, NOT IMPLEMENTED **
Reserved	31:11	RSV	0x0	Reserved.

11.27.2.57 PCIE_RXINT[0..255]

Address:	**0x100000 + 0x40*i + 0x4008**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
Interrupt	7:0	RW	0x0	Defines the MSI-X interrupt vector number used by this queue. This number is relative to the base of the interrupt blocks assigned to a VF or PF.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disable this interrupt.
Reserved	31:10	RSV	0x0	Reserved.

11.27.2.58 PCIE_SRRCTL[0..255]

Address:	**0x100000 + 0x40*i + 0x4009**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
BSIZEPACKET	7:0	RW	0x8	Receive Buffer Size for Packet Buffer. The value is in 256 bytes resolution. Value can be from 256 bytes to 16 KB. Default buffer size is 2 KB. This field should not be set to 0x0.
BSIZEHEADER	13:8	RW	0x4	Receive Buffer Size for Header Buffer. The value is in 64 bytes resolution. Value can be from 64 bytes to 1024 bytes. ***Note:*** The maximum supported header size is limited to 1023. Default buffer size is 256 bytes. Values above 1024 bytes are reserved for internal use only.
DESCTYPE	15:14	RW	00b	Define the descriptor type in Rx: 00b = No header split 01b = Header split — Header uses header buffer, unless it cannot fit. Packet split boundary defined in *PSRTYPE*. 10b = Small/large split — Small packets use small packet buffer, large packets use normal packet data buffer. 11b = Reserved. ***Note:*** If *DESCTYPE* is set to 1 or 2, *BSIZEHEADER* must be bigger than zero, and *BufferChainingEn* must be set to zero. Otherwise, no header split occurs.

Address: 0x100000 + 0x40*i + 0x4009
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description

PSRTYPE[0..13]	29:16	RW	0b	(14 x 1-bit)
				Packets are split according to the lowest-indexed entry that applies to the packet and that is enabled.
				For example, if bits 6 and 8 are set, then an IPv4 packet that is not TCP is split after the IPv4 header.
				This bit mask table enables or disables each type of header to be split. A value of 1b enables an entry.
				PSTYPE[0] = Split after TCP of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[1] = Split after UDP of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[2] = Split after IPv4 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[3] = Split after IPv6 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[4] = Split after L2 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[5] = Split after NVGRE/VXLAN header, but not inner packet.
				PSTYPE[6] = Split after TCP of outer header.
				PSTYPE[7] = Split after UDP of outer header.
				PSTYPE[8] = Split after IPv4 of outer header.
				PSTYPE[9] = Split after IPv6 of outer header.
				PSTYPE[10] = Split after L2 of outer header.
				All other values are reserved.

LoopbackSuppress	30	RW	0b	Defines if Loopback Suppress (compare SGLORT) is enabled or not.
BufferChainingEn	31	RW	0b	Buffer Chaining enable. 0b = Any packet longer than the data buffer size is terminated with a TOO_BIG error status in Rx descriptor write-back. The remainder of the frame is not posted to host, it is silently dropped. 1b = A packet can be spread over more than one single receive data buffer.

11.27.2.59 PCIE_QPRC[0..255]

Address: 0x100000 + 0x40*i + 0x400A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
PRC	31:0	RO	0x0	Number of packets received for the queue.

11.27.2.60 PCIE_QPRDC[0..255]

Address: 0x100000 + 0x40*i + 0x400B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
PRDC	31:0	RO	0x0	Total number of receive packets dropped for the queue. Packets can be dropped for the following reasons: • Rx queue is disabled in the RXQCTL[n] register. • No free descriptors in the Rx queue and set to drop in this case. • VLAN membership violation (also counted in PCIE_STATS_VLAN_DROP). • Loopback suppress (also counted in PCIE_STATS_LOOPBACK_DROP). • Packet too small (smaller than 48-bytes, though the switch will never forward such packets to the PEP). • Rx path disabled (i.e., DMA_CTRL.*RxEnable* bit is cleared).

11.27.2.61 PCIE_QBRC_L[0..255]

Address:	**0x100000 + 0x40*i + 0x400C**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BRC_L	31:0	RO	0x0	Lower 32 bits of the statistic counter. The QBRC_L[n] and QBRC_H[n] registers make up a logical 48-bit counter of received bytes that were posted to the programmed Rx queues of the packets counted by QPRC[n]. The counter counts all bytes from DMAC (included) to TIME TAG (excluded).

11.27.2.62 PCIE_QBRC_H[0..255]

Address:	**0x100000 + 0x40*i + 0x400D**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BRC_H	15:0	RO	0x0	Higher 16 bits of the statistic counter described in QBRC_L.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.63 PCIE_RX_SGLORT[0..255]

Address:	**0x100000 + 0x40*i + 0x400E**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SGlort	15:0	RW	0x0	Source GLORT to use when checking for loopback suppress.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.64 PCIE_TDBAL[0..255]

Address:	**0x100000 + 0x40*i + 0x8000**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Reserved	6:0	RSV	0x0	Reserved.
TDBAL	31:7	RW	0x0	Transmit Descriptor Base Address Low. This register contains the lower bits of the 64-bit descriptor base address. The lower seven bits are ignored. The Transmit Descriptor Base Address must point to a 128 byte-aligned block of data.

11.27.2.65 PCIE_TDBAH[0..255]

Address: **0x100000 + 0x40*i + 0x8001**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
TDBAH	31:0	RW	0x0	Transmit Descriptor Base Address [63:32]. This register contains the upper 32 bits of the 64-bit Descriptor base address.

11.27.2.66 PCIE_TDLEN[0..255]

Address: **0x100000 + 0x40*i + 0x8002**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
LEN	19:7	RW	0x4	Descriptor Ring Length. This register sets the number of bytes allocated for descriptors in the circular descriptor buffer. It is expressed in units of 128 bytes. Minimum supported value for this field is 4 (32 descriptors of 16 bytes each). ***Note:*** Validated lengths up to 512 KB (32768 descriptors of 16 bytes).
Reserved	31:20	RSV	0x0	Reserved.

11.27.2.67 PCIE_TPH_TXCTRL[0..255]

Address: **0x100000 + 0x40*i + 0x8003**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	4:0	RSV	0x0	Reserved.
TxDescriptorTPHEN	5	RW	0b	Descriptor TPH Enable. 0b = Hardware does not enable TPH for descriptor write-backs. 1b = Hardware enables TPH for all Tx descriptors written back into memory. This bit is cleared by default.
Reserved	8:6	RSV	000b	Reserved.
TXdescRDROEn	9	RW	1b	Tx Descriptor Read Relax Order Enable.
Reserved	10	RSV	0b	Reserved.
TXdescWBROen	11	RW	1b	Relax Order Enable of Tx Descriptor as well as head pointer write back (when set).
Reserved	12	RSV	0b	Reserved.
TXDataReadROEn	13	RW	1b	Tx Data Read Relax Order Enable.
Reserved	23:14	RSV	0x0	Reserved.
CPUID	31:24	RO	0x0	Physical ID for TPH operated in DeviceSpecific mode. • Legacy TPH capable platforms — The device driver, upon discovery of the physical CPU ID and CPU bus ID, programs the *CPUID* field with the physical CPU and bus ID associated with this Tx queue. • TPH 1.0 capable platforms — The device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. • This field is not operational in the FM10000, and has been made RO.

11.27.2.68 PCIE_TDH[0..255]

Address: 0x100000 + 0x40*i + 0x8004
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
TDH	15:0	RW	0x0	Transmit Descriptor Head. This register contains the head pointer for the transmit descriptor ring. It is expressed in descriptor units. Hardware controls this pointer. The values in these registers might point to descriptors that are still not in the host memory. As a result, the host cannot rely on these values in order to determine which descriptor to release. The only time that software should write to this register is after a reset (hardware reset or CTRL.*RST*) and before enabling the transmit function (TXDCTL.*ENABLE*). If software were to write to this register while the transmit function was enabled, the on-chip descriptor buffers might be invalidated and the hardware could become confused.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.69 PCIE_TDT[0..255]

Address: 0x100000 + 0x40*i + 0x8005
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
TDT	15:0	RW	0x0	Transmit Descriptor Tail. This register contains the tail pointer for the transmit descriptor ring. It is expressed in descriptor units. Software writes the tail pointer to add more descriptors to the transmit ready queue. Hardware attempts to transmit all packets referenced by descriptors between head and tail.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.70 PCIE_TXDCTL[0..255]

This register controls the fetching and write-back of transmit descriptors. The two threshold values are used to determine when descriptors is read from host memory and only applies in the pull descriptor model.

Address: 0x100000 + 0x40*i + 0x8006
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
PTHRESH	6:0	RO	0x0	** OBSOLETE, NOT IMPLEMENTED ** Pre-Fetch Threshold. Controls when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed transmit descriptors the has in its on-chip buffer. If this number drops below *PTHRESH*, the algorithm considers pre-fetching descriptors from host memory. However, this fetch does not happen unless there are at least *HTHRESH* valid descriptors in host memory to fetch. ***Note:*** *HTHRESH* should be given a non-zero value each time *PTHRESH* is used.
HTHRESH	13:7	RO	0x0	** OBSOLETE, NOT IMPLEMENTED ** Host Threshold.

Address: 0x100000 + 0x40*i + 0x8006
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
ENABLE	14	RW	0b	Transmit Queue Enable. When set, this bit enables the operation of a specific transmit queue: Default value for all queues is 0b.
Reserved	15	RSV	0b	Reserved.
MaxTime	27:16	RW	0x0	Max idle time before posting last descriptor write back and an interrupt. Time is in 512 * PCLK periods. The RS overwrites the write back. A value of 0b means no delay.
PushDesc	28	RW	0b	Defines if this queue has the descriptors written into the controller (push model) or if the controller read descriptors from host memory (pull model). 0b = PULL — Descriptors read by controller from host memory. 1b = PUSH — Descriptors written by host into controller.
Reserved	31:29	RSV	000b	Reserved.

11.27.2.71 PCIE_TXQCTL[0..255]

This register is RW when accessed from PF and RO when accessed from VF.

Address: 0x100000 + 0x40*i + 0x8007
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
VF	5:0	RW	0x1	Indicates to which VF this queue belongs. This Field is RW when accessed from PF and RO when accessed from VF. Driver is allowed to change this field only while the *ENABLE* bit of the queue is clear.
OwnedByVF	6	RW	0b	Indicates this queue belongs to a Virtual Function. It invalidates the *VF* field if cleared. This Field is RW when accessed from PF and RO when accessed from VF. Driver is allowed to change this field only while the *ENABLE* bit of the queue is clear.
PC	9:7	RW	000b	Defines the Priority Flow Control Class (Pause Class). This Field is RW when accessed from PF and RO when accessed from VF.
TC	15:10	RW	0x0	Defines the shaping group to which this queue belongs (Traffic Class). Meaningful only when *UnlimitedBW* bit is cleared.
VID	27:16	RW	0x0	Defines the default VLAN ID for the queue if none supplied. This value could be 0b. If this default VLAN ID is set to 0b and the host set to 0b, the switch assigns a default VLAN ID.
UnlimitedBW	28	RW	0b	This bit provides the ability to create a 65th TC with unlimited bandwidth. Usage model envisaged is for the PF's queue when there are 64 enabled VFs, each one with its own TC. Setting this bit makes the *TC* field meaningless.
PushModeDis	29	RW	0b	This bit provides the ability for the PF to disallow the VF to use the queue in push mode. This is a chicken bit in case a security issue is found in push mode.
Reserved	31:30	RSV	00b	Reserved.

11.27.2.72 PCIE_TXINT[0..255]

Address:	0x100000 + 0x40*i + 0x8008			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Interrupt	7:0	RW	0x0	Defines the MSI-X interrupt vector number used by this queue. This number is relative to the base of the interrupt blocks assigned to a VF or PF.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disable this interrupt.
Reserved	31:10	RSV	0x0	Reserved.

11.27.2.73 PCIE_QPTC[0..255]

Address:	0x100000 + 0x40*i + 0x8009			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT_MEM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PTC	31:0	RO	0x0	Number of packets transmitted for the queue. A packet is considered as transmitted if it is was forwarded to the Ethernet switch for switching and transmission to the network.

11.27.2.74 PCIE_QBTC_L[0..255]

Address:	0x100000 + 0x40*i + 0x800A			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT_MEM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BTCL	31:0	RO	0x0	Lower 32 bits of the statistic counter. The QBTC_L and QBTC_H registers make up a logical 48-bit counter of transmitted bytes of the packets counted by the matched QPTC counter. These registers count all bytes in the packets from the DMAC field (included) through the TIME TAG field (excluded). These registers must be accessed as two consecutive 32-bit entities while the QBTC_L register is read first, or a single 64-bit read cycle.

11.27.2.75 PCIE_QBTC_H[0..255]

Address:	0x100000 + 0x40*i + 0x800B			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT_MEM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BTCH	15:0	RO	0x0	Higher 16 bits of the statistic counter described in QBTC_L.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.76 PCIE_TQDLOC[0..255]

Address: 0x100000 + 0x40*i + 0x800C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Base	15:0	RW	0x0	Defines the base location of the descriptor area in 64-byte chunks. Only 3 allocation schemes are supported where Base for Q index i is set to: ix8 When DMA_CTRL.*MaxNumOfQs* is set to 00b (256 Qs). ix16 When DMA_CTRL.*MaxNumOfQs* is set to 01b (128 Qs). ix32 When DMA_CTRL.*MaxNumOfQs* is set to 10b (64 Qs). This field is read only when accessed from VF.
Size	19:16	RW	0x0	Defines the size of the descriptor area in log2(N). Minimum is 2 (4 descriptors) and maximum is 8 (256 descriptors). Only 3 Sizes supported: 5 (32 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 00b (256 Qs). 6 (64 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 01b (128 Qs). 7 (128 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 10b (64 Qs). This field is read only when accessed from VF.
Reserved	31:20	RSV	0x0	Reserved.

11.27.2.77 PCIE_TX_SGLORT[0..255]

Address: 0x100000 + 0x40*i + 0x800D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
SGlort	15:0	RW	0x0	Source GLORT to use when sending a packet to switch for this Q.
Reserved.	31:16	RSV	0x0	Reserved.

11.27.2.78 PCIE_PFVTCTL[0..255]

Address: 0x100000 + 0x40*i + 0x800E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
FTagDescEnable	31:0	RW	0x0	Indicates if a Tx Queue is allowed to send packets with FTAG or not.

11.27.2.79 PCIE_TX_DESC[0..255][0..255][0..3]

Used to store data and control. Indexing is [queue#,desc#,word#].

Address: 0x100000 + 0x400*k + 0x4*j + 0x1*i + 0x40000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Data	31:0	WO	0x0	

11.27.2.80 PCIE_PBACL[0..71]

MSI-X bit pending array. Indexes 0..7 are for physical function, and 8..71 for virtual functions. The PBACL[8..71] are also visible through VF BAR2/3.

Address: 0x100000 + 0x1*i + 0x10000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
PendingBits	31:0	CW1	0x0	Indicates a pending bit. Writings 1's clears corresponding bits. Reading returns current value.

11.27.2.81 PCIE_INT_MAP[0..15]

Defines the interrupt number for each source (PF). The index to this table indicates the type of interrupt and the output is the index to the MSI-X vector table. For interrupt allocation, refer to Section 6.12, “Interrupts and Faults”.

The following vectors are used:

- PCIE_INT_MAP[0] = Mailbox
- PCIE_INT_MAP[1] = Fault
- PCIE_INT_MAP[2] = Switch up/down event
- PCIE_INT_MAP[3] = Switch interrupt
- PCIE_INT_MAP[4] = SRAM interrupt (must be mapped to MSI-X vector 0)
- PCIE_INT_MAP[5] = VFLR interrupt
- PCIE_INT_MAP[6] = Max hold time interrupt
- PCIE_INT_MAP[7] = Visa trap interrupt

The PCIE_INT_MAP[8..15] are not used in this design.

Address: 0x100000 + 0x1*i + 0x10080
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Interrupt	7:0	RW	0x0	Interrupt number.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disabled interrupt. Whenever an INT_MAP register is transitioned in/out of Timer 3 (disabled) state, the driver either disables the interrupt before, or cleans up any associated cause thereafter.
Reserved	31:10	RSV	0x0	Reserved.

11.27.2.82 PCIE_MSIX_VECTOR[0..767]

This table should normally be accessed through the BAR2/3 map and not through the normal PF register map (BAR0/1).

When accessed through PF BAR2/3 only entries [0..255] are visible (PF reserved area). When accessed through VF BAR 2/3 only the portion allowed to this VF is visible (per VF reserved area).

The structure of this table is defined in *PCI Local Bus Specification 3.0* and *PCI Express Bus Specification 3.0*.

Address: 0x100000 + 0x4*i + 0x11000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
Addr	63:0	RW	0x0	Message Signal Interrupt address. Lower 2 bits must be set to 00b.
Data	95:64	RW	0x0	Message Signal Interrupt data.
Mask	96	RW	1b	When this bit is set, the function is prohibited from sending a message using this MSI-X Table entry. However, any other MSI-X Table entries programmed with the same vector is still capable of sending an equivalent message unless they are also masked. This bit's state after reset is 1b (entry is masked). This bit is kept asserted to 1b until writes to *Addr* and *Data* fields are completed.
Reserved	127:97	RSV	0x0	Reserved.

11.27.2.83 PCIE_INT_CTRL

Address: 0x100000 + 0x12000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
NextVector	9:0	RW	0x0	Defines/holds next vector to process. Software could update any time, hardware updates as it walks through vectors.
EnableModerator	10	RW	0b	Enable interrupt moderators. This bit is normally set to 1b. It could be temporarily set to 0b to change the link lists of timers in a non-orderly manner (disabling moderator is not required if software proceed in order when changing link lists). If set to 0b, the software waits 50 ns before making changes to the link list.
Reserved	31:11	RSV	0x0	Reserved.

11.27.2.84 PCIE_ITR[0..767]

The entire set of 768 registers is visible from PF, and a portion is visible through VF.

Address: 0x100000 + 0x1*i + 0x12400
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
Interval0	11:0	RW	0x0	Defines minimum interval between interrupts, timer 0.
Interval1	23:12	RW	0x0	Defines minimum interval between interrupts, timer 1
Timer0Expired	24	RO	0b	Indicates timer 0 has expired.

Address: 0x100000 + 0x1*i + 0x12400
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
Timer1Expired	25	RO	0b	Indicates timer 1 has expired.
Pending0	26	SW1	0b	Indicates a pending event is waiting for time 0 expiration. 0b = No effect. 1b = Sets the interrupt.
Pending1	27	SW1	0b	Indicates a pending event is waiting for time 1 expiration. 0b = No effect. 1b = Sets the interrupt.
Pending2	28	SW1	0b	Indicates a pending event is waiting for time 2 expiration. 0b = No effect. 1b = Sets the interrupt.
AutoMask	29	RW	0b	Defines if the interrupt is auto-masked by hardware. If set to 1b, the hardware automatically blocks interrupts (*Mask* set to 1b) when the interrupt is posted, and thus blocks future interrupts until the software writes the clear mask command in the *Mask* field.
Mask	31:30	RW	01b	Control masking of this interrupt: On write: 00b = Do not change mask status. 01b = Set mask (block interrupts). 10b = Clear mask (unblock interrupts). 11b = Reserved. On read: 00b = Interrupt not blocked. 01b = Interrupt blocked. 10b = Reserved. 11b = Reserved.

11.27.2.85 PCIE_ITR2[0..767]

Address: 0x100000 + 0x2*i + 0x12800
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
NextVector	9:0	RW	0x0	Defines next vector in the linked list.
Reserved	31:10	RSV	0x0	Reserved.
LastPosting	47:32	RO	0x0	Hold last time an MSI-X interrupt was posted.
Reserved	63:48	RSV	0x0	Reserved.

11.27.2.86 PCIE_IP

Indicates interrupts from PCIe. Resets on COLD_RESET. Register not accessible while the PEP is on WARM_RESET.

Address:	**0x100000 + 0x13000**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD**

Field Name	Bit(s)	Type	Default	Description
HotReset	0	CW1	0b	Indicates a hot reset (a.k.a. PCIe in-band reset) or a PCIe link down occurred.
DeviceStateChange	1	CW1	0b	Indicates a device state change occurred (e.g., D0 to D3hot or D3hot to D0).
Mailbox	2	CW1	0b	Global Mailbox Interrupt pending. Read PCIE_GMBX register to recover source of interrupt.
VPD_Request	3	CW1	0b	Indicates a VPD request from the host.
SramError	4	RO	0b	Indicates a correctable or an uncorrectable ECC/parity error occurred on an SRAM of the PEP. Read PCIE_SRAM_IP register to retrieve the concerned block. The bit is kept asserted until all asserted bits in PCIE_SRAM_IP are cleared by the appropriate reset or by CW1.
PFLR	5	CW1	0b	Indicated a PFLR occurred.
DataPathReset	6	CW1	0b	Indicates a Data Path reset occurred.
OutOfReset	7	CW1	0b	Indicates the PEP has left any of its reset state: cold, warm, hot, PFLR, or data path reset.
NotInReset	8	RO	1b	Allows the Switch Manager and the host to detect that the PEP is in one of its reset states. ***Note:*** If PEP is in WARM reset, reads by (a non-local) SM return 0x0. Since unlike other bits in this register this bit is not reset by SM, it can be polled by host to detect the PEP has exited reset.
Timeout	9	CW1	0b	Indicates an access to the PEP from DBI has timed out.
VFLR	10	CW1	0b	Indicates a VFLR occurred.
Reserved	31:11	RSV	0x0	Reserved.

11.27.2.87 PCIE_IM

Resets on COLD_RESET. Register not accessible while the PEP is on WARM_RESET.

Address:	**0x100000 + 0x13001**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD**

Field Name	Bit(s)	Type	Default	Description
Mask	7:0	RW	0xFF	Mask for PCIE_IP cause register toward the SM. Setting to 1 blocks interrupt.
Reserved1	8	RW	1b	This bit has no effect since it corresponds to *NotInReset* bit in PCIE_IP, which never generates interrupt.
Mask9_10	10:9	RW	11b	Mask for bits 9 and 10 in PCIE_IP cause register toward the SM. Setting to '1' blocks interrupt.
Reserved	31:11	RSV	0x0	Reserved.

11.27.2.88 PCIE_IB

Resets on COLD_RESET. Register not accessible while the PEP is on WARM_RESET.

Address:	**0x100000 + 0x13002**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD**

Field Name	Bit(s)	Type	Default	Description
Mask	7:0	RW	0xFF	Mask for PCIE_IP cause registers toward the BSM. Setting to 1b blocks interrupt.
Reserved1	8	RW	1b	This bit has no effect since it corresponds to *NotInReset* bit in PCIE_IP, which never generates interrupt.
Mask9_10	10:9	RW	11b	Mask for bits 9 and 10 in PCIE_IP cause register toward the SM. Setting to ‘1’ blocks interrupt.
Reserved	31:9	RSV	0x0	Reserved.

11.27.2.89 PCIE_SRAM_IP

Indicates correctable and uncorrectable ECC error events on SRAMs. UE_ prefix is used for Uncorrectable Error, and CE_ for Correctable Error.The errors are reported per functional block. Resets on COLD_RESET.

Address:	**0x100000 + 0x13003**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD**

Field Name	Bit(s)	Type	Default	Description
UE_PCW	0	CW1	0b	Uncorrectable error in the PCW block and registers. Requires a PCIe in-band reset.
Reserved	1	RSV	0b	Reserved.
UE_FUM	2	CW1	0b	Uncorrectable error in the FUM block, excepted to Mailbox and Interrupt memories. Requires a PCIe in-band reset.
UE_MBX_INT	3	CW1	0b	Uncorrectable error on Mailbox and Interrupt memories. Requires a PCIe in-band reset.
UE_RHI	4	CW1	0b	Uncorrectable error in the RHI block and registers. Requires a Data Path reset.
UE_THI	5	CW1	0b	Uncorrectable error in the THI block and registers. Requires a Data Path reset.
UE_PP	6	CW1	0b	Uncorrectable error in the RPP or TPP block and registers. Requires a Data Path reset.
UE_PCA	7	CW1	0b	Uncorrectable error in PCA block and registers. Requires a Data Path reset.
CE_PCW	8	CW1	0b	Correctable error in PCW block and registers.
Reserved	9	RSV	0b	Reserved.
CE_FUM	10	CW1	0b	Correctable error in FUM block and registers.
CE_MBX_INT	11	CW1	0b	Correctable error on Mailbox and Interrupt memories.
CE_RHI	12	CW1	0b	Correctable error in RHI block and registers.
CE_THI	13	CW1	0b	Correctable error in THI block and registers.
CE_PP	14	CW1	0b	Correctable error in RPP or TPP block and registers.
CE_PCA	15	CW1	0b	Correctable error in PCA block and registers.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.90 PCIE_SRAM_IM

Resets on COLD_RESET.

Address:	**0x100000 + 0x13004**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Mask	15:0	RW	0xFFFF	Mask for PCIE_SRAM_IP cause register. Setting to 1b blocks interrupt.
Reserved	31:16	RSV	0x0	Reserved.

11.27.2.91 PCIE_VLAN_TABLE[0..64][0..127]

Index j in the range 1..64 is for VF, index j=0 is for PF.

Address:	**0x100000 + 0x80*j + 0x1*i + 0x14000**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VSI_Membership[0..31]	31:0	RW	0b	(32 x 1-bit) Each bit n in register i affects packets with VLAN tags equal to 32xi+n for the VF/PF indicated via j. 128 PCIE_VLAN_TABLE registers compose a table of 4096 bits that cover all possible VLAN tags. Each bit when set enables packets associated with this VLAN to pass. Each bit when cleared results in the packets associated with this VLAN being dropped.

11.27.2.92 PCIE_MBMEM[0..2047]

Mailboxes memories. The first 1024 entries are used for VF mailboxes (64 x 64-bytes). VF sees only one of these mailbox at the time. The next 1024 entries are free and intended for SM/PF communication.

Address:	**0x100000 + 0x1*i + 0x18000**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Data	31:0	RW	0x0	Mailbox data.

11.27.2.93 PCIE_MBX[0..63]

Address: **Atomicity:** **Reset Domains:**	**0x100000 + 0x1*i + 0x18800** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Owner	0	CW1	1b	Hardware semaphore. The requester reads this field to claim ownership over the mailbox. 0b = Ownership has already been granted to another software entity, and it will try again later. 1b = Requestor has been granted ownership. The owner writes to this field to release ownership. 0b = No effect. 1b = Release ownership. ** Do not use this bit. Hardware semaphore mechanism is not operational. Use a lockless mailbox mechanism instead, where the mailbox memory is split in two halves, one half for the PF and one half for the VF **
Req	1	WO	0b	Message to VF ready. Setting this bit causes an interrupt to the relevant VF (latched in PCIE_VFMBX).This bit always reads as 0b.
Ack	2	WO	0b	Message from VF received. Setting this bit causes an interrupt to the relevant VF (latched in PCIE_VFMBX). This bit always reads as 0b.
PFReqInterrupt	3	CW1	0b	VF has posted a message for PF. Setting this bit clears the interrupt.
PFAckInterrupt	4	CW1	0b	VF indicates message from PF was received. Setting this bit clears the interrupt.
InterruptEnable	6:5	RW	00b	Mailbox interrupt enable from the VF. Read as 0 if disabled. Read as 1 if enabled. 00b = Do not change the mode. 01b = Enable mailbox interrupt from the VF. 10b = Disable mailbox interrupt from the VF. 11b = Do not change the mode.
Reserved	31:7	RSV	0x0	Reserved.

11.27.2.94 PCIE_MBICR[0..1]

Address: **Atomicity:** **Reset Domains:**	**0x100000 + 0x1*i + 0x18840** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VF2PFInt	31:0	RO	0x0	Mirrors of 'PCIE_MBX[vf].*PFReqInterrupt* OR PCIE_MBX[vf].*PFAckInterrupt*', encoded PCIE_MBICR[vf/32].*VF2PFInt*[vf%32].

11.27.2.95 PCIE_GMBX

Address:	**0x100000 + 0x18842**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**

Field Name	Bit(s)	Type	Default	Description
Owner	0	CW1	1b	Hardware semaphore. The requester reads this field to claim ownership over the mailbox. 0b = Ownership has already been granted to another software entity, and it will try again later. 1b = Requestor has been granted ownership. The owner writes to this field to release ownership. 0b = No effect. 1b = Release ownership. ** Do not use this bit. Hardware semaphore mechanism is not operational. Use a lockless mailbox mechanism instead, where the mailbox memory is split in two halves, one half for the PF and one half for the SM **
GlobalReq	1	WO	0b	Request from PF to SM is ready. Setting this bit causes a global interrupt to be latched in PCIE_IP. This bit always reads as 0b.
GlobalAck	2	WO	0b	Message from SM to PF was read. Setting this bit causes a global interrupt to be latched in PCIE_IP. This bit always reads as 0b.
PFReqInterrupt	3	CW1	0b	Set when message from SM to PF is ready.
PFAckInterrupt	4	CW1	0b	Set when message from PF to SM has been read by SM.
PFInterruptEnable	6:5	RW	00b	Controls if the global mailbox host interrupts are enabled. Read as 0 if disabled. Read as 1 if enabled. 00b = Do not change the mode. 01b = Enable global mailbox host interrupts. 10b = Disable global mailbox host interrupts. 11b = Do not change the mode.
PFReq	7	WO	0b	Message from SM to PF is ready. Setting this bit causes an interrupt to be latched in PCIE_EICR. This bit is always read as 0b.
PFAck	8	WO	0b	Message fro PF to SM was read. Setting this bit causes an interrupt to the be latched in PCIE_EICR. This bit always reads as 0b.
GlobalReqInterrupt	9	CW1	0b	Set when message from PF to SM is ready.
GlobalAckInterrupt	10	CW1	0b	Set when message from SM to PF has been read by PF.
GlobalInterruptEnable	12:11	RW	00b	Controls if the global mailbox global interrupts are enabled. Read as 0 if disabled. Read as 1 if enabled. 00b = Do not change the mode. 01b = Enable global interrupts. 10b = Disable global interrupts. 11b = Do not change the mode.
Reserved	31:13	RSV	0x0	Reserved.

11.27.2.96 PCIE_PFVFLRE

Address:	**0x100000 + 0x18844**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DetectEvent[0..63]	63:0	RO	0b	(64 x 1-bit) Indication of FLR occurrence on VF[n]. These bits are accessible only to the PF and are cleared by writing 0x1 to the matched bit in the PFVFLREC registers.

11.27.2.97 PCIE_PFVFLREC

Address:	**0x100000 + 0x18846**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ClearEvent[0..63]	63:0	CW1	0b	(64 x 1-bit) Clear FLR event on VF[n]. Writing a 1b to bit 'i' clears the FLR event on VF[n] indicated in the PFVFLRE[n] registers.

11.27.2.98 PCIE_TEST_CFG0

Address:	**0x100000 + 0x18849**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PEPLoopback	0	RW	0b	PEP level loopback from Tx to Rx. Useful for PEP debugging.
DPRStatsClearEn	1	RW	0b	Debug feature that enables clearing of the per-queue stats QPTC, QBTC, QPRDC, QPRC and QBRC on datapath reset events. 0b = Disable clearing. 1b = Enable clearing.
Reserved	31:2	RSV	0x0	Reserved.

11.27.2.99 PCIE_INT_SRAM_CTRL

itr2=0, itr=1, msix_vec_2=2, msix_vec_1=3, msix_vec_0=4, vfint_map=5, txint=6, rxint=7, mbmem=8.

Address:	**0x100000 + 0x18850**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..8]	17:0	RW	00b	(9 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..8]	26:18	CW1	0b	(9 x 1-bit)
UErr[0..8]	35:27	CW1	0b	(9 x 1-bit)
BistDonePass[0..8]	44:36	RO	0b	(9 x 1-bit)
BistDoneFail[0..8]	53:45	RO	0b	(9 x 1-bit)
Reserved	63:54	RSV	0x0	Reserved.

11.27.2.100 PCIE_FUM_SRAM_CTRL

qbtc_h=0, qbtc_l=1, qptc=2, qbrc_h=3, qbrc_l=4, qprc=5, qprdc=6.

Address:	**0x100000 + 0x18852**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..6]	20:14	CW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	CW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	RO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	RO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.27.2.101 PCIE_PCA_SRAM_CTRL

qmap=0.

Address: 0x100000 + 0x18854
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
ErrWrite	1:0	RW	00b	00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr	2	CW1	0b	
UErr	3	CW1	0b	
BistDonePass	4	RO	0b	
BistDoneFail	5	RO	0b	
Reserved	31:6	RSV	0x0	Reserved.

11.27.2.102 PCIE_PP_SRAM_CTRL

srrctl=0, vlan_vf_table=1, vlan_pf_table=2, txfifo=3, tso_meta=4, tso1hdr=5, tso0hdr=6, rxfifo=7, reta=8, rssrk0=9, rssrk1=10, rssrk2=11, rssrk=12, rssrk=13.

Address: 0x100000 + 0x18858
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..13]	27:0	RW	00b	(14 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..13]	41:28	CW1	0b	(14 x 1-bit)
UErr[0..13]	55:42	CW1	0b	(14 x 1-bit)
BistDonePass[0..13]	69:56	RO	0b	(14 x 1-bit)
BistDoneFail[0..13]	83:70	RO	0b	(14 x 1-bit)
Reserved	127:84	RSV	0x0	Reserved.

11.27.2.103 PCIE_PCW_SRAM_CTRL

radm_dataq1=0, radm_dataq0=1, radm_hdrq1=2, radm_hdrq0=3, sotram=4, retryram1=5, retryram0=6.

Address:	**0x100000 + 0x1885C**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..6]	20:14	CW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	CW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	RO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	RO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.27.2.104 PCIE_RHI_SRAM_CTRL1

dw1=0, dw0=1, rdp=2,desc3=3, desc2=4, desc1=5, desc0=6.

Address:	**0x100000 + 0x1885E**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..6]	13:0	RW	00b	(7 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..6]	20:14	CW1	0b	(7 x 1-bit)
UErr[0..6]	27:21	CW1	0b	(7 x 1-bit)
BistDonePass[0..6]	34:28	RO	0b	(7 x 1-bit)
BistDoneFail[0..6]	41:35	RO	0b	(7 x 1-bit)
Reserved	63:42	RSV	0x0	Reserved.

11.27.2.105 PCIE_RHI_SRAM_CTRL2

Rdt=0, rdlen=1, rdh=2, rdbal=3, rdbah=4.

Address: 0x100000 + 0x18860
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..4]	9:0	RW	00b	(5 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..4]	14:10	CW1	0b	(5 x 1-bit)
UErr[0..4]	19:15	CW1	0b	(5 x 1-bit)
BistDonePass[0..4]	24:20	RO	0b	(5 x 1-bit)
BistDoneFail[0..4]	29:25	RO	0b	(5 x 1-bit)
Reserved	31:30	RSV	00b	Reserved.

11.27.2.106 PCIE_THI_SRAM_CTRL1

dw1=0, dw0=1, desc3=2, desc2=3 desc1=4, desc0=5, rate=6, maxcredit=7, credit=8, rsvd=9, cb1=10, cb0=11, desc_info1=12, desc_info0=13.

Address: 0x100000 + 0x18864
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..13]	27:0	RW	00b	(14 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..13]	41:28	CW1	0b	(14 x 1-bit)
UErr[0..13]	55:42	CW1	0b	(14 x 1-bit)
BistDonePass[0..13]	69:56	RO	0b	(14 x 1-bit)
BistDoneFail[0..13]	83:70	RO	0b	(14 x 1-bit)
Reserved	127:84	RSV	0x0	Reserved.

11.27.2.107 PCIE_THI_SRAM_CTRL2

Txdctl=0, tdh=1, tdt=2, tdlen=3, tdbah=4, tdbal=5.

Address:	**0x100000 + 0x18868**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..5]	11:0	RW	00b	(6 x 2-bits) 00b = Nothing 01b = Correctable 10b = Correctable 11b = Uncorrectable
CErr[0..5]	17:12	CW1	0b	(6 x 1-bit)
UErr[0..5]	23:18	CW1	0b	(6 x 1-bit)
BistDonePass[0..5]	29:24	RO	0b	(6 x 1-bit)
BistDoneFail[0..5]	35:30	RO	0b	(6 x 1-bit)
Reserved	63:36	RSV	0x0	Reserved.

11.27.2.108 PCIE_FPGA_REVID

Register present only at PEP validation stage, not in silicon. Hardware default to this register changes upon each release. It encodes the work week drop tag, EAS version, and FPGA version.

Address:	**0x100000 + 0x1886A**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
DayInWW	3:0	RO	0x0	Day in the work week. 0000b = Sunday 0001b = Monday 0010b = Tuesday 0011b = Wednesday 0100b = Thursday 0101b = Friday 0110b = Saturday
WW	11:4	RO	0x0	Work week expressed in 2 digits. Values in the range 0x01-0x52.
Year	15:12	RO	0x0	Last digit of the 201x year. Values in the range 0x0-0x9.
EASVersion	27:16	RO	0x0	For instance EAS revision 1.35 is encoded as 0x135.
HWVersion	31:28	RO	0x0	

11.27.2.109 PCIE_TIMEOUT_CFG

Address:	**0x100000 + 0x1886B**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MemInitTimer	15:0	RW	0x4064	Defines the maximum duration in PCLK cycles units of the memory init sequence. Default setting is maximum 131 µs for the slowest clock case.
CSRAccessTimeout	23:16	RW	0x50	Defines the maximum delay in PCLK cycles units for getting ack on CSR access from host or SM/BSM. Beyond this time, the PEP returns 0xFFFFFFFF on reads, writes are aborted, and the address is logged in the PCIE_FUM_FAULT register. Default setting is 80 cycles.
Reserved	31:24	RSV	0x0	Reserved.

11.27.2.110 PCIE_LVMMC

Address:	**0x100000 + 0x18880**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
TxDropHDRLEN0	0	CR	0b	Bit set when HDRLEN equals 0 and MSS greater than 0. The queue is disabled on such events.
TxDropMSS0	1	CR	0b	Bit set when HDRLEN greater than 0 and MSS equals 0. The queue is disabled on such events.
TxDropHDRLENGtPKTLEN	2	CR	0b	Bit set when HDRLEN greater than Packet Size. The queue is disabled on such events.
TxDropMinPkt	3	CR	0b	Bit set when Packet Size smaller than 17 bytes (or 25 bytes if FTAG included). The packet is silently dropped and the queue remains enabled.
TxDropMaxPkt	4	CR	0b	Bit set when Packet Size greater than 15 KB. The queue is disabled on such events.
TxDropMaxHdr	5	CR	0b	Bit set when HDRLEN greater than 192 bytes. The queue is disabled on such events.
TxDropMinHdr	6	CR	0b	Bit set when HDRLEN smaller than 54 bytes. The queue is disabled on such events.
TxMinTDLEN	7	CR	0b	Bit set when TDLEN smaller than 32. The attempt to set TDLEN below 32 is ignored.
TxDropBUFLEN0First	8	CR	0b	Bit set when BUFLEN equals 0 on first descriptor. The queue is disabled on such events.
TxDropBUFLEN0Mid	9	CR	0b	Bit set when BUFLEN equals 0 on a middle descriptor (not the first descriptor). The queue is disabled on such events.
TxDropMaxDesc	10	CR	0b	Bit set when the number of descriptors for a single packet or for a TSO is greater than the internal descriptor cache size. The queue is disabled on such events.
TxDropFTAGInsDis	11	CR	0b	Bit set when FTAG bit is set in descriptor regardless of whether PFVTCTL[Q].*FtagDescEnable* bit is cleared. The queue is disabled on such events.
TxDropMinMSS	12	CR	0b	Bit set when MSS smaller than MinMSS. The queue is disabled on such events.

Address: 0x100000 + 0x18880
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
TxDropMaxTSO	13	CR	0b	Bit set when TSO greater than (256 KB -1). The queue is disabled on such events.
TxDropURCAData	14	CR	0b	Bit set when UR or CA is returned internally for a data read request. The queue is disabled on such events.
TxDropURCADesc	15	CR	0b	Bit set when UR or CA is returned internally for a descriptor read request. The queue is disabled on such events.
TxDropMaxTCB	16	CR	0b	Bit set when a Tx packet exceeds the TCB size of 16 KB. It can happen when a packet spans over too many data buffers which are not aligned to 16-byte address boundaries. The queue is disabled on such events.
TxTDBALHQEn	17	CR	0b	Bit set when modifying TDBAL or TDBAH while the queue is enabled. The write access is ignored.
TxTDTGtTDLEN	18	CR	0b	Bit set when TDT is set beyond TDLEN. TDT points to TDLEN.
TxTXDCTLQEn	19	CR	0b	Bit set when TXDCTL is modified while the queue is enabled. The write access is ignored.
Reserved	31:20	CR	0x0	Spare cause fields.

11.27.2.111 PCIE_LVMMI

Address: 0x100000 + 0x18881
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
FirstMaliciousEvent	13:0	CR	0x0	Bits 7:0 = Index of the queue. Bits 12:8 = Malicious cause code as per LVMMC register. Bit 13 = First malicious valid.
SecondMaliciousEvent	27:14	CR	0x0	Bits 21:14 = Index of the queue. Bits 26:22 = Malicious cause code as per LVMMC register. Bit 27 = Second malicious valid.
Reserved	31:28	RSV	0x0	Reserved.

11.27.2.112 PCIE_HOST_MISC

Address: 0x100000 + 0x19000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
Mgmt_TO	19:0	RW	0x800	Timeout value for MGMT accesses to PEP.
Mgmt_TO_En	20	RW	1b	Timeout enable.
Timeout	21	CW1	0b	Timeout status bit.
Reserved	31:22	RSV	0x0	Reserved.

11.27.2.113 PCIE_HOST_LANE_CTRL

Address:	**0x100000 + 0x19001**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MasterLane	2:0	RW	000b	Select the number of the master lane in the SerDes group. This lane acts as the master, and drives the 400 MHz clock to all the rest of the lanes. Also, it is the source of the refclk sync logic in the PCIE SerDes. Valid values are: **PEP0:** 000b = Default for PEP0 001b = x2 reversed 011b = x4 reversed 111b = x8 reversed **PEP1:** 100b = Default for PEP1 101b = x2 reversed 111b = x4 reversed The correct values are loaded by BSM at boot time.
RxStandby_En	3	RW	1b	Enable requirement for PIPE Rx-standby logic. When cleared, the Rx-standby is driven from Synopsys core.
Sym_Realign	4	RW	1b	Controls the i_rx_8b10b_realign input to SerDes.
FTS_Align	5	RW	1b	Controls the i_fts_align_en input to SerDes.
IntDisableAutoAssert	6	RW	0b	Enables auto-assertion of the i_pcs_interrupt_disable after PCIE reset to SerDes.
Reserved	31:7	RSV	0x0	Reserved.

11.27.2.114 PCIE_SERDES_CTRL[0..7]

Access to register management interface, per serdes lane. The access is first enabled, and then each access is as follows:

- Set *InterruptCode*=XX, *DataWrite*=XX, *Interrupt*=1b.
- Poll until *Interrupt* and *InProgress* are 0b, *DataRead* is the response.

The PCIE_SERDES_CTRL[pep][0..7] are accessible only for the serdes lanes currently attached to the PEP. If an even-numbered PEP is in x8 mode, it can access all 8 lanes and the corresponding odd-numbered PEP must not access any lane. For PEPs in x4 mode, even-numbered PEPs can access only lanes 0-3 and odd-numbered PEPs can access only lanes 4-7. For PEP # 8 (mgmt), only lane 0 exists and may be accessed. Access to inaccessible or nonexistent lanes must not be attempted, even though the PCIE_SERDES_CTRL register is defined.

Normally, access to PCIE_SERDES_CTRL registers should be done by the Switch Manager or Boot State Machine. They should not be used by the host driver, as the host driver should be the same for all PEPs, and thus not cognitive about the particular PEP the driver is attached to.

The control values used for *InterruptCode*, *DataRead*, and *DataWrite* are reserved to Intel.

Address:	**0x100000 + 0x2*i + 0x19010**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD**

Field Name	Bit(s)	Type	Default	Description
Reserved	0	RSV	0b	Reserved.
Interrupt	1	RW	0b	Set to interrupt the SerDes microprocessor to process the command sent. Self-clears when command done.
InProgess	2	RO	0b	Indicates if the interrupt is in progress or not. Self-clears when command done.
Reserved	15:3	RSV	0x0	Reserved.
InterruptCode	31:16	RW	0x0	Interrupt code to send to SerDes.
DataWrite	47:32	RW	0x0	Interrupt data to send to SerDes.
DataRead	63:48	RO	0x0	Interrupt data receives from SerDes.

11.28 PCIE_VF Registers Description

11.28.1 PCIE_VF Map

Table 11-40 PCI Express Registers - Virtual Function Map

Name	Address (Base = 0x200000)	Atomicity	Brief Description
PCIE_VFCTRL	Base + 0x0	32	PCI Express Registers - Virtual Function.
PCIE_VFPBACL[0..7]	Base + 0x1*i + 0x8	32	VF MSI-X PBA Clear.
PCIE_VFMBX	Base + 0x10	32	VF Mailbox.
PCIE_VFMBMEM[0..15]	Base + 0x1*i + 0x20	32	VF Mailbox Memory.
PCIE_VFINT_MAP[0..15]	Base + 0x1*i + 0x30	32	VF Interrupt Mapping.
PCIE_VFSYSTIME	Base + 0x40	32	System Time Register.
PCIE_VFITR[0..31]	Base + 0x1*i + 0x60	32	Interrupt Timers.
PCIE_VFMRQC	Base + 0x2100	32	Multiple Receive Queues Command Register.
PCIE_VFRSSRK[0..9]	Base + 0x1*i + 0x800	32	RSS Random Key Register.
PCIE_VFRETA[0..31]	Base + 0x1*i + 0x1000	32	Redirection Table.
PCIE_VFRDBAL[0..255]	Base + 0x40*i + 0x4000	32	Receive Descriptor Base Address Low.
PCIE_VFRDBAH[0..255]	Base + 0x40*i + 0x4001	32	Receive Descriptor Base Address High.
PCIE_VFRDLEN[0..255]	Base + 0x40*i + 0x4002	32	Receive Descriptor Length.
PCIE_VFTPH_RXCTRL[0..255]	Base + 0x40*i + 0x4003	32	Rx TPH Control Register.
PCIE_VFRDH[0..255]	Base + 0x40*i + 0x4004	32	Receive Descriptor Head.
PCIE_VFRDT[0..255]	Base + 0x40*i + 0x4005	32	Receive Descriptor Tail.
PCIE_VFRXQCTL[0..255]	Base + 0x40*i + 0x4006	32	Receive Queue Control.
PCIE_VFRXDCTL[0..255]	Base + 0x40*i + 0x4007	32	Receive Descriptor Control.
PCIE_VFRXINT[0..255]	Base + 0x40*i + 0x4008	32	Receive Interrupt.
PCIE_VFSRRCTL[0..255]	Base + 0x40*i + 0x4009	32	Split Receive Control Registers.
PCIE_VFQPRC[0..255]	Base + 0x40*i + 0x400A	32	Queue Packets Received Count.
PCIE_VFQPRDC[0..255]	Base + 0x40*i + 0x400B	32	Queue Packets Received Drop Count.
PCIE_VFQBRC_L[0..255]	Base + 0x40*i + 0x400C	32	Queue Bytes Received Count Low.
PCIE_VFQBRC_H[0..255]	Base + 0x40*i + 0x400D	32	Queue Bytes Received Count High.
PCIE_VFTDBAL[0..255]	Base + 0x40*i + 0x8000	32	Transmit Descriptor Base Address Low.
PCIE_VFTDBAH[0..255]	Base + 0x40*i + 0x8001	32	Transmit Descriptor Base Address High.
PCIE_VFTDLEN[0..255]	Base + 0x40*i + 0x8002	32	Transmit Descriptor Length.
PCIE_VFTPH_TXCTRL[0..255]	Base + 0x40*i + 0x8003	32	Tx TPH Control Registers.
PCIE_VFTDH[0..255]	Base + 0x40*i + 0x8004	32	Transmit Descriptor Head.
PCIE_VFTDT[0..255]	Base + 0x40*i + 0x8005	32	Transmit Descriptor Tail.
PCIE_VFTXDCTL[0..255]	Base + 0x40*i + 0x8006	32	Transmit Descriptor Control.
PCIE_VFTXQCTL[0..255]	Base + 0x40*i + 0x8007	32	Transmit Queue Control.

Table 11-40 PCI Express Registers - Virtual Function Map [Continued]

Name	Address (Base = 0x200000)	Atomicity	Brief Description
PCIE_VFTXINT[0..255]	Base + 0x40*i + 0x8008	32	Transmit Interrupt.
PCIE_VFQPTC[0..255]	Base + 0x40*i + 0x8009	32	Queue Packets Transmitted Count.
PCIE_VFQBTC_L[0..255]	Base + 0x40*i + 0x800A	32	Queue Bytes Transmitted Count Low.
PCIE_VFQBTC_H[0..255]	Base + 0x40*i + 0x800B	32	Queue Bytes Transmitted Count High.
PCIE_VFTQDLOC[0..255]	Base + 0x40*i + 0x800C	32	Tx Descriptors Location in Tx Cache.
PCIE_VFTXSGLORT[0..255]	Base + 0x40*i + 0x800D	32	Source GLORT for each Q.
PCIE_VFVTCTL[0..255]	Base + 0x40*i + 0x800E	32	VT Control Register.
PCIE_VFTX_DESC[0..255][0..255][0..3]	Base + 0x400*k + 0x4*j + 0x1*i + 0x40000	32	Transmit Descriptor Cache

11.28.2 PCIE_VF Registers

11.28.2.1 PCIE_VFCTRL

Address: 0x200000 + 0x0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Reserved	2:0	RSV	000b	Reserved.
RST	3	RW	0b	VF Reset. This bit performs a reset of the interrupt registers of the VF. It has a similar effect as a VFLR (reseting the VF internal states) except that it does not reset the VF configuration space. Only portions of the ITR registers are reset. The *PendingX* and *TimerXExpired* values are reset in the ITR register, while the *IntervalX*, *Automask* and *Mask* are not affected. The driver sets the bit for at least 100 μs before clearing it. To complete the VF reset, all the queues owned by the VF are disabled by the driver.
Reserved	31:4	RSV	0x0	Reserved.

11.28.2.2 PCIE_VFPBACL[0..7]

Address: 0x200000 + 0x1*i + 0x8
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
PENBIT	31:0	CW1	0x0	MSI-X Pending Bits Clear. Writing to any bit: 0b = No effect. 1b = Clears the corresponding MSIXPBA bit. Reading this register returns the PBA vector.

11.28.2.3 PCIE_VFMBX

Address:	**0x200000 + 0x10**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Owner	0	CW1	1b	Hardware semaphore. The requester reads this field to claim ownership over the mailbox. 0b = Ownership has already been granted to another software entity, and it will try again later. 1b = Requestor has been granted ownership. The owner writes to this field to release ownership. 0b = No effect. 1b = Release ownership. ** Do not use this bit. Hardware semaphore mechanism is not operational. Use a lockless mailbox mechanism instead, where the mailbox memory is split in two halves, one half for the PF and one half for the VF **
PFReq	1	WO	0b	Message from VF for PF is ready. Setting this bit causes an interrupt to be latched in PCIE_MBX[vf].*PFReqInterrupt*. This bit always reads as 0b.
PFAck	2	WO	0b	Message from PF to VF was received. Setting this bit causes an interrupt to be latched in PCIE_MBX[vf].*PFAckInterrupt*. This bit always reads as 0b.
ReqInterrupt	3	CW1	0b	Set when message from PF to VF is ready.
AckInterrupt	4	CW1	0b	Set when message from VF to PF has been read by PF.
InterruptEnable	6:5	RW	00b	Controls if mailbox interrupts to VF are enabled or not. Read as 0 if disabled. Read as 1 if enabled. 00b = Do not change the mode. 01b = Enable mailbox interrupts to VF. 10b = Disable mailbox interrupts to VF. 11b = Do not change the mode.
Reserved	31:7	RSV	0x0	Reserved.

11.28.2.4 PCIE_VFMBMEM[0..15]

Mailbox memory for PF and VF drivers communication. The mailbox size for each VM is 64 bytes accessed by 32-bit registers. Locations must be accessed as 32-bit words.

Address:	**0x200000 + 0x1*i + 0x20**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
MailboxData	31:0	RW	0x0	Mailbox Data field composed of 16 x 4-byte registers.

11.28.2.5 PCIE_VFINT_MAP[0..15]

Defines the interrupt number for each source (VF). The index to this table indicates the type of interrupt and the output is the index to the MSI-X vector table offset relative to the section reserved for this VF.

The following vectors are used:

- PCIE_VFINT_MAP[0]: Mailbox

The PCIE_VFINT_MAP[1..15] are not used and reserved for future usage. They are all mapped to entry [0] for read and write.

Address: 0x200000 + 0x1*i + 0x30
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT

Field Name	Bit(s)	Type	Default	Description
Interrupt	4:0	RW	0x0	Interrupt number.
Reserved	7:5	RSV	000b	Reserved.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disable this interrupt. Whenever an INT_MAP register is transitions in/out of Timer 3 (disabled) state, the driver either disables the interrupt before, or cleans up any associated cause thereafter.
Reserved	31:10	RSV	0x0	Reserved.

11.28.2.6 PCIE_VFSYSTIME

Copy of PCIE_SYSTIME.

Address: 0x200000 + 0x40
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD

Field Name	Bit(s)	Type	Default	Description
CurrentTime	63:0	RW	0x0	System time LSB register.

11.28.2.7 PCIE_VFITR[0..31]

The VF view of the PCIE_ITR[0..767] table (refer to Section 11.27.2.84). The VF view is limited to the portion belonging to this VF, and the index 0..N is relative to that view.

Address: 0x200000 + 0x1*i + 0x60
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM

Field Name	Bit(s)	Type	Default	Description
Interval0	11:0	RW	0x0	Defines minimum interval between interrupts, timer 0.
Interval1	23:12	RW	0x0	Defines minimum interval between interrupts, timer 1.
Timer0Expired	24	RO	0b	Indicates timer 0 has expired.

Address:	**0x200000 + 0x1*i + 0x60**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
Timer1Expired	25	RO	0b	Indicates timer 1 has expired.
Pending0	26	SW1	0b	Indicates a pending event is waiting for time 0 expiration. 0b = No effect. 1b = Sets the interrupt.
Pending1	27	SW1	0b	Indicates a pending event is waiting for time 1 expiration. 0b = No effect. 1b = Sets the interrupt.
Pending2	28	SW1	0b	Indicates a pending event is waiting for time 2 expiration. 0b = No effect. 1b = Sets the interrupt.
AutoMask	29	RW	0b	Defines if the interrupt is auto-masked by hardware. If set to 1b, the hardware automatically blocks interrupts (Mask set to 1b) when the interrupt is posted, and thus blocks future interrupts until the software writes the clear mask command in the *Mask* field.
Mask	31:30	RW	01b	Control masking of this interrupt: On write: 00b = Do not change mask status. 01b = Set mask (block interrupts). 10b = Clear mask (unblock interrupts). 11b = Reserved. On read: 00b = Interrupt not blocked. 01b = Interrupt blocked. 10b = Reserved. 11b = Reserved.

11.28.2.8 PCIE_VFMRQC

VF view of PCIE_MRQC (refer to Section 11.27.2.30 for details).

Address:	**0x200000 + 0x2100**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
TcpIPv4	0	RW	0b	Enable TcpIPv4 hash function.
IPv4	1	RW	0b	Enable IPv4 hash function.
Reserved	3:2	RSV	00b	Reserved.
IPv6	4	RW	0b	Enable IPv6 hash function.
TcpIPv6	5	RW	0b	Enable TcpIPv6 hash function.
UdpIPV4	6	RW	0b	Enable UdpIPv4 hash function.
UdpIPV6	7	RW	0b	Enable UdpIPv6 hash function.
Reserved	31:8	RSV	0x0	Reserved.

11.28.2.9 PCIE_VFRSSRK[0..9]

Address: 0x200000 + 0x1*i + 0x800
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
K0	7:0	RW	0x0	RSS Key Byte [4*n+0] of the RSS random key, for each register [n].
K1	15:8	RW	0x0	RSS Key Byte [4*n+1] of the RSS random key, for each register [n].
K2	23:16	RW	0x0	RSS Key Byte [4*n+2] of the RSS random key, for each register [n].
K3	31:24	RW	0x0	RSS Key Byte [4*n+3] of the RSS random key, for each register [n].

11.28.2.10 PCIE_VFRETA[0..31]

The contents of the redirection table are not defined following reset of the Memory Configuration registers. System software must initialize the table prior to enabling multiple receive queues. It can also update the redirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets. Therefore, it is not guaranteed that a table update takes effect on a specific packet boundary.

Address: 0x200000 + 0x1*i + 0x1000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Entry0	7:0	RW	0x0	Entry0 defines the RSS output index for hash value [4*n+0]. While [n] is the register index, equals to 0...31.
Entry1	15:8	RW	0x0	Entry1 defines the RSS output index for hash value [4*n+1]. While [n] is the register index, equals to 0...31.
Entry2	23:16	RW	0x0	Entry2 defines the RSS output index for hash value [4*n+2]. While [n] is the register index, equals to 0...31.
Entry3	31:24	RW	0x0	Entry3 defines the RSS output index for hash value [4*n+3]. While [n] is the register index, equals to 0...31.

11.28.2.11 PCIE_VFRDBAL[0..255]

VF view of PCIE_RDBAL (refer to Section 11.27.2.49 for details).

Address: 0x200000 + 0x40*i + 0x4000
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
RDBAL	31:7	RW	0x0	Receive Descriptor Base Address Low.

11.28.2.12 PCIE_VFRDBAH[0..255]

VF view of PCIE_RDBAH (refer to Section 11.27.2.50 for details).

Address: 0x200000 + 0x40*i + 0x4001
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
RDBAH	31:0	RW	0x0	Receive Descriptor Base Address [63:32].

11.28.2.13 PCIE_VFRDLEN[0..255]

VF view of PCIE_RDLEN (refer to Section 11.27.2.51 for details).

Address: 0x200000 + 0x40*i + 0x4002
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
LEN	19:7	RW	0x20	
Reserved	31:20	RSV	0x0	Reserved.

11.28.2.14 PCIE_VFTPH_RXCTRL[0..255]

VF view of PCIE_TPH_RXCTRL (refer to Section 11.27.2.52 for details). This register is READ ONLY for VF.

Address: 0x200000 + 0x40*i + 0x4003
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
Reserved1	4:0	RO	0x0	Reserved.
RxDescriptorTPHEN	5	RO	0b	Descriptor TPH EN. 0b = Hardware does not enable TPH for descriptor write-backs. 1b = Hardware enables TPH for all Rx descriptors written back into memory.
RxHeaderTPHEN	6	RO	0b	Rx Header TPH EN. 0b = Hardware does not enable TPH for Rx Headers. 1b = Hardware enables TPH for all received header buffers.
RxPayloadTPHEN	7	RO	0b	Payload TPH EN. 0b = Hardware does not enable TPH for Ethernet payloads. 1b = Hardware enables TPH for all Ethernet payloads written into memory. By default this bit is cleared (0b).
Reserved2	8	RO	0b	Reserved.
RXdescReadROEn	9	RO	1b	Rx Descriptor Read Relax Order Enable.
Reserved3	10	RO	0b	Reserved.
RXdescWBROen	11	RO	0b	Rx Descriptor Write Back Relax Order Enable. This bit must be 0b to enable correct functionality of the descriptors write-back.
Reserved4	12	RO	0b	Reserved.
RXdataWriteROEn	13	RO	1b	Rx data Write Relax Order Enable.

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x4003** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Reserved5	14	RO	0b	Reserved.
RxRepHeaderROEn	15	RO	1b	Rx Split Header Relax Order Enable.
Reserved6	23:16	RO	0x0	Reserved.
CPUID	31:24	RO	0x0	Physical ID for TPH operated in DeviceSpecific mode. • Legacy TPH capable platforms — The device driver, upon discovery of the physical CPU ID and CPU bus ID, programs the *CPUID* field with the physical CPU and bus ID associated with this Rx queue. • TPH 1.0 capable platforms — The device driver programs a value, based on the relevant APIC ID, associated with this Rx queue. ***Note:*** This field is not operational in the FM10000.

11.28.2.15 PCIE_VFRDH[0..255]

VF view of PCIE_RDH (refer to Section 11.27.2.53 for details).

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x4004** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RDH	15:0	RO	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.16 PCIE_VFRDT[0..255]

VF view of PCIE_RDT (refer to Section 11.27.2.54 for details).

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x4005** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RDT	15:0	RW	0x0	Data.
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.17 PCIE_VFRXQCTL[0..255]

Address: 0x200000 + 0x40*i + 0x4006
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
ENABLE	0	RW	0b	Receive Queue Enable. When set, the *ENABLE* bit enables the operation of the specific receive queue.
Reserved	1	RSV	0b	Reserved.
VF	7:2	RO	0x0	Indicates to which VF this queue belongs. This field is RW when accessed from PF, and RO when accessed from VF.
OwnedByVF	8	RO	0b	Indicates this queue belongs to a Virtual Function. Invalidates the *VF* field if cleared. This field is RW when accessed from PF and RO when accessed from VF.
Reserved	31:9	RSV	0x0	Reserved.

11.28.2.18 PCIE_VFRXDCTL[0..255]

VF view of PCIE_RXDCTL (refer to Section 11.27.2.56 for details). This register is READ ONLY.

Address: 0x200000 + 0x40*i + 0x400
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
MaxTime	7:0	RO	0x0	Max time before writing back descriptors. Time is in 512 * PCLK periods. A value of 0 means no delay. Only used on end of packet descriptors. For non-end of packet descriptors, the descriptors are held until a cache line is full and then written.
WriteBackImm	8	RO	0b	Defines if descriptor is written back immediately regardless how full the 64-byte cache line is.
DropOnEmpty	9	RO	0b	Defines if frames are dropped or held when a queue is empty. 0b = HOLD — Hold onto the frame if queue has no descriptors until descriptors become available. 1b = DROP — Drop the frame if queue has no descriptors.
WriteRSSHash	10	RO	0b	Defines, for 16-byte descriptor, if the RSS-HASH field or DGLORT/SGLORT is written to the descriptor. For 32-byte descriptors, both fields get written. 0b = DGLORT/SGLORT — Write DGLORT/SGLORT to descriptor. 1b = RSS_HASH — Write RSS Hash (32 bits) to descriptor. ** OBSOLETE, NOT IMPLEMENTED **
Reserved	31:11	RSV	0x0	Reserved.

11.28.2.19 PCIE_VFRXINT[0..255]

VF view of PCIE_RXINT (refer to Section 11.27.2.57 for details).

Address:	**0x200000 + 0x40*i + 0x4008**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Interrupt	7:0	RW	0x0	Defines the MSI-X interrupt vector number used by this queue. This number is relative to the base of the interrupt blocks assigned to a VF or PF.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disable this interrupt.
Reserved	31:10	RSV	0x0	Reserved.

11.28.2.20 PCIE_VFSRRCTL[0..255]

VF view of PCIE_VFSRRCTL (refer to Section 11.27.2.58 for details).

Address:	**0x200000 + 0x40*i + 0x4009**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
BSIZEPACKET	7:0	RW	0x8	Receive Buffer Size for Packet Buffer. The value is in 256 bytes resolution. Value can be from 256 bytes to 16 KB. Default buffer size is 2 KB. This field should not be set to 0x0.
BSIZEHEADER	13:8	RW	0x4	Receive Buffer Size for Header Buffer. The value is in 64 bytes resolution. Value can be from 64 bytes to 1024 bytes. ***Note:*** The maximum supported header size is limited to 1023. Default buffer size is 256 bytes. Values above 1024 bytes are reserved for internal use only.
DESCTYPE	15:14	RW	00b	Define the descriptor type in Rx: 00b = No header split 01b = Header split — Header uses header buffer, unless it cannot fit. Packet split boundary defined in *PSRTYPE*. 10b = Small/large split — Small packets use small packet buffer, large packets use normal packet data buffer. 11b = Reserved. ***Note:*** If *DESCTYPE* is set to 01b or 10b, *BSIZEHEADER* must be bigger than zero, and *BufferChainingEn* must be set to zero. Otherwise, no header split occurs.

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x4009** **32** **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**

PSRTYPE[0..13]	29:16	RW	0b	(14 x 1-bit)
				Packets are split according to the lowest-indexed entry that applies to the packet and that is enabled.
				For example, if bits 6 and 8 are set, then an IPv4 packet that is not TCP is split after the IPv4 header.
				This bit mask table enables or disables each type of header to be split. A value of 1b enables an entry.
				PSTYPE[0] = Split after TCP of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[1] = Split after UDP of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[2] = Split after IPv4 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[3] = Split after IPv6 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[4] = Split after L2 of inner header (VXLAN, NVGRE, or NGE).
				PSTYPE[5] = Split after NVGRE/VXLAN header, but not inner packet.
				PSTYPE[6] = Split after TCP of outer header.
				PSTYPE[7] = Split after UDP of outer header.
				PSTYPE[8] = Split after IPv4 of outer header.
				PSTYPE[9] = Split after IPv6 of outer header.
				PSTYPE[10] = Split after L2 of outer header.
				All other values are reserved.

LoopbackSuppress	30	RW	0b	Defines if Loopback Suppress (compare SGLORT) is enabled or not.
BufferChainingEn	31	RW	0b	Buffer Chaining enable. 0b = Any packet longer than the data buffer size is terminated with a TOO_BIG error status in Rx descriptor write-back. The remainder of the frame is not posted to host, it is silently dropped. 1b = A packet can be spread over more than one single receive data buffer.

11.28.2.21 PCIE_VFQPRC[0..255]

VF view of PCIE_QPRC (refer to Section 11.27.2.59 for details).

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x400** **32** **DEVICE, MASTER, COLD, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Data	31:0	RO	0x0	Data.

11.28.2.22 PCIE_VFQPRDC[0..255]

VF view of PCIE_QPRDC (refer to Section 11.27.2.60 for details).

Address: **Atomicity:** **Reset Domains:**	**0x200000 + 0x40*i + 0x400B** **32** **DEVICE, MASTER, COLD, PCIE_HOT_MEM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PRDC	31:0	RO	0x0	

11.28.2.23 PCIE_VFQBRC_L[0..255]

VF view of PCIE_QBRC_L (refer to Section 11.27.2.61 for details).

Address: **0x200000 + 0x40*i + 0x400C**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
BRC_L	31:0	RO	0x0	

11.28.2.24 PCIE_VFQBRC_H[0..255]

VF view of PCIE_QBRC_H (refer to Section 11.27.2.62 for details).

Address: **0x200000 + 0x40*i + 0x400D**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
BRC_H	15:0	RO	0x0	Data.
Reserved	31:15	RSV	0x0	Reserved.

11.28.2.25 PCIE_VFTDBAL[0..255]

VF view of PCIE_TDBAL (refer to Section 11.27.2.64 for details).

Address: **0x200000 + 0x40*i + 0x8000**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
TDBAL	31:7	RW	0x0	

11.28.2.26 PCIE_VFTDBAH[0..255]

VF view of PCIE_TDBAH (refer to Section 11.27.2.65 for details).

Address: **0x200000 + 0x40*i + 0x8001**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
TDBAH	31:0	RW	0x0	

11.28.2.27 PCIE_VFTDLEN[0..255]

VF view of PCIE_TDLEN (refer to Section 11.27.2.66 for details).

Address:	**0x200000 + 0x40*i + 0x8002**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	6:0	RSV	0x0	Reserved.
LEN	19:7	RW	0x4	
Reserved	31:20	RSV	0x0	Reserved.

11.28.2.28 PCIE_VFTPH_TXCTRL[0..255]

VF view of PCIE_TPH_TXCTRL (refer to Section 11.27.2.67 for details). This register in is READ ONLY when accessed from VF.

Address:	**0x200000 + 0x40*i + 0x8003**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Reserved	4:0	RSV	0x0	Reserved.
TxDescriptorTPHEN	5	RO	0b	Descriptor TPH Enable. 0b = Hardware does not enable TPH for descriptor write-backs. 1b = Hardware enables TPH for all Tx descriptors written back into memory. This bit is cleared by default.
Reserved	8:6	RSV	000b	Reserved.
TXdescRDROEn	9	RO	1b	Tx Descriptor Read Relax Order Enable.
Reserved	10	RSV	0b	Reserved.
TXdescWBROen	11	RO	1b	Relax Order Enable of Tx Descriptor as well as head pointer write back (when set).
Reserved	12	RSV	0b	Reserved.
TXDataReadROEn	13	RO	1b	Tx Data Read Relax Order Enable.
Reserved	23:14	RSV	0x0	Reserved.
CPUID	31:24	RO	0x0	Physical ID for TPH operated in DeviceSpecific mode. • Legacy TPH capable platforms — The device driver, upon discovery of the physical CPU ID and CPU bus ID, programs the *CPUID* field with the physical CPU and bus ID associated with this Tx queue. • TPH 1.0 capable platforms — The device driver programs a value, based on the relevant APIC ID, associated with this Tx queue. ***Note:*** This field is not operational in the FM10000.

11.28.2.29 PCIE_VFTDH[0..255]

VF view of PCIE_TDH (refer to Section 11.27.2.68 for details).

Address: **0x200000 + 0x40*i + 0x8004**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
TDH	15:0	RW	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.30 PCIE_VFTDT[0..255]

VF view of PCIE_TDT (refer to Section 11.27.2.69 for details).

Address: **0x200000 + 0x40*i + 0x8005**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
TDT	15:0	RW	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.31 PCIE_VFTXDCTL[0..255]

VF view of PCIE_TXDCTL (refer to Section 11.27.2.70 for details).

Address: **0x200000 + 0x40*i + 0x8006**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
PTHRESH	6:0	RO	0x0	** OBSOLETE, NOT IMPLEMENTED ** Pre-Fetch Threshold. Controls when a prefetch of descriptors is considered. This threshold refers to the number of valid, unprocessed transmit descriptors the has in its on-chip buffer. If this number drops below *PTHRESH*, the algorithm considers pre-fetching descriptors from host memory. However, this fetch does not happen unless there are at least *HTHRESH* valid descriptors in host memory to fetch. ***Note:*** *HTHRESH* should be given a non-zero value each time *PTHRESH* is used.
HTHRESH	13:7	RO	0x0	** OBSOLETE, NOT IMPLEMENTED ** Host Threshold.
ENABLE	14	RW	0b	Transmit Queue Enable. When set, this bit enables the operation of a specific transmit queue: Default value for all queues is 0b.
Reserved	15	RSV	0b	Reserved.
MaxTime	27:16	RW	0x0	Max idle time before posting last descriptor write back and an interrupt. Time is in 512 * PCLK periods. The RS overwrites the write back. A value of 0b means no delay.

Address: 0x200000 + 0x40*i + 0x8006
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM

Field Name	Bit(s)	Type	Default	Description
PushDesc	28	RW	0b	Defines if this queue has the descriptors written into the controller (push model) or if the controller read descriptors from host memory (pull model). 0b = PULL — Descriptors read by controller from host memory.\ 1b = PUSH — Descriptors written by host into controller. Setting this bit to 1b is meaningless if PCIE_VFTXQCTL.*PushModeDis* has been asserted by PF.
Reserved	31:29	RSV	000b	Reserved.

11.28.2.32 PCIE_VFTXQCTL[0..255]

This register is RW when accessed from PF and RO when accessed from VF.

Address: 0x200000 + 0x40*i + 0x8007
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH

Field Name	Bit(s)	Type	Default	Description
VF	5:0	RO	0x0	Indicates to which VF this queue belongs. This field is RW when accessed from PF and RO when accessed from VF.
OwnedByVF	6	RO	0b	Indicates this queue belongs to a Virtual Function. Invalidates the *VF* field if cleared. This field is RW when accessed from PF and RO when accessed from VF.
PC	9:7	RO	000b	Defines the Priority Flow Control Class (Pause Class). This field is RW when accessed from PF and RO when accessed from VF.
TC	15:10	RO	0x0	Defines the shaping group to which this queue belongs (Traffic Class). Meaningful only when *UnlimitedBW* bit is cleared.
VID	27:16	RO	0x0	Defines the default VLAN ID for the queue if none supplied. This value could be 0b. If this default VLAN ID is set to 0b and the host set to 0b, the switch assigns a default VLAN ID. This field is RW when accessed from PF and RO when accessed from VF.
UnlimitedBW	28	RO	0b	This bit provides the ability to create a 65th TC with unlimited bandwidth. Usage model envisaged is for the PF's queue when there are 64 enabled VFs, each one with its own TC. Setting this bit makes the *TC* field meaningless.
PushModeDis	29	RO	0b	This bit provides the ability for the PF to disallow the VF to use the queue in push mode. This is a chicken bit for the case a security issue is found in push mode.
Reserved	31:30	RSV	00b	Reserved.

11.28.2.33 PCIE_VFTXINT[0..255]

VF view of PCIE_TXINT (refer to Section 11.27.2.72 for details).

Address:	**0x200000 + 0x40*i + 0x8008**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
Interrupt	7:0	RW	0x0	Defines the MSI-X interrupt vector number used by this queue. This number is relative to the base of the interrupt blocks assigned to a VF or PF.
InterruptTimer	9:8	RW	00b	Defines which timer to use for interrupt vector. Valid values are 0,1,2 or 3: 00b = Use ITR timer 0. 01b = Use ITR timer 1. 10b = Use ITR timer 2 (immediate, no timer). 11b = Disable this interrupt.
Reserved	31:10	RSV	0x0	Reserved.

11.28.2.34 PCIE_VFQPTC[0..255]

VF view of PCIE_QPTC (refer to Section 11.27.2.73 for details).

Address:	**0x200000 + 0x40*i + 0x8009**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
PTC	31:0	RO	0x0	

11.28.2.35 PCIE_VFQBTC_L[0..255]

VF view of PCIE_QBTC_L (refer to Section 11.27.2.74 for details).

Address:	**0x200000 + 0x40*i + 0x800A**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
BTCL	31:0	RO	0x0	

11.28.2.36 PCIE_VFQBTC_H[0..255]

VF view of PCIE_QBTC_H (refer to Section 11.27.2.75 for details).

Address:	**0x200000 + 0x40*i + 0x800B**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT_MEM**

Field Name	Bit(s)	Type	Default	Description
BTCH	15:0	RO	0x0	
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.37 PCIE_VFTQDLOC[0..255]

VF view of PCIE_TQLOC (refer to Section 11.27.2.76 for details). This register is READ ONLY, VF cannot change the location of descriptors in the cache.

Address:	**0x200000 + 0x40*i + 0x800C**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Base	15:0	RO	0x0	Defines the base location of the descriptor area in 64-byte chunks. Only 3 allocation schemes are supported where Base for Q index i is set to: ix8 When DMA_CTRL.*MaxNumOfQs* is set to 00b (256 Qs). ix16 When DMA_CTRL.*MaxNumOfQs* is set to 01b (128 Qs). ix32 When DMA_CTRL.*MaxNumOfQs* is set to 10b (64 Qs).
Size	19:16	RO	0x0	Defines the size of the descriptor area in log2(N). Minimum is 2 (4 descriptors) and maximum is 8 (256 descriptors). Only 3 Sizes supported: 5 (32 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 00b (256 Qs). 6 (64 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 01b (128 Qs). 7 (128 descriptors) When DMA_CTRL.*MaxNumOfQs* is set to 10b (64 Qs).
Reserved	31:20	RSV	0x0	Reserved.

11.28.2.38 PCIE_VFTXSGLORT[0..255]

Address:	**0x200000 + 0x40*i + 0x800D**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
SGlort	15:0	RO	0x0	Source GLORT to use when sending a packet to switch for this Q. This field is read only to VF.
Reserved	31:16	RSV	0x0	Reserved.

11.28.2.39 PCIE_VFVTCTL[0..255]

VF view of PCIE_PFVTCTL (refer to Section 11.27.2.78 for details). This register is READ ONLY, VF cannot change the FTAG insertion permission.

Address:	**0x200000 + 0x40*i + 0x800E**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_HOT, PCIE_DATAPATH**

Field Name	Bit(s)	Type	Default	Description
FTagDescEnable	0	RO	0b	
Reserved	31:1	RSV	0x0	Reserved.

11.28.2.40 PCIE_VFTX_DESC[0..255][0..255][0..3]

Used to store data and control. Indexing is [queue#,desc#,word#].

Address:	**0x200000 + 0x400*k + 0x4*j + 0x1*i + 0x40000**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_WARM, PCIE_DATAPATH_MEM**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	WO	0x0	

11.29 PCIE_CFG Registers Description

The PCIE config registers are accessible via PCI Configuration Cycles using the lower 8 bits of the their address (0x000-0x3FF in byte address, 0x000-0x0FF in word address).

The same configuration registers are also visible in the global address range and are all R/W in this case. Default values are loaded via serial EEPROM at boot time using the global map view.

11.29.1 PCIE_CFG Map

Table 11-41 PCI Express Configuration Registers - Physical Function Map

Name	Address (Base = 0x120000)	Atomicity	Brief Description
PCIE_CFG_ID	Base + 0x0	32	
PCIE_CFG_CMD	Base + 0x1	32	
PCIE_CFG_1	Base + 0x2	32	
PCIE_CFG_2	Base + 0x3	32	
PCIE_CFG_BAR0	Base + 0x4	32	
PCIE_CFG_BAR1	Base + 0x5	32	
PCIE_CFG_BAR2	Base + 0x6	32	
PCIE_CFG_BAR3	Base + 0x7	32	
PCIE_CFG_BAR4	Base + 0x8	32	
PCIE_CFG_BAR5	Base + 0x9	32	
PCIE_CFG_CARDBUS	Base + 0xA	32	
PCIE_CFG_SUBID	Base + 0xB	32	
PCIE_CFG_EXP_ROM	Base + 0xC	32	
PCIE_CFG_CAP_PTR	Base + 0xD	32	
PCIE_CFG_RSVD	Base + 0xE	32	
PCIE_CFG_INT	Base + 0xF	32	
PCIE_CFG_PM_CAP	Base + 0x10	32	Power Management Capability.
PCIE_CFG_PM_CTRL	Base + 0x11	32	Power Management Control and Status.
PCIE_CFG_PCIE_CAP	Base + 0x1C	32	PCI Express Capability.
PCIE_CFG_PCIE_DEV_CAP	Base + 0x1D	32	
PCIE_CFG_PCIE_DEV_CTRL	Base + 0x1E	32	
PCIE_CFG_PCIE_LINK_CAP	Base + 0x1F	32	
PCIE_CFG_PCIE_LINK_CTRL	Base + 0x20	32	
PCIE_CFG_PCIE_DEV_CAP2	Base + 0x25	32	
PCIE_CFG_PCIE_DEV_CTRL2	Base + 0x26	32	
PCIE_CFG_PCIE_LINK_CTRL2	Base + 0x28	32	
PCIE_CFG_MSIX_CAP	Base + 0x2C	32	MSI-X Capability.
PCIE_CFG_MSIX_TABLE_OFFSET	Base + 0x2D	32	MSI-X Table Offset Register.

Table 11-41 PCI Express Configuration Registers - Physical Function Map [Continued]

Name	Address (Base = 0x120000)	Atomicity	Brief Description
PCIE_CFG_MSIX_PBA	Base + 0x2E	32	MSI-X Pending Bit Array — PBA Offset.
PCIE_CFG_VPD_CAP	Base + 0x34	32	
PCIE_CFG_VPD_DATA	Base + 0x35	32	
PCIE_CFG_AER_HDR	Base + 0x40	32	
PCIE_CFG_AER_UNERR_STATUS	Base + 0x41	32	
PCIE_CFG_AER_UNERR_MASK	Base + 0x42	32	
PCIE_CFG_AER_UNERR_SEVERITY	Base + 0x43	32	
PCIE_CFG_AER_COERR_STATUS	Base + 0x44	32	
PCIE_CFG_AER_COERR_MASK	Base + 0x45	32	
PCIE_CFG_AER_CTRL	Base + 0x46	32	
PCIE_CFG_AER_HEADER_LOG0	Base + 0x47	32	
PCIE_CFG_AER_HEADER_LOG1	Base + 0x48	32	
PCIE_CFG_AER_HEADER_LOG2	Base + 0x49	32	
PCIE_CFG_AER_HEADER_LOG3	Base + 0x4A	32	
PCIE_CFG_SPD_HDR	Base + 0x52	32	
PCIE_CFG_SPD_NUMBER_L	Base + 0x53	32	
PCIE_CFG_SPD_NUMBER_H	Base + 0x54	32	
PCIE_CFG_ARI_HDR	Base + 0x56	32	
PCIE_CFG_ARI_CTRL	Base + 0x57	32	
PCIE_CFG_SPCIE_HDR	Base + 0x5A	32	
PCIE_CFG_SPCIE_LINK_CTRL3	Base + 0x5B	32	
PCIE_CFG_SPCIE_ERR_STS	Base + 0x5C	32	
PCIE_CFG_SPCIE_LINK_EQ01	Base + 0x5D	32	
PCIE_CFG_SPCIE_LINK_EQ23	Base + 0x5E	32	
PCIE_CFG_SPCIE_LINK_EQ45	Base + 0x5F	32	
PCIE_CFG_SPCIE_LINK_EQ67	Base + 0x60	32	
PCIE_CFG_SRIOV_HDR	Base + 0x62	32	
PCIE_CFG_SRIOV_CAP	Base + 0x63	32	
PCIE_CFG_SRIOV_CTRL	Base + 0x64	32	
PCIE_CFG_SRIOV_CFG	Base + 0x65	32	
PCIE_CFG_SRIOV_NUM	Base + 0x66	32	
PCIE_CFG_SRIOV_MAP	Base + 0x67	32	
PCIE_CFG_SRIOV_DEVID	Base + 0x69	32	
PCIE_CFG_SRIOV_PAGE_SUP	Base + 0x69	32	
PCIE_CFG_SRIOV_PAGE_CFG	Base + 0x6A	32	
PCIE_CFG_SRIOV_BAR0	Base + 0x6B	32	
PCIE_CFG_SRIOV_BAR1	Base + 0x6C	32	

Table 11-41 PCI Express Configuration Registers - Physical Function Map [Continued]

Name	Address (Base = 0x120000)	Atomicity	Brief Description
PCIE_CFG_SRIOV_BAR2	Base + 0x6D	32	
PCIE_CFG_SRIOV_BAR3	Base + 0x6E	32	
PCIE_CFG_SRIOV_BAR4	Base + 0x6F	32	
PCIE_CFG_SRIOV_BAR5	Base + 0x70	32	
PCIE_CFG_SRIOV_MIG	Base + 0x71	32	
PCIE_CFG_TPH_HDR	Base + 0x72	32	
PCIE_CFG_TPH_CAP	Base + 0x73	32	
PCIE_CFG_TPH_CTRL	Base + 0x74	32	
PCIE_CFG_ACS_HDR	Base + 0x76	32	
PCIE_CFG_ACS_CAP	Base + 0x77	32	
PCIE_PORTLOGIC	Base + 0x1C0	32	
PCIE_PORTLOGIC_LINK_STATE	Base + 0x1CA	32	

11.29.2 PCIE_CFG Registers

11.29.2.1 PCIE_CFG_ID

Address:	**0x120000 + 0x0**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
VendorID	15:0	RO	0x8086	
DeviceID	31:16	RO	0x15A4	

11.29.2.2 PCIE_CFG_CMD

Address:	**0x120000 + 0x1**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
IOSpaceEn	0	RO	0b	I/O Access Enable Not used in this design. Always set to 0b.
MemSpaceEn	1	RW	0b	Memory Access Enable.
BusMasterEn	2	RW	0b	Enable Mastering (Also named Bus Master Enable (BME).)
SpecialCycEn	3	RO	0b	Special Cycle Monitoring (Hardwired to 0b.)
MemWrInv	4	RO	0b	MWI Enable (Hardwired to 0b.)
VGAPaletteSnp	5	RO	0b	Palette Snoop Enable (Hardwired to 0b.)
ParErrResp	6	RW	0b	Parity Error Response.
IdselStep	7	RO	0b	Wait Cycle Enable (Hardwired to 0b.)
SERRnEn	8	RW	0b	SERR# Enable.

Address: **0x120000 + 0x1**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
FastB2BEn	9	RO	0b	Fast Back-to-Back Enable (Hardwired to 0b.)
INTAssertDis	10	RW	0b	Interrupt Disable When set, devices are prevented from generating legacy interrupt messages.
Reserved	18:11	RSV	0x0	Reserved.
INTStat	19	RO	0b	Interrupt Status.
CapList	20	RO	1b	New Capabilities. Indicates that this device implements extended capabilities. It supports: • PCI Power Management • Message Signaled Interrupts (MSI) • Enhanced Message Signaled Interrupts (MSI-X) • VPD • The PCIe extensions.
Cap66Mhz	21	RO	0b	66 MHz Capable (Hard wire to 0b.)
Reserved	22	RSV	0b	Reserved.
CapFastB2B	23	RO	0b	Fast Back-to-Back Capable (Hardwired to 0b.)
MstrDataParErr	24	CW1	0b	Data Parity Reported.
DevselTiming	26:25	RO	00b	DEVSEL Timing (Hardwired to 0b.)
SigTgtAbort	27	CW1	0b	Signaled Target Abort.
RcvTgtAbort	27	CW1	0b	Received Target Abort.
RcvMstAbort	29	CW1	0b	Received Master Abort.
SigSysErr	30	CW1	0b	Signaled System Error.
DetectParErr	31	CW1	0b	Detected Parity Error.

11.29.2.3 PCIE_CFG_1

Address: **0x120000 + 0x2**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
RevisionID	7:0	RO	0x0	Set to 0b for first version.
ClassCode	31:8	RO	0x020000	The class code is a read-only value that identifies the device functionality, and is set to 0x020000 (Ethernet Adapter) by default.

11.29.2.4 PCIE_CFG_2

Address: **0x120000 + 0x3**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CacheLineSize	7:0	RW	0x0	Implemented by PCIe devices as a read/write field for legacy compatibility purposes, but has no impact on any PCIe device functionality.
LatencyTimer	15:8	RO	0x0	Not used.
HeaderType	22:16	RO	0x0	Tied to 0b for Endpoint.

Address: 0x120000 + 0x3
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
MultiFunction	23	RW	0b	Must be set to 1b by NVM/BSM whenever SR-IOV is enabled (which is the default).
BIST	31:24	RO	0x0	

11.29.2.5 PCIE_CFG_BAR0

Address: 0x120000 + 0x4
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	10b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.
Prefetchable	3	RO	1b	The PEP address space does not contain volatile registers.
Zero	21:4	RO	0x0	Bits [21:4] of base address assigned to this device. 4 MB Region.
Address	31:22	RW	0x0	Bits [31:22] of base address assigned to this device.

11.29.2.6 PCIE_CFG_BAR1

Address: 0x120000 + 0x5
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RW	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.29.2.7 PCIE_CFG_BAR2

Address: 0x120000 + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	10b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.
Prefetchable	3	RO	1b	The PEP address space does not contain volatile registers.
Zero	12:4	RO	0x0	Bits [12:4] of base address assigned to this device. 8 KB Region.
Address	31:13	RW	0x0	Bits [31:13] of base address assigned to this device.

11.29.2.8 PCIE_CFG_BAR3

Address: **0x120000 + 0x7**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RW	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.29.2.9 PCIE_CFG_BAR4

Address: **0x120000 + 0x8**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	10b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.
Prefetchable	3	RO	1b	The PEP address space does not contain volatile registers.
Zero	25:4	RO	0x0	Bits [25:4] of base address assigned to this device. 64 MB Region.
Address	31:26	RW	0x0	Bits [31:26] of base address assigned to this device.

11.29.2.10 PCIE_CFG_BAR5

Address: **0x120000 + 0x9**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RW	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.29.2.11 PCIE_CFG_CARDBUS

Address: **0x120000 + 0xA**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CardbusPtr	31:0	RO	0x0	Not used.

11.29.2.12 PCIE_CFG_SUBID

Address: 0x120000 + 0xB
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SubVendorID	15:0	RO	0x8086	This value can be loaded automatically from the EEPROM at power up or reset. A value of 0x8086 is the default for this field at power up if the EEPROM does not respond or is not programmed. All functions are initialized to the same value.
SubDeviceID	31:16	RO	0x0	This value can be loaded automatically from the EEPROM at power up with a default value of 0x0000.

11.29.2.13 PCIE_CFG_EXP_ROM

Address: 0x120000 + 0xC
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Enable	0	RO	0b	Not used.
Reserved	10:1	RSV	0x0	Reserved.
Address	31:11	RO	0x0	Not used.

11.29.2.14 PCIE_CFG_CAP_PTR

Address: 0x120000 + 0xD
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapabilityPtr	7:0	RO	0x40	The Capabilities Pointer field (Cap_Ptr) is an 8-bit field that provides an offset in the PCI configuration space for the location of the first item in the capabilities linked list. The value is 0x40, which is the address of the first entry: PCI power management.
Reserved	31:8	RSV	0x0	Reserved.

11.29.2.15 PCIE_CFG_RSVD

Address: 0x120000 + 0xE
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	31:0	RSV	0x0	Reserved.

11.29.2.16 PCIE_CFG_INT

Address: 0x120000 + 0xF
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
InterruptLine	7:0	RW	0xFF	Read/write register programmed by software to indicate which of the system interrupt request lines the device interrupt pin is bound to. Refer to the PCI definition for more details. Each PCI function has its own register.
InterruptPin	15:8	RW	0x0	Not used in this design.
MinGrant	23:16	RW	0x0	Min_Gnt not used. Hardwired to 0b.
MaxLatency	31:24	RW	0x0	Max_Lat not used. Hardwired to 0b.

11.29.2.17 PCIE_CFG_PM_CAP

Address: 0x120000 + 0x10
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapID	7:0	RO	0x01	Fixed. Identify PMC.
NextCapPtr	15:8	RO	0x70	Pointer to next capability.
PM_Version	18:16	RO	011b	Version. The device complies with the PCI PM specification revision 1.2.
PME_Clock	19	RO	0b	PME_Clock. Disabled. Hardwired to 0b.
Reserved	20	RSV	0b	Reserved.
DSI	21	RO	0b	The device requires its device driver to be executed following a transition to the D0 un-initialized state.
AUX_Current	23:22	RO	00b	AUX Current. Required current defined in the Data register.
D1_Support	24	RO	0b	The device does not support the D1 state.
D2_Support	25	RO	0b	The device does not support the D2 state.
PME_Support	29:26	RO	0x0	This 5-bit field indicates the power states in which the function can assert PME#.
Reserved	31:30	RSV	00b	Reserved.

11.29.2.18 PCIE_CFG_PM_CTRL

Address: 0x120000 + 0x11
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
PwrState	1:0	RW	00b	PowerState. This field is used to set and report the power state of a function as follows: 00b = DO 01b = D1Ignored if this value is written. 10b = D2Ignored if this value is written. 11b = D3

Address: **Atomicity:** **Reset Domains:**	**0x120000 + 0x11** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Reserved	2	RSV	0b	Reserved.
NoSoftRst	3	RO	1b	No_Soft_Reset. This bit is always set to 1b to indicate that the device does not perform an internal reset upon transitioning from D3hot to D0 via software control of the *PowerState* bits. Otherwise, the configuration context would be lost upon transition from the D3hot to the D0 state, and a full re-initialization sequence would be needed to return the device to the D0 Initialized state.
Reserved	7:4	RSV	0x0	Reserved.
PMEEn	8	RW	0b	PME_En. Writing a 1b to this register enables Wakeup.
Data_Select	12:9	RW	0x0	This 4-bit field is used to select which data is to be reported through the Data register and *Data_Scale* field. These bits are writeable only when power management is enabled via the EEPROM. Not used in this design.
Data_Scale	14:13	RO	00b	This field indicates the scaling factor that’s used when interpreting the value of the Data register. This field equals 01 (indicating 0.1 watt/units) and the *Data_Select* field is set to 0, 3, 4, 7, (or 8 for function 0). Otherwise, it equals 00. Not used in this design.
PME_Status	15	CW1	0b	This bit is set to 1b when the function detects a wake-up event independent of the state of the *PME_En* bit. Writing a 1b clears this bit.
BSE	23:16	RO	0x0	Bridge Support Extension. Not used in this design.
Data	31:24	RO	0x0	Data returned from Data Select. Not used in this design.

11.29.2.19 PCIE_CFG_PCIE_CAP

Address: **Atomicity:** **Reset Domains:**	**0x120000 + 0x1C** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CapID	7:0	RO	0x10	This field equals 0x10 indicating the linked list item as being the PCIe Capabilities registers.
NextCapPtr	15:8	RO	0xB0	Offset to the next capability item in the capability list.
CapabilityVersion	19:16	RO	0x2	Indicates the PCIe capability structure version. The device supports PCIe version 2 (also loaded from EEPROM).
DevicePortType	23:20	RO	0x0	Indicates the type of PCIe functions. All functions are native PCI functions with a value of 0000b.
SlotImplemented	24	RO	0b	The device does not implement slot options. Therefore, this field is hardwired to 0b.
InterrruptMessageNumber	29:25	RO	0x0	The device does not implement multiple MSI per function. Therefore, this field is hardwired to 0x0.
Reserved	31:30	RSV	00b	Reserved.

11.29.2.20 PCIE_CFG_PCIE_DEV_CAP

Address: 0x120000 + 0x1D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
MaxPayloadSize	2:0	RO	010b	Max Payload Size Supported. Indicates the maximum payload the device can support for TLPs. It is loaded from the EEPROM with a default value of 512 bytes. Supported values are 000b to 010b (128 bytes to 512 bytes).
PhantomFunctionSupport	4:3	RO	00b	Not supported by the device.
ExtendedTagField	5	RO	0b	Maximum supported size of the *Tag* field. The device supports a 5-bit *Tag* field for all functions.
EndpointL0Latency	8:6	RO	011b	Indicates the acceptable latency the device can withstand due to the transition from L0s state to the L0 state. Value loaded from EEPROM. A value of 011b equals 512 ns.
EndpointL1Latency	11:9	RO	110b	Indicates the acceptable latency the device can withstand due to the transition from L1 state to the L0 state. A value of 110b equals 32-64 µs.
AttentionButtonPreset	12	RO	0b	Hardwired in the device to 0b.
AttentionIndicatorPreset	13	RO	0b	Hardwired in the device to 0b.
PowerIndicatorPreset	14	RO	0b	Hardwired in the device to 0b.
RoleBasedErrorReporting	15	RO	1b	Role Based Error Reporting. Hard wired in the device to 1b.
Reserved	17:16	RSV	00b	Reserved.
SlotPowerLimit	25:18	RO	0x0	Used in upstream ports only. This value is set by the Set_Slot_Power_Limit Message.
SlotPowerScale	27:26	RO	00b	Slot Power Limit Scale. Used in upstream ports only. This value is set by the Set_Slot_Power_Limit Message.
FunctionLevelReset	28	RO	1b	A value of 1b indicates the Function supports the optional Function Level Reset (FLR) mechanism.
Reserved	31:29	RSV	000b	Reserved.

11.29.2.21 PCIE_CFG_PCIE_DEV_CTRL

Address: 0x120000 + 0x1E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CorrectableErrorReportingEnable	0	RW	0b	Enable error report.
NonFatalErrorReportingEnable	1	RW	0b	Enable error report.
FatalErrorReportingEnable	2	RW	0b	Enable error report.
UnsupportedRequestReportingEnable	3	RW	0b	Enable error report.
EnableRelaxedOrdering	4	RW	1b	If this bit is set, the FM10000 is permitted to set the Relaxed Ordering bit in the *Attribute* field of write transactions that do not need strong ordering. See the PCIE_CTRL_EXT register bit *RO_DIS* for more details.

Address:	**0x120000 + 0x1E**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
MaxPayloadSize	7:5	RW	000b	Sets the maximum TLP payload size for FM10000 functions. As a receiver, the FM10000 must handle TLPs as large as the set value. As a transmitter, the FM10000 must not generate TLPs exceeding the set value. The Max Payload Size field supported in the Device Capabilities register indicates permissible values that can be programmed. In ARI mode, Max Payload Size is determined solely by the field in function 0, while it is meaningless in the other function(s).
ExtendedTagfieldEnable	8	RO	0b	Not implemented in the FM10000.
PhantomFunctionsEnable	9	RO	0b	Not implemented in the FM10000.
AuxiliaryPowerPMEnable	10	SRW	0b	The bit is SRW to reflect the PCIe block capability to support auxiliary power, though the FM10000 may not have auxiliary power source.
EnableNoSnoop	11	RW	1b	When set, the Function is permitted to set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency.
MaxReadRequestSize	14:12	RW	010b	Sets maximum read request size for the FM10000 as a requester. 000b = MAX_128 001b = MAX_256 010b = MAX_512 011b = MAX_1024 100b = MAX_2048 101b = MAX_4096 All other values are reserved.
InitiateFLR	15	RW	0b	A write of 1b initiates FLR to the function. The value read by software from this bit is always 0b.
CorrectableDetected	16	CW1	0b	Indicates status of correctable error detection.
NonFatalErrorDetected	17	CW1	0b	Indicates status of non-fatal error detection.
FatalErrorDetected	18	CW1	0b	Indicates status of fatal error detection.
UnsupportedRequestDetected	19	CW1	0b	Indicates that the FM10000 received an unsupported request. This field is identical in all functions. The FM10000 cannot distinguish which function causes the error.
AuxPowerDetected	20	RO	0b	If Aux Power is detected, this field is set to 1b. It is a strapping signal from the periphery and is identical for all functions. Resets on LAN Power Good and PE_RST_N only.
TransactionPending	21	RO	0b	Indicates whether the FM10000 has ANY transactions pending. Transactions include completions for any outstanding non-posted request for all used traffic classes.
Reserved	31:22	RSV	0x0	Reserved.

11.29.2.22 PCIE_CFG_PCIE_LINK_CAP

Address: 0x120000 + 0x1F
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SupportedLinkSpeeds	3:0	RO	0x3	Supported Link Speeds. Indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 GbE link speed supported. 0010b = 5 GbE and 2.5 GbE link speeds supported. 0011b = 8 GbE, 5 GbE and 2.5 GbE link speeds supported. All other values are reserved.
MaxLinkWidth	9:4	RO	0x8	Max Link Width. Indicates the maximum link width. Depending on the SKU, The FM10000 supports a x1, x2, x4 and x8-link width with a default value of eight lanes. Defined encodings are: 000001b = x1 000010b = x2 000100b = x4 001000b = x8 All other values are reserved.
ActiveStateLinkPMSupport	11:10	RO	10b	Indicates the level of the active state of power management supported in the FM10000. Defined encodings are: 00b = No ASPM Support. 01b = Reserved. 10b = L1 supported. This is the default. 11b = Reserved. Value loaded from EEPROM.
L0sExitLatency	14:12	RO	011b	L0s Exit Latency. Indicates the exit latency from L0s to L0 state. 000b = Less than 64 ns 001b = 64 ns - 128 ns 010b = 128ns - 256 ns 011b = 256 ns - 512 ns 100b = 512 ns - 1 µs 101b = 1 µs - 2 µs 110b = 2 µs - 4 µs 111b = Reserved Value loaded from EEPROM.
L1ExitLatency	17:15	RO	110b	L1 Exit Latency. Indicates the exit latency from L1 to L0 state. 000b = Less than 1 µs 001b = 1 µs - 2 µs 010b = 2 µs - 4 µs 011b = 4 µs - 8 µs 100b = 8 µs - 16 µs 101b = 16 µs - 32 µs 110b = 32 µs - 64 µs 111b = L1 transition not supported. Value loaded from EEPROM.
ClockPowerManagement	18	RO	0b	Hardwired to 0b.
SurpriseDownErrorReportingCapable	19	RO	0b	Hardwired to 0b.

Address: 0x120000 + 0x1F
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
DataLinkLayerLinkActiveReportingCapable	20	RO	0b	Hardwired to 0b.
LinkBandwidthNotificationCapability	21	RO	0b	Hardwired to 0b.
ASPMOptionalCompliance	22	RO	1b	ASPM Optional Compliance. This bit must be set to 1b. Components that were implemented according to an earlier PCIe specification version have this bit set to 0b. Software is permitted to use the value of this bit to help determine whether to enable ASPM or run ASPM compliance tests.
Reserved	23	RSV	0b	Reserved.
PortNumber	31:24	RO	0x0	The PCIe port number for the given PCIe link. This field is set in the link training phase.

11.29.2.23 PCIE_CFG_PCIE_LINK_CTRL

Address: 0x120000 + 0x20
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ActStateLinkPM	1:0	RW	00b	Active State Link PM Control. This field controls the active state PM supported on the link. Link PM functionality is determined by the lowest common denominator of all functions. Defined encodings are: 00b = ASPM Disabled. 01b = Reserved. 10b = L1 entry Enabled. 11b = Reserved. In ARI mode, the ASPM is determined solely by the field in function 0 while it is meaningless in the other function(s).
Reserved	2	RSV	0b	Reserved.
RdCplBoundary	3	RW	0b	Read Completion Boundary.
LinkDisable	4	RO	0b	Link Disable. Reserved for endpoint devices. Hardwired to 0b.
RetrainClock	5	RO	0b	Retrain Clock. Not applicable for endpoint devices. Hardwired to 0b.
CommonClkCfg	6	RW	0b	Common Clock Configuration. 0b = Indicates that the FM10000 and the component at the other end of the link are operating with an asynchronous clock. This parameter affects the L0s exit latencies. 1b = Indicates that the FM10000 and the component at the other end of the link are operating with a common reference clock. In ARI mode, the common clock configuration is determined solely by the field in function 0 while it is meaningless in the other function(s).
ExtSync	7	RW	0b	When set, this bit forces an extended Tx of the FTS ordered set in FTS and an extra TS1 at the exit from L0s prior to entering L0.
EnClkPwrMan	8	RW	0b	Enable clock power management. Not supported in the FM10000.
Reserved	9	RSV	0b	Reserved.

Address:	**0x120000 + 0x20**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LinkBandwdMgmtIntEn	10	RO	0b	Link Bandwidth Management Interrupt Enable. Not supported in the FM10000. Hardwired to 0b.
LinkAutoBandwdIntEn	11	RO	0b	Link Autonomous Bandwidth Interrupt Enable. Not supported in the FM10000. Hardwired to 0b.
Reserved	15:12	RSV	0x0	Reserved.
CurrentLinkSpeed	19:16	RO	0x1	Current Link Speed. Indicates the negotiated Link speed of the given PCIe link. Defined encodings are: 0000b = Reserved. 0001b = 2.5 GbE PCIe link. 0010b = 5 GbE PCIe link. 0011b = 8 GbE PCIe link. All other values are reserved.
CurrentLinkWidth	24:20	RO	0x1	Negotiated Link Width. Indicates the negotiated width of the link. Relevant encodings for the FM10000 are: 00001b = x1 00010b = x2 00100b = x4 01000b = x8 All other encodings are reserved.
Reserved	26:25	RSV	00b	Reserved.
LnkTraining	27	RO	0b	Link Training. Indicates that link training is in progress. This field is not applicable and is reserved for endpoint devices, and is hardwired to 0b.
SlotClkCfg	28	RO	0b	Slot Clock Configuration. When set, indicates that the FM10000 uses the physical reference clock that the platform provides at the connector. This bit must be cleared if the FM10000 uses an independent clock. The Slot Clock Configuration bit is loaded from EEPROM.
DataLinkAct	29	RO	0b	Data Link Layer Link Active. Not supported in the FM10000. Hardwired to 0b.
LinkBandStat	30	RO	0b	Link Bandwidth Management Status. Not supported in the FM10000. Hardwired to 0b.
LinkAutoBandStat	31	RO	0b	Link Autonomous Bandwidth Status. This bit is not applicable and is reserved for endpoints.

11.29.2.24 PCIE_CFG_PCIE_DEV_CAP2

Address:	**0x120000 + 0x25**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CplTimeoutCfg	3:0	RO	0xF	Completion Timeout Ranges Supported. Indicates FM10000 support for the optional completion timeout programmability mechanism. Four time value ranges are defined: • Range A: 50 µs to 10 ms. • Range B: 10 ms to 250 ms. • Range C: 250 ms to 4 s. • Range D: 4 s to 64 s. Bits are set according to the following values to show the timeout value ranges that the FM10000 supports. 0000b = Completion timeout programming not supported. The FM10000 must implement a timeout value in the range of 50 µs to 50 ms. 0001b = Range A. 0010b = Range B. 0011b = Ranges A and B. 0110b = Ranges B and C. 0111b = Ranges A, B and C. 1110b = Ranges B, C and D. 1111b = Ranges A, B, C and D. All other encodings are reserved.
CplTimeoutDisSuppt	4	RO	1b	Completion Timeout Disable Supported.
Reserved	11:5	RSV	0x0	Reserved.
TPHCplSup	13:12	RO	01b	TPH Completer Supported. Value indicates Completer support for TPH or Extended TPH. Defined encodings are: 00b = TPH and Extended TPH Completer not supported. 01b = TPH Completer supported; Extended TPH Completer not supported. 10b = Reserved. 11b = Both TPH and Extended TPH Completer supported. When TPH is enabled, the device is enabled as both a requester and a completer. No control is given to support these features individually.
Reserved	31:14	RSV	0x0	Reserved.

11.29.2.25 PCIE_CFG_PCIE_DEV_CTRL2

Address:	**0x120000 + 0x26**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CplTimeoutVal	3:0	RW	0x0	Completion Timeout Value. For devices that support completion timeout programmability, this field enables system software to modify the completion timeout value. Defined encodings are: 0000b = Default range: 50 µs to 50 ms. ***Note:*** It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: 0001b = 50 µs to 100 µs 0010b = 1 ms to 10 ms Values available if Range B (10 ms to 250 ms) programmability range is supported: 0101b = 16 ms to 55 ms 0110b = 65 ms to 210 ms Values available if Range C (250 ms to 4 s) programmability range is supported: 1001b = 260 ms to 900 ms 1010b = 1 s to 3.5 s Values available if the Range D (4 s to 64 s) programmability range is supported: 1101b = 4 s to 13 s 1110b = 17 s to 64 s Values not defined are reserved. Software is permitted to change the value of this field at any time. For requests already pending when the completion timeout value is changed, hardware is permitted to use either the new or the old value for the outstanding requests and is permitted to base the start time for each request either on when this value was changed or on when each request was issued.
CplTimeoutDis	4	RW	0b	Completion Timeout Disable. When set to 1b, this bit disables the completion timeout mechanism. Software is permitted to set or clear this bit at any time. When set, the completion timeout detection mechanism is disabled. If there are outstanding requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding requests. If this is done, it is permitted to base the start time for each request on either the time this bit was cleared or the time each request was issued.
Reserved	31:5	RSV	0x0	Reserved.

11.29.2.26 PCIE_CFG_PCIE_LINK_CTRL2

Address: **0x120000 + 0x28**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
TargetSpeed	3:0	SRW	0x3	Target Link Speed. This field is used to set the target compliance mode speed when software is using the Enter Compliance bit to force a link into compliance mode. Defined encodings are: 0001b = 2.5 GbE target link speed. 0010b = 5 GbE target link speed. 0011b = 8 GbE target link speed. All other encodings are reserved. If a value is written to this field that does not correspond to a speed included in the Supported Link Speeds field, the result is undefined. The default value of this field is the highest link speed supported by the FM10000 (as reported in the Supported Link Speeds field of the Link Capabilities register).
EnterCompliance	4	SRW	0b	Enter Compliance. Software is permitted to force a link to enter compliance mode at the speed indicated in the Target Link Speed field by setting this bit to 1b in both components on a link and then initiating a hot reset on the link. The default value of this field following a fundamental reset is 0b.
HwAutoSpeedDis	5	SRW	0b	Hardware Autonomous Speed Disable. When set to 1b, this bit disables hardware from changing the link speed for reasons other than attempting to correct unreliable link operation by reducing link speed.
SelectDeEmphasis	6	RO	0b	Selectable De-Emphasis.
TransmitMargin	9:7	SRW	000b	Transmit Margin. This field controls the value of the non- de-emphasized voltage level at the Transmitter pins. Defined encodings are: 000b = Normal operating range. 001b = 800-1200 mV for full swing and 400-700 mV for half-swing. 010b = (n-1) — Values must be monotonic with a non-zero slope. The value of n must be greater than 3 and less than 7. At least two of these must be below the normal operating range of n: 200-400 mV for full-swing and 100-200 mV for half-swing. 111b = (n) — Reserved. All other values are reserved.
EnterModeComp	10	SRW	0b	Enter Modified Compliance. When this bit is set to 1b, the device transmits modified compliance pattern if the LTSSM enters Polling.Compliance state.
CompSOS	11	SRW	0b	Compliance SOS. When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns.

Address: **0x120000 + 0x28**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
PCIE_CAP_COMPLIANCE_PRESET	15:12	SRW	0x0	Compliance preset/de-emphasis for 8.0 GT/s Data Rate. This field sets the transmitter preset in Polling. Compliance state if the entry occurred due to the Enter Compliance bit being 1b. For 5.0 GT/s data rate, this field sets the de-emphasis level in Polling. Compliance state if the entry occurred due to the Enter Compliance bit being 1b. When the Link is operating at 2.5 GT/s, setting this bit field has no effect. Defined Encodings are: 0000b = 6 dB 0001b = 3.5 dB
CAPCURRDEEMPHASIS	16	RO	1b	Current De-emphasis Level. When the Link is operating at 5.0 GT/s speed, this bit reflects the level of de-emphasis. Encodings are: 0b = 6 dB 1b = 3.5 dB The value in this bit is undefined when the Link is not operating at 5.0 GT/s speed. The default value of this field is changed automatically by hardware upon link setup.
Reserved	31:17	RSV	0x0	Reserved.

11.29.2.27 PCIE_CFG_MSIX_CAP

More than one MSI-X capability structure per function is prohibited while a function is permitted to have both an MSI and an MSI-X capability structure.

In contrast to the MSI capability structure, which directly contains all of the control/status information for the function's vectors, the MSI-X capability structure instead points to an MSI-X table structure and an MSI-X Pending Bit Array (PBA) structure, each residing in memory space.

Each structure is mapped by a BAR belonging to the function that begins at 0x10 in the configuration space. A BAR Indicator Register (BIR) indicates which BAR and a Qword-aligned offset indicates where the structure begins relative to the base address associated with the BAR.The BAR is 64-bit, but must map to the memory space. A function is permitted to map both structures with the same BAR or map each structure with a different BAR.

The MSI-X table structure typically contains multiple entries, each consisting of several fields: Message Address, Message Upper Address, Message Data, and Vector Control. Each entry is capable of specifying a unique vector.

The PBA structure contains the function's pending bits, one per table entry, organized as a packed array of bits within Qwords. The last Qword is not necessarily fully populated.

To request service using a given MSI-X table entry, a function performs a Dword memory write transaction using:

- The contents of the Message Data field entry for data
- The contents of the Message Upper Address field for the upper 32 bits of the address
- The contents of the Message Address field entry for the lower 32 bits of the address

A memory read transaction from the address targeted by the MSI-X message produces undefined results.

The MSI-X table and MSI-X PBA are permitted to co-reside within a naturally aligned 4 KB address range, though they must not overlap with each other.

MSI-X table entries and Pending bits are each numbered 0 through N-1, where N-1 is indicated by the Table Size field in the MSI-X Message Control register. For a given arbitrary MSI-X table entry K, its starting address can be calculated with the formula:

Entry starting address = Table base + K*16

For the associated *Pending* bit K, its address for Qword access and bit number within that Qword can be calculated with the formulas:

Qword address = PBA base + (K div 64)*8

Qword bit# = K mod 64

Software that chooses to read *Pending* bit K with Dword accesses can use these formulas:

Dword address = PBA base + (K div 32)*4

Dword bit# = K mod 32

Address: 0x120000 + 0x2C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapID	7:0	RO	0x11	Capability ID.
NextCapPtr	15:8	RO	0xD0	Next capability offset.
TableSize	26:16	RO	0xFF	System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. The device supports up to 256 different interrupt vectors for the physical function.
Reserved	29:27	RSV	000b	Reserved.
FunctionMask	30	RO	0b	Function Mask. 0b = Each vector’s Mask bit determines whether the vector is masked or not. 1b = All of the vectors associated with the function are masked, regardless of their per-vector Mask bit states. Setting or clearing the MSI-X Function Mask bit has no effect on the state of the per-vector Mask bits.
MSIX_Enable	31	RW	0b	MSIX Enable. 0b = The function is prohibited from using MSI-X to request service. 1b = If set, and the MSI Enable bit in the MSI Message Control register is 0b, the function is permitted to use MSI-X to request service and is prohibited from using its INTx# pin. System configuration software sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a function’s service request.

11.29.2.28 PCIE_CFG_MSIX_TABLE_OFFSET

Address: 0x120000 + 0x2D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
BIR	2:0	RO	010b	Indicates which one of a functions BARs, beginning at 0x10 in the configuration space, is used to map the functions MSI-X table into the memory space. BIR values 0...5 correspond to BARs 0x10...0x 24 respectively.
Offset	31:3	RO	0x0	Used as an offset from the address contained in one of the functions BARs to point to the base of the MSI-X table. The lower three Table BIR bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. ***Note:*** This field is read only.

11.29.2.29 PCIE_CFG_MSIX_PBA

Address: 0x120000 + 0x2E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
BIR	2:0	RO	010b	Indicates which one of a functions BARs, beginning at 0x10 in the configuration space, is used to map the functions MSI-X PBA into the memory space. BIR values 0...5 correspond to BARs 0x10...0x 24 respectively.
Offset	31:3	RO	0x0200	Used as an offset from the address contained in one of the functions BARs to point to the base of the MSI-X PBA. The lower three PBA BIR bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. ***Note:*** This field is read only.

11.29.2.30 PCIE_CFG_VPD_CAP

Address: 0x120000 + 0x34
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapID	7:0	RO	0x3	This field equals 0x11, indicating the linked list item as being the VPD registers.
NextCapPtr	15:8	RO	0x00	Offset to the next capability item in the capability list. A 0x00 value indicates that it is the last item in the capability-linked list.
Address	30:16	RW	0x0	Dword-aligned byte address of the VPD area in the NVM to be accessed. The register is read/write, and the initial value at power-up is indeterminate. The two LSBs are RO as zero.
Write	31	RW	0b	Flag used to indicate when the transfer of data between the VPD Data register and the storage component completes. The Flag register is written when the VPD Address register is written. 0b = Read Set by hardware when data is valid. 1b = Write Cleared by hardware when data write is complete. The VPD address and data should not be modified before the action is done.

11.29.2.31 PCIE_CFG_VPD_DATA

Address: 0x120000 + 0x35
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	VPD Data. VPD data can be read or written through this register. The LS byte of this register (at offset 4 in this capability structure) corresponds to the byte of VPD at the address specified by the VPD Address register. The data read from or written to this register uses the normal PCI byte transfer capabilities. Four bytes are always transferred between this register and the VPD storage component. Reading or writing data outside of the VPD space in the storage component is not allowed. In a write access, the data should be set before the address and the flag is set.

11.29.2.32 PCIE_CFG_AER_HDR

Address: 0x120000 + 0x40
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ExtCapId	15:0	RO	0x1	Extended Capability ID. PCIe extended capability ID indicating advanced error reporting capability.
Version	19:16	RO	0x2	Version Number. PCIe advanced error reporting extended capability version number.
NextCapPtr	31:20	RO	0x148	Next Capability Pointer.

11.29.2.33 PCIE_CFG_AER_UNERR_STATUS

The Uncorrectable Error Status register reports error status of individual uncorrectable error sources on a PCIe device. An individual error status bit that is set to 1b indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit. Register is cleared by LAN_PWR_GOOD.

Address: 0x120000 + 0x41
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	CW1	0b	Data Link Protocol Error Status.
SurpriseDownError	5	RO	0b	Surprise Down Error Status.
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	CW1	0b	Poisoned TLP Status.
FlowControlError	13	CW1	0b	Flow Control Protocol Error Status.
CompletionTimeout	14	CW1	0b	Completion Timeout Status.
CompleterAbort	15	CW1	0b	Completer Abort Status.
UnexpecteCompl	16	CW1	0b	Unexpected Completion Status.
RxOverflow	17	CW1	0b	Receiver Overflow Status.

Address: **0x120000 + 0x41**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
MalformedTLP	18	CW1	0b	Malformed TLP Status.
ECRCError	19	CW1	0b	ECRC Error Status.
UnsupportedRequestError	20	CW1	0b	Unsupported Request Error Status.
ACSViolation	21	RO	0b	ACS Violation Status.
Reserved	31:22	RSV	0x0	Reserved.

11.29.2.34 PCIE_CFG_AER_UNERR_MASK

The Uncorrectable Error Mask register controls reporting of individual uncorrectable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. Note that there is a mask bit per bit of the Uncorrectable Error Status register. This register's fields are sticky (reset on power up only).

Address: **0x120000 + 0x42**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	RW	0b	Data Link Protocol Error Mask.
SurpriseDownError	5	RO	0b	Surprise Down Error Mask.
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	RW	0b	Poisoned TLP Mask.
FlowControlError	13	RW	0b	Flow Control Protocol Error Mask.
CompletionTimeout	14	RW	0b	Completion Timeout Mask.
CompleterAbort	15	RW	0b	Completer Abort Mask.
UnexpecteCompl	16	RW	0b	Unexpected Completion Mask.
RxOverflow	17	RW	0b	Receiver Overflow Mask.
MalformedTLP	18	RW	0b	Malformed TLP Mask.
ECRCError	19	RW	0b	ECRC Error Mask.
UnsupportedRequestError	20	RW	0b	Unsupported Request Error Mask.
Reserved	21	RSV	0b	Reserved
UncorrectableInternalError	22	RW	1b	
Reserved	23	RSV	0b	Reserved
AtomicOpEggressBlocked	24	RO	0b	
TLPPrefixBlockedError	25	RO	0b	
Reserved	31:26	RSV	0x0	Reserved.

11.29.2.35 PCIE_CFG_AER_UNERR_SEVERITY

The Uncorrectable Error Severity register controls whether an individual uncorrectable error is reported as a fatal error. An uncorrectable error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered non-fatal. This register's fields are sticky (reset on power up only).

Address: 0x120000 + 0x43
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	RW	1b	Data Link Protocol Error Severity.
SurpriseDownError	5	RO	1b	
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	RW	0b	Poisoned TLP Severity.
FlowControlError	13	RW	1b	Flow Control Protocol Error Severity.
CompletionTimeout	14	RW	0b	Completion Timeout Severity.
CompleterAbort	15	RW	0b	Completer Abort Severity.
UnexpecteCompl	16	RW	0b	Unexpected Completion Severity.
RxOverflow	17	RW	1b	Receiver Overflow Severity.
MalformedTLP	18	RW	1b	Malformed TLP Severity.
ECRCError	19	RW	0b	ECRC Error Severity.
UnsupportedRequestError	20	RW	0b	Unsupported Request Error Severity.
Reserved	21	RSV	0b	Reserved.
UncorrectableInternalError	22	RW	1b	
Reserved	23	RSV	0b	Reserved.
AtomicOpEggressBlocked	24	RO	0b	
TLPPrefixBlockedError	25	RO	0b	
Reserved	31:26	RSV	0x0	Reserved.

11.29.2.36 PCIE_CFG_AER_COERR_STATUS

The Correctable Error Status register reports error status of individual correctable error sources on a PCIe device. When an individual error status bit is set to 1b it indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit. Register is cleared by LAN_PWR_GOOD.

Address: 0x120000 + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
RxError	0	CW1	0b	Receiver Error Status.
Reserved	5:1	RSV	0x0	Reserved.
BadTLP	6	CW1	0b	Bad TLP Status.
BadDLLP	7	CW1	0b	Bad DLLP Status.
ReplayRollover	8	CW1	0b	REPLAY_NUM Rollover Status.
Reserved	11:9	RSV	000b	Reserved.

Address: 0x120000 + 0x44
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ReplayTimeout	12	CW1	0b	Replay Timer Timeout Status.
NonFatalError	13	CW1	0b	Advisory Non-Fatal Error Status.
CorrectableInternalError	14	CW1	0b	Corrected Internal Error Status.
Reserved	31:15	RSV	0x0	Reserved.

11.29.2.37 PCIE_CFG_AER_COERR_MASK

The Correctable Error Mask register controls reporting of individual correctable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. There is a mask bit per bit in the Correctable Error Status register. This register’s fields are sticky (reset on power up only).

Address: 0x120000 + 0x45
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
RxError	0	RW	0b	Receiver Error Status.
Reserved	5:1	RSV	0x0	Reserved.
BadTLP	6	RW	0b	Bad TLP Status.
BadDLLP	7	RW	0b	Bad DLLP Status.
ReplayRollover	8	RW	0b	REPLAY_NUM Rollover Status.
Reserved	11:9	RSV	000b	Reserved.
ReplayTimeout	12	RW	0b	Replay Timer Timeout Status.
NonFatalError	13	RW	1b	Advisory Non-Fatal Error Status.
CorrectableInternalError	14	RW	1b	Corrected Internal Error Status.
Reserved	31:15	RSV	0x0	Reserved.

11.29.2.38 PCIE_CFG_AER_CTRL

Address: 0x120000 + 0x46
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ErrorIndex	4:0	RO	0x0	Vector pointing to the first recorded error in the Uncorrectable Error Status register. This is a read-only field that identifies the bit position of the first uncorrectable error reported in the Uncorrectable Error Status register.
ECRCGenCap	5	RO	1b	ECRC Generation Capable. If set, this bit indicates that the function is capable of generating ECRC. This bit is loaded from EEPROM.
ECRCGenEn	6	RW	0b	ECRC Generation Enable. When set, ECRC generation is enabled. This field is sticky (reset on power up only).

Address: 0x120000 + 0x46
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ECRCCheckCap	7	RO	1b	ECRC Check Capable. If set, this bit indicates that the function is capable of checking ECRC. This bit is loaded from EEPROM.
ECRCCheckEn	8	RW	0b	ECRC Check Enable. When set Set, ECRC checking is enabled. This field is sticky (reset on power up only).
Reserved	31:9	RSV	0x0	Reserved.

11.29.2.39 PCIE_CFG_AER_HEADER_LOG0

The header log register captures the header for the transaction that generated an error.

Address: 0x120000 + 0x47
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.29.2.40 PCIE_CFG_AER_HEADER_LOG1

The header log register captures the header for the transaction that generated an error.

Address: 0x120000 + 0x48
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.29.2.41 PCIE_CFG_AER_HEADER_LOG2

The header log register captures the header for the transaction that generated an error.

Address: 0x120000 + 0x479
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.29.2.42 PCIE_CFG_AER_HEADER_LOG3

The header log register captures the header for the transaction that generated an error.

Address:	**0x120000 + 0x4A**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.29.2.43 PCIE_CFG_SPD_HDR

The PCIe device serial number capability is an optional extended capability that can be implemented by any PCIe device.The device serial number is a read-only 64-bit value that is unique for a given PCIe device.

All multi-function devices that implement this capability must implement it for function 0; other functions that implement this capability must return the same device serial number value as that reported by function 0.

Address:	**0x120000 + 0x52**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CapId	15:0	RO	0x3	Capability ID.
Version	19:16	RO	0x1	Version.
NextCapPtr	31:20	RO	0x158	Next capability pointer.

11.29.2.44 PCIE_CFG_SPD_NUMBER_L

The Serial Number register is a 64-bit field that contains the IEEE defined 64-bit Extended Unique Identifier (EUI-64*).

PCIE_CFG_SPD_NUMBER_L.SerialNumber[0] = EUI[7] = LSB of Extension ID
PCIE_CFG_SPD_NUMBER_L.SerialNumber[1] = EUI[6] = Middle byte
PCIE_CFG_SPD_NUMBER_L.SerialNumber[2] = EUI[5] = MSB of Extension ID
PCIE_CFG_SPD_NUMBER_L.SerialNumber[3] = EUI[4] = 0xFF
PCIE_CFG_SPD_NUMBER_H.SerialNumber[0] = EUI[3] = 0xFF
PCIE_CFG_SPD_NUMBER_H.SerialNumber[1] = EUI[2] = LSB of Company ID
PCIE_CFG_SPD_NUMBER_H.SerialNumber[2] = EUI[1] = Middle byte
PCIE_CFG_SPD_NUMBER_H.SerialNumber[3] = EUI[0] = MSB of Company ID

For example, assume that the company ID is (Intel) 00-A0-C9 and the extension identifier is 23-45-67. The register then contains:

PCIE_CFG_SPD_NUMBER_L = 0xFF234567 PCIE_CFG_SPD_NUMBER_H = 0x00A0C9FF

Refer to the official document defining the EUI numbering:

http://standards.ieee.org/regauth/oui/tutorials/EUI64.html

Address: 0x120000 + 0x53
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SerialNumber[0..3]	31:0	RO	0x0	(4 x 8-bits) EUI bytes 7..4 (reverse order). *SerialNumber*[0] = EUI[7] *SerialNumber*[1] = EUI[6] *SerialNumber*[2] = EUI[5] *SerialNumber*[3] = EUI[4]

11.29.2.45 PCIE_CFG_SPD_NUMBER_H

See PCIE_CFG_SPD_NUMBER_L for details.

Address: 0x120000 + 0x54
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SerialNumber[0..3]	31:0	RO	0x0	(4 x 8-bits) EUI bytes 3..0 (reverse order). The serial number registers PCIE_CFG_SPD_NUMBER_L and PCIE_CFG_SPD_NUMBER_H combine to use the 48-bit Ethernet MAC address according to the following definition: *SerialNumber*[0] = EUI[3] *SerialNumber*[1] = EUI[2] *SerialNumber*[2] = EUI[1] *SerialNumber*[3] = EUI[0]

11.29.2.46 PCIE_CFG_ARI_HDR

In order to allow more than eight functions per endpoint without requesting an internal switch, as is usually needed in virtualization scenarios, the PCI-SIG defines a new capability that allows a different interpretation of the Bus, Device, and Function fields.

Address: 0x120000 + 0x56
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapId	15:0	RO	0x000E	Capability ID.
Version	19:16	RO	0x1	Version.
NextCapPtr	31:20	RO	0x168	Next capability pointer.

11.29.2.47 PCIE_CFG_ARI_CTRL

Address:	**0x120000 + 0x57**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Reserved	7:0	RSV	0x0	Reserved.
NFP	15:8	RO	0x0	Next function pointer. Zero, as there is none.
Reserved	31:16	RSV	0x0	Reserved.

11.29.2.48 PCIE_CFG_SPCIE_HDR

The Secondary PCI Express Extended Capability structure is required for all Ports and RCRBs that support a Link speed of 8.0 GT/s or higher. For Multi-Function Upstream Ports, this capability must be implemented in Function 0 and must not be implemented in other Functions.

Address:	**0x120000 + 0x5A**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CapId	15:0	RO	0x0019	Capability ID.
Version	19:16	RO	0x1	Version.
NextCapPtr	31:20	RO	0x188	Next capability pointer.

11.29.2.49 PCIE_CFG_SPCIE_LINK_CTRL3

Address:	**0x120000 + 0x5B**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
PerfEq	0	RSV	0b	
LinkEqIntEn	1	RSV	0b	
Reserved	31:2	RSV	0x0	Reserved.

11.29.2.50 PCIE_CFG_SPCIE_ERR_STS

Each bit indicates if the corresponding Lane detected a Lane-based error. A value of 1b indicates that a Lane based-error was detected on the corresponding Lane Number.

Address:	**0x120000 + 0x5C**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
LaneErrSts	7:0	CW1	0x0	
Reserved	31:8	RSV	0x0	Reserved.

11.29.2.51 PCIE_CFG_SPCIE_LINK_EQ01

The Equalization Control register consists of control fields required for per Lane equalization and number of entries in this register are sized by Maximum Link Width.

Address:	**0x120000 + 0x5D**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DwnTxPreset0	3:0	RO	0x0	
DwnRxPresetHint0	6:4	RO	0x0	
Reserved	7	RSV	0b	Reserved.
UpTxPreset0	11:8	RO	0xF	
UpRxPresetHint0	14:12	RO	111b	
Reserved	15	RSV	0b	Reserved.
DwnTxPreset1	19:16	RO	0x0	
DwnRxPresetHint1	22:20	RO	0x0	
Reserved	23	RSV	0b	Reserved.
UpTxPreset1	27:24	RO	0xF	
UpRxPresetHint1	30:28	RO	111b	
Reserved	31	RSV	0b	Reserved.

11.29.2.52 PCIE_CFG_SPCIE_LINK_EQ23

Address:	**0x120000 + 0x5E**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DwnTxPreset2	3:0	RO	0x0	
DwnRxPresetHint2	6:4	RO	0x0	
Reserved	7	RSV	0b	Reserved.
UpTxPreset2	11:8	RO	0xF	
UpRxPresetHint2	14:12	RO	111b	
Reserved	15	RSV	0b	Reserved.
DwnTxPreset3	19:16	RO	0x0	
DwnRxPresetHint3	22:20	RO	0x0	
Reserved	23	RSV	0b	Reserved.
UpTxPreset3	27:24	RO	0xF	
UpRxPresetHint3	30:28	RO	111b	
Reserved	31	RSV	0b	Reserved.

11.29.2.53 PCIE_CFG_SPCIE_LINK_EQ45

Address: 0x120000 + 0x5F
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
DwnTxPreset4	3:0	RO	0x0	
DwnRxPresetHint4	6:4	RO	0x0	
Reserved	7	RSV	0b	Reserved.
UpTxPreset4	11:8	RO	0xF	
UpRxPresetHint4	14:12	RO	111b	
Reserved	15	RSV	0b	Reserved.
DwnTxPreset5	19:16	RO	0x0	
DwnRxPresetHint5	22:20	RO	0x0	
Reserved	23	RSV	0b	Reserved.
UpTxPreset5	27:24	RO	0xF	
UpRxPresetHint5	30:28	RO	111b	
Reserved	31	RSV	0b	Reserved.

11.29.2.54 PCIE_CFG_SPCIE_LINK_EQ67

Address: 0x120000 + 0x60
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
DwnTxPreset6	3:0	RO	0x0	
DwnRxPresetHint6	6:4	RO	0x0	
Reserved	7	RSV	0b	Reserved.
UpTxPreset6	11:8	RO	0xF	
UpRxPresetHint6	14:12	RO	111b	
Reserved	15	RSV	0b	Reserved.
DwnTxPreset7	19:16	RO	0x0	
DwnRxPresetHint7	22:20	RO	0x0	
Reserved	23	RSV	0b	Reserved.
UpTxPreset7	27:24	RO	0xF	
UpRxPresetHint7	30:28	RO	111b	
Reserved	31	RSV	0b	Reserved.

11.29.2.55 PCIE_CFG_SRIOV_HDR

Address: 0x120000 + 0x62
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ID	15:0	RO	0x0010	PCIe Extended Capability ID. PCIe extended capability ID for the SR-IOV capability.
Version	19:16	RO	0x1	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 0x1 for this version of the specification.
Nextpointer	31:20	RO	0x1C8	Next Capability Offset. This field contains the offset to the next PCIe extended capability structure or 0x000 if no other items exist in the linked list of capabilities. This field must be set to 0x000 in NVM/BSM by default to not expose the TPH capabilities.

11.29.2.56 PCIE_CFG_SRIOV_CAP

Address: 0x120000 + 0x63
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	0	RSV	0b	Reserved.
ARICapHierPre	1	RO	1b	ARI Capable Hierarchy Preserved. If set, the ARI Capable Hierarchy bit is preserved across certain power state transitions.
Reserved	31:2	RSV	0x0	Reserved.

11.29.2.57 PCIE_CFG_SRIOV_CTRL

Address: 0x120000 + 0x64
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
VFE	0	RW	0b	VF Enable/Disable. VF Enable manages the assignment of VFs to the associated PF. 0b = Disabled — VFs must be disabled and not visible in the PCIe fabric; VFs cannot respond to or issue PCIe transactions. 1b = Enabled — VFs must be enabled, associated with the PF, and exists in the PCIe fabric. When enabled, VFs must respond to and can issue PCIe transactions following all other rules for PCIe functions. In addition, if VF Enable is cleared after having been set, all of the VFs must no longer: • Issue PCIe transactions • Respond to configuration space or memory space accesses. The behavior must be as if an FLR was issued to each of the VFs. Specifically, VFs must not retain any context after VF Enable has been cleared. Any errors already logged via PF error reporting registers, remain logged. However, no new VF errors must be logged after VF Enable is cleared.
Reserved	2:1	RSV	00b	Reserved.

Address:	0x120000 + 0x64			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VFMSE	3	RW	0b	Memory Space Enable for Virtual Functions. VF MSE controls memory space enable for all VFs associated with this PF as with the Memory Space Enable bit in a functions PCI command register. The default value for this bit is 0b. When VF Enable is 1b, virtual function memory space access is permitted only when VF MSE is Set. VFs follow the same error reporting rules as defined in the base specification if an attempt is made to access a virtual functions memory space when VF Enable is 1b and VF MSE is 0b. ***Note:*** Virtual functions memory space cannot be accessed when VF Enable is 0b. Thus, VF MSE is “don't care” when VF Enable is 0b, however, software may choose to set VF MSE after programming the VF BARn registers, prior to setting VF Enable to 1b.
VFARI	4	RW	0b	VF ARI Enable. Device can locate VFs in function numbers 8 to 255 of the captured bus number.
Reserved	31:5	RSV	0x0	Reserved.

11.29.2.58 PCIE_CFG_SRIOV_CFG

Address:	0x120000 + 0x65			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
InitialVFs	15:0	RO	0x40	*InitialVFs* indicates the number of VFs that are initially associated with the PF. If VF Migration Capable is cleared, this field must contain the same value as *TotalVFs*. This field is loaded from EEPROM.
TotalVFs	31:16	RO	0x40	*TotalVFs* defines the maximum number of VFs that can be associated with the PF. This field reflects the setting made in the *InitialVFs* field.

11.29.2.59 PCIE_CFG_SRIOV_NUM

Address:	0x120000 + 0x66			
Atomicity:	32			
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
NumVFs	15:0	RW	0x0	*NumVFs* defines the number of VFs software has assigned to the PF. Software sets *NumVFs* to any value between one and the *TotalVFs* as part of the process of creating VFs. *NumVFs* VFs must be visible in the PCIe fabric after both *NumVFs* is set to a valid value and VF *Enable* is set to 1b.
FDL	23:16	RO	0x0	Function Dependency Link, Defines dependencies between physical functions allocation. In the FM10000 there are no constraints.
Reserved	31:24	RSV	0x0	Reserved.

11.29.2.60 PCIE_CFG_SRIOV_MAP

Address:	0x120000 + 0x67
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
FVO	15:0	RO	0x1	First VF Offset. Defines the requestor ID (RID) offset of the first VF that is associated with the PF that contains this capability structure. The content of this field is valid only when VF Enable is set. If VF Enable is 0b, the contents are undefined. If the ARI *Enable* bit is set, this field changes to 0x80. ***Note:*** Enabling SR-IOV requires ARI *Enable* bit be set.
VFS	31:16	RO	0x1	VF Stride. Defines the requestor ID (RID) offset from one VF to the next one for all VFs associated with the PF that contains this capability structure. The next VFs 16-bit RID is calculated by adding the contents of this field to the RID of the current VF. The contents of this field is valid only when VF Enable is set and *NumVFs* is non-zero. If VF Enable is 0b or if *NumVFs* is zero, the contents are undefined.

11.29.2.61 PCIE_CFG_SRIOV_DEVID

Address:	0x120000 + 0x68
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	15:0	RSV	0x0	Reserved.
DEVID	31:16	RO	0x15A5	VF Device ID. This field contains the device ID that should be presented for every VF to the Virtual Machine (VM). The value of this field can be read from the IOV Control Word 2 in the EEPROM.

11.29.2.62 PCIE_CFG_SRIOV_PAGE_SUP

Address:	0x120000 + 0x69
Atomicity:	32
Reset Domains:	DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
SuppPageSize	31:0	RO	0x553	For PFs that supports the stride-based BAR mechanism, this field defines the supported page sizes. This PF supports a page size of 2^(n+12) if bit n is set. For example, if bit 0 is set, the Endpoint (EP) supports 4 KB page sizes. Endpoints are required to support 4 KB, 8 KB, 64 KB, 256 KB, 1 MB and 4 MB page sizes. All other page sizes are optional.

11.29.2.63 PCIE_CFG_SRIOV_PAGE_CFG

Address: 0x120000 + 0x6A
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Pagesize	31:0	RW	0x1	Defines the page size the system uses to map the PF's and associated VFs' memory addresses. Software must set the value of the System Page Size to one of the page sizes set in the Supported Page Sizes field. As with Supported Page Sizes, if bit n is set in System Page Size, the PF and its associated VFs are required to support a page size of 2^(n+12). For example, if bit 1 is set, the system is using an 8 KB page size. The results are undefined if more than one bit is set in System Page Size. The results are undefined if a bit is set in System Page Size that is not set in Supported Page Sizes. When System Page Size is set, the PF and associated VFs are required to align all BAR resources on a System Page Size boundary. Each BAR size, including VF BARn Size (described later) must be aligned on a System Page Size boundary. Each BAR size, including VF BARn Size must be sized to consume a multiple of System Page Size bytes. All fields requiring page size alignment within a function must be aligned on a System Page Size boundary. VF *Enable* must be zero when System Page Size is set. The results are undefined if System Page Size is set when VF *Enable* is set.

11.29.2.64 PCIE_CFG_SRIOV_BAR0

Address: 0x120000 + 0x6B
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Mem	0	RO	0b	0b indicates memory space.
MemType	2:1	RO	10b	Indicates the address space size. Only 64-bit supported. This bit is loaded from EEPROM.
PrefetchMem	3	RO	1b	This bit is loaded from EEPROM. 0b = Non-prefetchable space. 1b = Prefetchable space.
MemAddrSpace	31:4	RW	0x0	Which bits are RW bits and which are dependent on the memory mapping window size. The size is a maximum between 2 MB and the page size.

11.29.2.65 PCIE_CFG_SRIOV_BAR1

Address: 0x120000 + 0x6C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Addresshigh	31:0	RW	0x0	MSB part of BAR0.

11.29.2.66 PCIE_CFG_SRIOV_BAR2

Address: **0x120000 + 0x6D**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Mem	0	RO	0b	0b indicates memory space.
MemType	2:1	RO	10b	Indicates the address space size. Only 64-bit supported. This bit is loaded from EEPROM.
PrefetchMem	3	RO	1b	This bit is loaded from EEPROM. 0b = Non-prefetchable space. 1b = Prefetchable space.
MemAddrSpace	31:4	RW	0x0	Which bits are RW bits and which are dependent on the memory mapping window size. The size is a maximum between 8 KB and the page size.

11.29.2.67 PCIE_CFG_SRIOV_BAR3

Address: **0x120000 + 0x6E**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Addresshigh	31:0	RW	0x0	MSB part of BAR2.

11.29.2.68 PCIE_CFG_SRIOV_BAR4

Address: **0x120000 + 0x6F**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Mem	0	RO	0b	0b indicates memory space.
MemType	2:1	RO	00b	Indicates the address space size. Only 64-bit supported. This bit is loaded from EEPROM.
PrefetchMem	3	RO	0b	This bit is loaded from EEPROM. 0b = Non-prefetchable space. 1b = Prefetchable space.
MemAddrSpace	31:4	RO	0x0	Which bits are RW bits and which are dependent on the memory mapping window size. The size is a maximum between 16 KB and the page size. It is RO since VFs have no access to BAR4.

11.29.2.69 PCIE_CFG_SRIOV_BAR5

Address:	**0x120000 + 0x70**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Addresshigh	31:0	RO	0x0	MSB part of BAR4. It is RO since VFs have no access to BAR4.

11.29.2.70 PCIE_CFG_SRIOV_MIG

SR-IOV VF Migration State Array Offset Register.

Address:	**0x120000 + 0x71**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Reserved	31:0	RSV	0x0	Not implemented in this device.

11.29.2.71 PCIE_CFG_TPH_HDR

Address:	**0x120000 + 0x72**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
ID	15:0	RO	0x0017	PCIe Extended Capability ID. PCIe extended capability ID for the SR-IOV capability.
Version	19:16	RO	0x1	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 0x1 for this version of the specification.
Nextpointer	31:20	RO	0x000	Next Capability Offset. This field contains the offset to the next PCIe extended capability structure or 0x000 if no other items exist in the linked list of capabilities. This field is set to 0x1D8 by NVM/BSM to expose the ACS capability to host.

11.29.2.72 PCIE_CFG_TPH_CAP

Address:	**0x120000 + 0x73**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
NoStMode	0	RO	1b	The FM10000 does support.
InterruptVector	1	RO	0b	The FM10000 does not support Interrupt Vector Mode of operation
DeviceSpecific	2	RO	0b	the FM10000 does not support Device Specific Mode of operation
Reserved	7:3	RSV	0x0	Reserved.
ExtendedTPH	8	RO	0b	Cleared to indicate that the function is not capable of generating requests with Extended TPH TLP Prefix

Address: 0x120000 + 0x73 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
StTableLocation	10:9	RO	00b	Value indicates if and where the ST Table is located. ST table is not available in the FM10000.
Reserved	15:11	RSV	0x0	Reserved.
StTableSize	31:16	RO	0x0	Sets to 0b. Not used.

11.29.2.73 PCIE_CFG_TPH_CTRL

Address: 0x120000 + 0x74 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
StModeSelect	2:0	RW	000b	Indicates the ST mode of operation selected. Defined encodings are: 000b = No Table Mode 001b = Interrupt Vector Mode (not supported by the FM10000) 010b = Device Specific Mode (not supported by the FM10000) All other values are reserved. Only a value of 000b is supported to indicate No Table mode of operation.
Reserved	7:3	RSV	0x0	Reserved.
TPH_ReqEn	8	RW	0b	Controls the ability to issue Request TLPs using TPH. Defined encodings are: 0b = The FM10000 is not permitted to issue transactions with TPH as Requester. 1b = The FM10000 is permitted to issue transactions with TPH as Requester. The default value of this field is 0b.
Reserved	31:9	RSV	0x0	Reserved.

11.29.2.74 PCIE_CFG_ACS_HDR

This capability is not exposed by default, unless TPH next pointer is changed by NVM setting.

Address: 0x120000 + 0x76 **Atomicity:** 32 **Reset Domains:** DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM				
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ID	15:0	RO	0x0D	PCIe Extended Capability ID. PCIe extended capability ID for the SR-IOV capability.
Version	19:16	RO	0x1	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 0x1 for this version of the specification.
Nextpointer	31:20	RO	0x0	Next Capability Offset. This field contains the offset to the next PCIe extended capability structure or 0x000 if no other items exist in the linked list of capabilities.

11.29.2.75 PCIE_CFG_ACS_CAP

Address:	**0x120000 + 0x77**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SourceValidation	0	RO	0b	The FM10000 does not support.
TranslationBlocking	1	RO	0b	The FM10000 does not support.
P2PRequestRedirect	2	RO	0b	The FM10000 does not support.
P2PCompletionRedirect	3	RO	0b	The FM10000 does not support.
UpstreamForwarding	4	RO	0b	The FM10000 does not support.
P2PEgressControl	5	RO	0b	The FM10000 does not support.
DirectTranslatedP2P	6	RO	0b	The FM10000 does not support.
Reserved	7	RSV	0b	Reserved.
EgressControlVectorSize	15:8	RO	0x0	The FM10000 does not support.
SourceValidationEnable	16	RO	0b	Must be hardwired to 0b since the ACS Source Validation functionality is not supported.
TranslationBlockingEnable	17	RO	0b	The FM10000 does not support.
P2PRequestRedirectEnable	18	RO	0b	The FM10000 does not support.
P2PCompletionRedirectEnable	19	RO	0b	The FM10000 does not support.
UpstreamForwardingEnable	20	RO	0b	The FM10000 does not support.
P2PEgressControlEnable	21	RO	0b	The FM10000 does not support.
DirectTranslatedP2PEnable	22	RO	0b	The FM10000 does not support.
Reserved	31:23	RSV	0x0	Reserved.

11.29.2.76 PCIE_PORTLOGIC

Address:	**0x120000 + 0x1C0**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
portlogic	31:0	RW	0x0	This is the base address of the debug and test registers of the link layer. Refer to the link layer documentation for further details. They are visible and accessible only via BAR4.

11.29.2.77 PCIE_PORTLOGIC_LINK_STATE

State of selected internal link layer signals, for debugging purposes.

Address:	**0x120000 + 0x1CA**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
xmlh_ltssm_state	5:0	vRO	0x0	LTSSM current state. 000000b = DETECT_QUIET 000001b = DETECT_ACT 000010b = POLL_ACTIVE 000011b = POLL_COMPLIANCE 000100b = POLL_CONFIG 000101b = PRE_DETECT_QUIET 000110b = DETECT_WAIT 000111b = CFG_LINKWD_START 001000b = CFG_LINKWD_ACEPT 001001b = CFG_LANENUM_WAIT 001010b = CFG_LANENUM_ACEPT 001011b = CFG_COMPLETE 001100b = CFG_IDLE 001101b = RCVRY_LOCK 001110b = RCVRY_SPEED 001111b = RCVRY_RCVRCFG 010000b = RCVRY_IDLE 010001b = L0 010010b = L0S 010011b = L123_SEND_EIDLE 010100b = L1_IDLE 010101b = L2_IDLE 010110b = L2_WAKE 010111b = DISABLED_ENTRY 011000b = DISABLED_IDLE 011001b = DISABLED 011010b = LPBK_ENTRY 011011b = LPBK_ACTIVE 011100b = LPBK_EXIT 011101b = LPBK_EXIT_TIMEOUT 011110b = HOT_RESET_ENTRY 011111b = HOT_RESET 100000b = RCVRY_EQ0 100001b = RCVRY_EQ1 100010b = RCVRY_EQ2 100011b = RCVRY_EQ3
mac_phy_txdatak	7:6	vRO	0x0	PIPE transmit K indication.
mac_phy_txdata	23:8	vRO	0x0	PIPE transmit data.
rmlh_rcvd_idle1	24	vRO	0x0	Receiver is receiving logical idle. 2n symbol is also idle (16-bit PHY interface only).
rmlh_rcvd_idle0	25	vRO	0x0	Receiver is receiving logical idle.
rmlh_ts_link_num_is_k237	26	vRO	0x0	Currently receiving k237 (PAD) in place of link number.
rmlh_ts_lane_num_is_k237	27	vRO	0x0	Currently receiving k237 (PAD) in place of lane number.
rmlh_ts_link_ctrl	31:28	vRO	0x0	Link control bits advertised by link partner.
rmlh_rcvd_lane_rev	32	vRO	0x0	Receiver detected lane reversal.
rmlh_ts2_rcvd	33	vRO	0x0	TS2 training sequence received (pulse).

Address: 0x120000 + 0x1CA
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
rmlh_ts1_rcvd	34	vRO	0x0	TS1 training sequence received (pulse).
rmlh_inskip_rcv	35	vRO	0x0	Receiver reports skip reception.
xmlh_link_up	36	vRO	0x0	LTSSM reports PHY link up.
xmtbyte_skip_sent	37	vRO	0x0	A skip ordered set has been transmitted.
Reserved	39:38	RSV	0x0	Reserved.
rmlh_ts_link_num	47:40	vRO	0x0	Link number advertised/confirmed by link partner.
Reserved	50:48	RSV	0x0	Reserved.
app_init_rst	51	vRO	0x0	Application request to initiate training reset.
mac_phy_txcompliance	52	vRO	0x0	PIPE transmit compliance request.
mac_phy_txelecidle	53	vRO	0x0	PIPE transmit electrical idle request.
mac_phy_txdetectrx_loopback	54	vRO	0x0	PIPE receiver detect/loopback request.
Reserved	58:55	RSV	0x0	Reserved.
xmlh_training_rst_n	59	vRO	0x0	LTSSM-negotiated link reset.
xmlh_rcvr_revrs_pol_en	60	vRO	0x0	LTSSM testing for polarity reversal.
xmlh_link_in_training	61	vRO	0x0	LTSSM performing link training.
xmlh_link_disable	62	vRO	0x0	LTSSM in DISABLE state. Link inoperable.
xmlh_scrambler_disable	63	vRO	0x0	Scrambling disabled for the link.

11.30 PCIE_CFG_VF Registers Description

The config registers for the virtual functions should be accessed via PCI Configuration Cycles using the lower 8 bits of the their address (0x000-0x3FF in byte address, 0x000-0x0FF in word address).

The same configuration registers are also visible in the global address range and are all R/W in this case. Default values are loaded via serial EEPROM at boot time.

11.30.1 PCIE_CFG_VF Map

Table 11-42 PCI Express Configuration Registers - Virtual Function Map

Name	Address (Base = 0x130000)	Atomicity	Brief Description
PCIE_VF_CFG_ID[0..63]	Base + 0x400*i + 0x0	32	
PCIE_VF_CFG_CMD[0..63]	Base + 0x400*i + 0x1	32	
PCIE_VF_CFG_1[0..63]	Base + 0x400*i + 0x2	32	
PCIE_VF_CFG_2[0..63]	Base + 0x400*i + 0x3	32	
PCIE_VF_CFG_BAR0[0..63]	Base + 0x400*i + 0x4	32	
PCIE_VF_CFG_BAR1[0..63]	Base + 0x400*i + 0x5	32	
PCIE_VF_CFG_BAR2[0..63]	Base + 0x400*i + 0x6	32	
PCIE_VF_CFG_BAR3[0..63]	Base + 0x400*i + 0x7	32	
PCIE_VF_CFG_BAR4[0..63]	Base + 0x400*i + 0x8	32	
PCIE_VF_CFG_BAR5[0..63]	Base + 0x400*i + 0x9	32	
PCIE_VF_CFG_CARDBUS[0..63]	Base + 0x400*i + 0xA	32	
PCIE_VF_CFG_SUBID[0..63]	Base + 0x400*i + 0xB	32	
PCIE_VF_CFG_EXP_ROM[0..63]	Base + 0x400*i + 0xC	32	
PCIE_VF_CFG_CAP_PTR[0..63]	Base + 0x400*i + 0xD	32	
PCIE_VF_CFG_RSVD[0..63]	Base + 0x400*i + 0xE	32	
PCIE_VF_CFG_INT[0..63]	Base + 0x400*i + 0xF	32	
PCIE_VF_CFG_PCIE_CAP[0..63]	Base + 0x400*i + 0x1C	32	PCI Express Capability.
PCIE_VF_CFG_PCIE_DEV_CAP[0..63]	Base + 0x400*i + 0x1D	32	
PCIE_VF_CFG_PCIE_DEV_CTRL[0..63]	Base + 0x400*i + 0x1E	32	
PCIE_VF_CFG_PCIE_LINK_CAP[0..63]	Base + 0x400*i + 0x1F	32	
PCIE_VF_CFG_PCIE_LINK_CTRL[0..63]	Base + 0x400*i + 0x20	32	
PCIE_VF_CFG_PCIE_DEV_CAP2[0..63]	Base + 0x400*i + 0x25	32	
PCIE_VF_CFG_PCIE_DEV_CTRL2[0..63]	Base + 0x400*i + 0x26	32	
PCIE_VF_CFG_PCIE_LINK_CTRL2[0..63]	Base + 0x400*i + 0x28	32	
PCIE_VF_CFG_MSIX_CAP[0..63]	Base + 0x400*i + 0x2C	32	MSI-X Capability.
PCIE_VF_CFG_MSIX_TOFF[0..63]	Base + 0x400*i + 0x2D	32	MSI-X Table Offset Register.
PCIE_VF_CFG_MSIX_PBA[0..63]	Base + 0x400*i + 0x2E	32	MSI-X Pending Bit Array — PBA Offset.
PCIE_VF_CFG_AER_HDR[0..63]	Base + 0x400*i + 0x40	32	

Table 11-42 PCI Express Configuration Registers - Virtual Function Map [Continued]

Name	Address (Base = 0x130000)	Atomicity	Brief Description
PCIE_VF_CFG_AER_UNERR_STATUS[0..63]	Base + 0x400*i + 0x41	32	
PCIE_VF_CFG_AER_UNERR_MASK[0..63]	Base + 0x400*i + 0x42	32	
PCIE_VF_CFG_AER_UNERR_SEVERITY[0..63]	Base + 0x400*i + 0x43	32	
PCIE_VF_CFG_AER_COERR_STATUS[0..63]	Base + 0x400*i + 0x44	32	
PCIE_VF_CFG_AER_COERR_MASK[0..63]	Base + 0x400*i + 0x45	32	
PCIE_VF_CFG_AER_CTRL[0..63]	Base + 0x400*i + 0x46	32	
PCIE_VF_CFG_AER_HEADER_LOG0[0..63]	Base + 0x400*i + 0x47	32	
PCIE_VF_CFG_AER_HEADER_LOG1[0..63]	Base + 0x400*i + 0x48	32	
PCIE_VF_CFG_AER_HEADER_LOG2[0..63]	Base + 0x400*i + 0x49	32	
PCIE_VF_CFG_AER_HEADER_LOG3[0..63]	Base + 0x400*i + 0x4A	32	
PCIE_VF_CFG_ARI_CAP[0..63]	Base + 0x400*i + 0x52	32	
PCIE_VF_CFG_ARI_CTRL[0..63]	Base + 0x400*i + 0x53	32	
PCIE_VF_CFG_TPH_HDR[0..63]	Base + 0x400*i + 0x56	32	
PCIE_VF_CFG_TPH_CAP[0..63]	Base + 0x400*i + 0x57	32	
PCIE_VF_CFG_TPH_CTRL[0..63]	Base + 0x400*i + 0x58	32	
PCIE_VF_CFG_ACS_HDR[0..63]	Base + 0x400*i + 0x76	32	
PCIE_VF_CFG_ACS_CAP[0..63]	Base + 0x400*i + 0x77	32	

11.30.2 PCIE_CFG_VF Registers

11.30.2.1 PCIE_VF_CFG_ID[0..63]

Read from the PF config space.

Address: 0x130000 + 0x400*i + 0x0
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
VendorID	15:0	RO	0xFFFF	
DeviceID	31:16	RO	0xFFFF	

11.30.2.2 PCIE_VF_CFG_CMD[0..63]

Address: 0x130000 + 0x400*i + 0x1
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
IOSpaceEn	0	RO	0b	I/O Access Enable. Not used in this design. Always set to zero.
MemSpaceEn	1	RO	0b	Memory Access Enable.

Address:	**0x130000 + 0x400*i + 0x1**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
BusMasterEn	2	RW	0b	Enable Mastering (Also named Bus Master Enable (BME).)
SpecialCycEn	3	RO	0b	Special Cycle Monitoring (Hardwired to 0b.)
MemWrInv	4	RO	0b	MWI Enable (Hardwired to 0b.)
VGAPaletteSnp	5	RO	0b	Palette Snoop Enable (Hardwired to 0b.)
ParErrResp	6	RO	0b	Parity Error Response.
IdselStep	7	RO	0b	Wait Cycle Enable (Hardwired to 0b.)
SERRnEn	8	RO	0b	SERR# Enable.
FastB2BEn	9	RO	0b	Fast Back-to-Back Enable (Hardwired to 0b.)
INTAssertDis	10	RO	0b	Interrupt Disable When set, devices are prevented from generating legacy interrupt messages.
Reserved	18:11	RSV	0x0	Reserved.
INTStat	19	RO	0b	Interrupt Status.
CapList	20	RO	1b	New Capabilities. Indicates that this device implements extended capabilities. It supports: • PCI Power Management • Message Signaled Interrupts (MSI) • Enhanced Message Signaled Interrupts (MSI-X) • VPD • The PCIe extensions.
Cap66Mhz	21	RO	0b	66 MHz Capable (Hard wire to 0b.)
Reserved	22	RSV	0b	Reserved.
CapFastB2B	23	RO	0b	Fast Back-to-Back Capable (Hardwired to 0b.)
MstrDataParErr	24	CW1	0b	Data Parity Reported.
DevselTiming	26:25	RO	00b	DEVSEL Timing (Hardwired to 0b.)
SigTgtAbort	27	CW1	0b	Signaled Target Abort.
RcvTgtAbort	28	CW1	0b	Received Target Abort.
RcvMstAbort	29	CW1	0b	Received Master Abort.
SigSysErr	30	CW1	0b	Signaled System Error.
DetectParErr	31	CW1	0b	Detected Parity Error.

11.30.2.3 PCIE_VF_CFG_1[0..63]

Address:	**0x130000 + 0x400*i + 0x2**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
RevisionID	7:0	RO	0x0	Set to 0b for first version.
ClassCode	31:8	RO	0x020000	The class code is a read-only value that identifies the device functionality, it is set to 0x020000 (Ethernet Adapter) by default.

11.30.2.4 PCIE_VF_CFG_2[0..63]

Address: 0x130000 + 0x400*i + 0x3
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CacheLineSize	7:0	RW	0x0	This field is implemented by PCIe devices as a read/write field for legacy compatibility purposes but has no impact on any PCIe device functionality.
LatencyTimer	15:8	RO	0x0	Not used.
HeaderType	23:16	RO	0x0	Single function. Set to 0b.
BIST	31:24	RO	0x0	

11.30.2.5 PCIE_VF_CFG_BAR0[0..63]

Address: 0x130000 + 0x400*i + 0x4
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	00b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.
Prefetchable	3	RO	0b	The PEP address space does not contain volatile registers.
Zero	20:4	RO	0x0	Bits [20:4] of base address assigned to this device. 2 MB Region.
Address	31:21	RO	0x0	Bits [31:21] of base address assigned to this device.

11.30.2.6 PCIE_VF_CFG_BAR1[0..63]

Address: 0x130000 + 0x400*i + 0x5
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RO	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.30.2.7 PCIE_VF_CFG_BAR2[0..63]

Address: 0x130000 + 0x400*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	00b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.

Address: 0x130000 + 0x400*i + 0x6
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Prefetchable	3	RO	0b	The PEP address space does not contain volatile registers.
Zero	12:4	RO	0x0	Bits [12:4] of base address assigned to this device. 8 KB Region.
Address	31:13	RO	0x0	Bits [31:13] of base address assigned to this device.

11.30.2.8 PCIE_VF_CFG_BAR3[0..63]

Address: 0x130000 + 0x400*i + 0x7
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RO	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.30.2.9 PCIE_VF_CFG_BAR4[0..63]

Address: 0x130000 + 0x400*i + 0x8
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AccessType	0	RO	0b	Only type 0b (MEMORY) is supported.
AddressSize	2:1	RO	00b	Address size on address decoding. 00b = Mode32 10b = Mode64 All other values are reserved.
Prefetchable	3	RO	0b	The PEP address space does not contain volatile registers.
Zero	25:4	RO	0x0	Bits [25:4] of base address assigned to this device. 64 MB Region.
Address	31:26	RO	0x0	Bits [31:26] of base address assigned to this device.

11.30.2.10 PCIE_VF_CFG_BAR5[0..63]

Address: 0x130000 + 0x400*i + 0x9
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
AddressHigh	31:0	RO	0x0	Bits [63:32] of base address assigned to this device. Only used if 64-bit addressing is active.

11.30.2.11 PCIE_VF_CFG_CARDBUS[0..63]

Address: **0x130000 + 0x400*i + 0xA**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CardbusPtr	31:0	RO	0x0	Not used.

11.30.2.12 PCIE_VF_CFG_SUBID[0..63]

Address: **0x130000 + 0x400*i + 0xB**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
SubVendorID	15:0	RO	0x8086	This value can be loaded automatically from the EEPROM at power up or reset. A value of 0x8086 is the default for this field at power up if the EEPROM does not respond or is not programmed. All functions are initialized to the same value.
SubDeviceID	31:16	RO	0x0	This value can be loaded automatically from the EEPROM at power up with a default value of 0x0000.

11.30.2.13 PCIE_VF_CFG_EXP_ROM[0..63]

Address: **0x130000 + 0x400*i + 0xC**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Enable	0	RO	0b	Not used.
Reserved	10:1	RSV	0x0	Reserved.
Address	31:11	RO	0x0	Not used.

11.30.2.14 PCIE_VF_CFG_CAP_PTR[0..63]

Address: **0x130000 + 0x400*i + 0xD**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CapabilityPtr	7:0	RO	0x70	The Capabilities Pointer field (Cap_Ptr) is an 8-bit field that provides an offset in the PCI configuration space for the location of the first item in the capabilities linked list. The value is 0x70, which is the address of the first entry: PCI capabilities.
Reserved	31:8	RSV	0x0	Reserved.

11.30.2.15 PCIE_VF_CFG_RSVD[0..63]

Address: 0x130000 + 0x400*i + 0xE
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	31:0	RSV	0x0	Reserved.

11.30.2.16 PCIE_VF_CFG_INT[0..63]

Address: 0x130000 + 0x400*i + 0xF
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
InterruptLine	7:0	RO	0x0	Read/write register programmed by software to indicate which of the system interrupt request lines the device interrupt pin is bound to. Refer to the PCI definition for more details. Each PCI function has its own register.
InterruptPin	15:9	RW	0x0	Not used in this design.
MinGrant	23:16	RW	0x0	Min_Gnt not used. Hardwired to 0b.
MaxLatency	31:24	RW	0x0	Max_Lat not used. Hardwired to 0b.

11.30.2.17 PCIE_VF_CFG_PCIE_CAP[0..63]

Address: 0x130000 + 0x400*i + 0x1C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapID	7:0	RO	0x10	This field equals 0x10 indicating the linked list item as being the PCIe Capabilities registers.
NextCapPtr	15:8	RO	0xB0	Offset to the next capability item in the capability list.
CapabilityVersion	19:16	RO	0x2	Indicates the PCIe capability structure version. The device supports PCIe version 2 (also loaded from EEPROM).
DevicePortType	23:20	RO	0x0	Indicates the type of PCIe functions. All functions are native PCI functions with a value of 0000b.
SlotImplemented	24	RO	0b	The device does not implement slot options. Therefore, this field is hardwired to 0b.
InterrruptMessageNumber	29:25	RO	0x0	The device does not implement multiple MSI per function. Therefore, this field is hardwired to 0b.
Reserved	31:30	RSV	00b	Reserved.

11.30.2.18 PCIE_VF_CFG_PCIE_DEV_CAP[0..63]

Address: 0x130000 + 0x400*i + 0x1D
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
MaxPayloadSize	2:0	RO	010b	Max Payload Size Supported. This field indicates the maximum payload the device can support for TLPs. It is loaded from the EEPROM with a default value of 512 bytes.
PhantomFunctionSupport	4:3	RO	00b	Not supported by the device.
ExtendedTagField	5	RO	0b	Maximum supported size of the *Tag* field. The device supports a 5-bit *Tag* field for all functions.
EndpointL0Latency	8:6	RO	011b	This field indicates the acceptable latency the device can withstand due to the transition from L0s state to the L0 state. Value loaded from EEPROM. A value of 011b equals 512 ns.
EndpointL1Latency	11:9	RO	110b	This field indicates the acceptable latency the device can withstand due to the transition from L1 state to the L0 state. A value of 110b equals 32-64 µs.
AttentionButtonPreset	12	RO	0b	Hardwired in the device to 0b.
AttentionIndicatorPreset	13	RO	0b	Hardwired in the device to 0b.
PowerIndicatorPreset	14	RO	0b	Hardwired in the device to 0b.
RoleBasedErrorReporting	15	RO	1b	Role Based Error Reporting. Hard wired in the device to 1b.
Reserved	17:16	RSV	00b	Reserved.
SlotPowerLimit	25:18	RO	0x0	Used in upstream ports only. This value is set by the Set_Slot_Power_Limit Message.
SlotPowerScale	27:26	RO	00b	Slot Power Limit Scale. Used in upstream ports only. This value is set by the Set_Slot_Power_Limit Message.
FunctionLevelReset	28	RO	1b	A value of 1b indicates the Function supports the optional Function Level Reset (FLR) mechanism.
Reserved	31:29	RSV	000b	Reserved.

11.30.2.19 PCIE_VF_CFG_PCIE_DEV_CTRL[0..63]

Address: 0x130000 + 0x400*i + 0x1E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CorrectableErrorReportingEnable	0	RO	0b	Enable error report.
NonFatalErrorReportingEnable	1	RO	0b	Enable error report.
FatalErrorReportingEnable	2	RO	0b	Enable error report.
UnsupportedRequestReportingEnable	3	RO	0b	Enable error report.
EnableRelaxedOrdering	4	RO	1b	If this bit is set, the FM10000 is permitted to set the Relaxed Ordering bit in the *Attribute* field of write transactions that do not need strong ordering. Refer to the PCIE_CTRL_EXT register bit *RO_DIS* for more details.

Address: 0x130000 + 0x400*i + 0x1E
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
MaxPayloadSize	7:5	RO	000b	This field sets the maximum TLP payload size for FM10000 functions. As a receiver, the FM10000 must handle TLPs as large as the set value. As a transmitter, the FM10000 must not generate TLPs exceeding the set value. The Max Payload Size field supported in the Device Capabilities register indicates permissible values that can be programmed. In ARI mode, Max Payload Size is determined solely by the field in function 0, while it is meaningless in the other function(s).
ExtendedTagfieldEnable	8	RO	0b	Not implemented in the FM10000.
PhantomFunctionsEnable	9	RO	0b	Not implemented in the FM10000.
AuxiliaryPowerPMEnable	10	RO	0b	No AUX power support in the FM10000.
EnableNoSnoop	11	RO	1b	When set, the Function is permitted to set the No Snoop bit in the Requester Attributes of transactions it initiates that do not require hardware enforced cache coherency.
MaxReadRequestSize	14:12	RO	010b	This field sets maximum read request size for the FM10000 as a requester. 000b = MAX_128 001b = MAX_256 010b = MAX_512 011b = MAX_1024 100b = MAX_2048 101b = MAX_4096 110b = Reserved 111b = Reserved
InitiateFLR	15	RW	0b	A write of 1b initiates FLR to the function. The value read by software from this bit is always 0b.
CorrectableDetected	16	CW1	0b	Indicates status of correctable error detection.
NonFatalErrorDetected	17	CW1	0b	Indicates status of non-fatal error detection.
FatalErrorDetected	18	CW1	0b	Indicates status of fatal error detection.
UnsupportedRequestDetected	19	CW1	0b	Indicates that the FM10000 received an unsupported request. This field is identical in all functions. The FM10000 cannot distinguish which function causes the error.
AuxPowerDetected	20	RO	0b	If Aux Power is detected, this field is set to 1b. It is a strapping signal from the periphery and is identical for all functions. Resets on LAN Power Good and PE_RST_N only.
TransactionPending	21	RO	0b	Indicates whether the FM10000 has ANY transactions pending. Transactions include completions for any outstanding non-posted request for all used traffic classes.
Reserved	31:22	RSV	0x0	Reserved.

11.30.2.20 PCIE_VF_CFG_PCIE_LINK_CAP[0..63]

Address:	**0x130000 + 0x400*i + 0x1F**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
SupportedLinkSpeeds	3:0	RO	0x3	Supported Link Speeds. This field indicates the supported Link speed(s) of the associated link port. Defined encodings are: 0001b = 2.5 GbE link speed supported. 0010b = 5 GbE and 2.5 GbE link speeds supported. 0011b = 8 GbE, 5 GbE and 2.5 GbE link speeds supported. All other values are reserved.
MaxLinkWidth	9:4	RO	0x8	Max Link Width. Indicates the maximum link width. Depending on the SKU, The FM10000 supports a x1, x2, x4 and x8-link width with a default value of eight lanes. Defined encodings are: 000001b = x1 000010b = x2 000100b = x4 001000b = x8 All other values are reserved.
ActiveStateLinkPMSupport	11:10	RO	10b	Indicates the level of the active state of power management supported in the FM10000. Defined encodings are: 00b = No ASPM Support. 01b = Reserved. 10b = L1 supported. This is the default. 11b = Reserved. All functions share the same value loaded from the EEPROM.
L0sExitLatency	14:12	RO	011b	L0s Exit Latency. Indicates the exit latency from L0s to L0 state. 000b = Less than 64 ns 001b = 64 ns - 128 ns 010b = 128ns - 256 ns 011b = 256 ns - 512 ns 100b = 512 ns - 1 μs 101b = 1 μs - 2 μs 110b = 2 μs - 4 μs 111b = Reserved All functions share the same value loaded from the EEPROM.
L1ExitLatency	17:15	RO	110b	L1 Exit Latency. Indicates the exit latency from L1 to L0 state. 000b = Less than 1 μs 001b = 1 μs - 2 μs 010b = 2 μs - 4 μs 011b = 4 μs - 8 μs 100b = 8 μs - 16 μs 101b = 16 μs - 32 μs 110b = 32 μs - 64 μs 111b = L1 transition not supported. All functions share the same value loaded from the EEPROM.

Address:	**0x130000 + 0x400*i + 0x1F**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ClockPowerManagement	18	RO	0b	Hardwired to 0b.
SurpriseDownErrorReportingCapable	19	RO	0b	Hardwired to 0b.
DataLinkLayerLinkActiveReportingCapable	20	RO	0b	Hardwired to 0b.
LinkBandwidthNotificationCapability	21	RO	0b	Hardwired to 0b.
Reserved	23:22	RSV	00b	Reserved.
PortNumber	31:24	RO	0x0	The PCIe port number for the given PCIe link. This field is set in the link training phase.

11.30.2.21 PCIE_VF_CFG_PCIE_LINK_CTRL[0..63]

Address:	**0x130000 + 0x400*i + 0x20**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RsvdZ	31:0	RO	0x0	

11.30.2.22 PCIE_VF_CFG_PCIE_DEV_CAP2[0..63]

Address:	**0x130000 + 0x400*i + 0x25**			
Atomicity:	**32**			
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CplTimeoutCfg	3:0	RO	0xF	Completion Timeout Ranges Supported. This field indicates FM10000 support for the optional completion timeout programmability mechanism. Four time value ranges are defined: • Range A: 50 µs to 10 ms. • Range B: 10 ms to 250 ms. • Range C: 250 ms to 4 s. • Range D: 4 s to 64 s. Bits are set according to the following values to show the timeout value ranges that the FM10000 supports. 0000b = Completion timeout programming not supported. The FM10000 must implement a timeout value in the range of 50 µs to 50 ms. 0001b = Range A. 0010b = Range B. 0011b = Ranges A and B. 0110b = Ranges B and C. 0111b = Ranges A, B and C. 1110b = Ranges B, C and D. 1111b = Ranges A, B, C and D. All other encodings are reserved.
CplTimeoutDisSuppt	4	RO	1b	Completion Timeout Disable Supported.
Reserved	11:5	RSV	0x0	Reserved

Address:	**0x130000 + 0x400*i + 0x25**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
TPHCplSup	13:12	RO	01b	TPH Completer Supported. Value indicates Completer support for TPH or Extended TPH. Defined encodings are: 00b = TPH and Extended TPH Completer not supported. 01b = TPH Completer supported; Extended TPH Completer not supported. 10b = Reserved. 11b = Both TPH and Extended TPH Completer supported. When TPH is enabled, the device is enabled as both a requester and a completer. No control is given to support these features individually.
Reserved	31:14	RSV	0x0	Reserved.

11.30.2.23 PCIE_VF_CFG_PCIE_DEV_CTRL2[0..63]

Address:	**0x130000 + 0x400*i + 0x26**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
CplTimeoutVal	3:0	RW	0x0	Completion Timeout Value. For devices that support completion timeout programmability, this field enables system software to modify the completion timeout value. Defined encodings are: 0000b = Default range: 50 µs to 50 ms ***Note:*** It is strongly recommended that the completion timeout mechanism not expire in less than 10 ms. Values available if Range A (50 µs to 10 ms) programmability range is supported: 0001b = 50 µs to 100 µs 0010b = 1 ms to 10 ms Values available if Range B (10 ms to 250 ms) programmability range is supported: 0101b = 16 ms to 55 ms 0110b = 65 ms to 210 ms Values available if Range C (250 ms to 4 s) programmability range is supported: 1001b = 260 ms to 900 ms 1010b = 1 s to 3.5 s Values available if the Range D (4 s to 64 s) programmability range is supported: 1101b = 4 s to 13 s 1110b = 17 s to 64 s Values not defined are reserved. Software is permitted to change the value of this field at any time. For requests already pending when the completion timeout value is changed, hardware is permitted to use either the new or the old value for the outstanding requests and is permitted to base the start time for each request either on when this value was changed or on when each request was issued.
CplTimeoutDis	4	RW	0b	Completion Timeout Disable. When set to 1b, this bit disables the completion timeout mechanism. Software is permitted to set or clear this bit at any time. When set, the completion timeout detection mechanism is disabled. If there are outstanding requests when the bit is cleared, it is permitted but not required for hardware to apply the completion timeout mechanism to the outstanding requests. If this is done, it is permitted to base the start time for each request on either the time this bit was cleared or the time each request was issued.
Reserved	31:5	RSV	0x0	Reserved.

11.30.2.24 PCIE_VF_CFG_PCIE_LINK_CTRL2[0..63]

Address: 0x130000 + 0x400*i + 0x28
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
TargetSpeed	3:0	SRW	0x0	Target Link Speed. This field is used to set the target compliance mode speed when software is using the Enter Compliance bit to force a link into compliance mode. Defined encodings are: 0001b = 2.5 GbE target link speed. 0010b = 5 GbE target link speed. 0011b = 8 GbE target link speed. All other values are reserved. If a value is written to this field that does not correspond to a speed included in the Supported Link Speeds field, the result is undefined. The default value of this field is the highest link speed supported by the FM10000 (as reported in the Supported Link Speeds field of the Link Capabilities register).
EnterCompliance	4	SRW	0b	Enter Compliance. Software is permitted to force a link to enter compliance mode at the speed indicated in the Target Link Speed field by setting this bit to 1b in both components on a link and then initiating a hot reset on the link. The default value of this field following a fundamental reset is 0b.
HwAutoSpeedDis	5	SRW	0b	Hardware Autonomous Speed Disable When set to 1b, this bit disables hardware from changing the link speed for reasons other than attempting to correct unreliable link operation by reducing link speed.
SelectDeEmphasis	6	RO	0b	Selectable De-Emphasis.
TransmitMargin	9:7	SRW	000b	Transmit Margin. This field controls the value of the non- de-emphasized voltage level at the Transmitter pins. Defined encodings are: 000b = Normal operating range. 001b = 800-1200 mV for full swing and 400-700 mV for half-swing. 010b = (n-1) — Values must be monotonic with a non-zero slope. The value of n must be greater than 3 and less than 7. At least two of these must be below the normal operating range of n: 200-400 mV for full-swing and 100-200 mV for half-swing. 111b = (n) — Reserved. All other values are reserved.
EnterModeComp	10	SRW	0b	Enter Modified Compliance. When this bit is set to 1b, the device transmits modified compliance pattern if the LTSSM enters Polling.Compliance state.
CompSOS	11	SRW	0b	Compliance SOS. When set to 1b, the LTSSM is required to send SOS periodically in between the (modified) compliance patterns.
Reserved	15:12	RSV	0x0	Reserved.
CAPCURRDEEMPHASIS	16	RO	1b	Current De-emphasis Level. When the Link is operating at 5.0 GT/s speed, this bit reflects the level of de-emphasis. Encodings are: 0b = -6 dB 1b = 3.5 dB The value in this bit is undefined when the Link is not operating at 5.0 GT/s speed. The default value of this field is changed automatically by hardware upon link setup.
Reserved	31:17	RSV	0x0	Reserved.

11.30.2.25 PCIE_VF_CFG_MSIX_CAP[0..63]

More than one MSI-X capability structure per function is prohibited while a function is permitted to have both an MSI and an MSI-X capability structure.

In contrast to the MSI capability structure, which directly contains all of the control/status information for the function's vectors, the MSI-X capability structure instead points to an MSI-X table structure and an MSI-X Pending Bit Array (PBA) structure, each residing in memory space.

Each structure is mapped by a BAR belonging to the function that begins at 0x10 in the configuration space. A BAR Indicator Register (BIR) indicates which BAR and a Qword-aligned offset indicates where the structure begins relative to the base address associated with the BAR.The BAR is 64-bit, but must map to the memory space. A function is permitted to map both structures with the same BAR or map each structure with a different BAR.

The MSI-X table structure typically contains multiple entries, each consisting of several fields: Message Address, Message Upper Address, Message Data, and Vector Control. Each entry is capable of specifying a unique vector.

The PBA structure contains the function's pending bits, one per table entry, organized as a packed array of bits within Qwords. The last Qword is not necessarily fully populated.

To request service using a given MSI-X table entry, a function performs a Dword memory write transaction using:

- The contents of the Message Data field entry for data
- The contents of the Message Upper Address field for the upper 32 bits of the address
- The contents of the Message Address field entry for the lower 32 bits of the address

A memory read transaction from the address targeted by the MSI-X message produces undefined results.

The MSI-X table and MSI-X PBA are permitted to co-reside within a naturally aligned 4 KB address range, though they must not overlap with each other.

MSI-X table entries and Pending bits are each numbered 0 through N-1, where N-1 is indicated by the Table Size field in the MSI-X Message Control register. For a given arbitrary MSI-X table entry K, its starting address can be calculated with the formula:

Entry starting address = Table base + K*16

For the associated *Pending* bit K, its address for Qword access and bit number within that Qword can be calculated with the formulas:

Qword address = PBA base + (K div 64)*8

Qword bit# = K mod 64

Software that chooses to read *Pending* bit K with Dword accesses can use these formulas:

Dword address = PBA base + (K div 32)*4

Dword bit# = K mod 32

Address: 0x130000 + 0x400*i + 0x2C
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapID	7:0	RO	0x11	Capability ID.
NextCapPtr	15:8	RO	0x00	Next capability offset.

Address: **0x130000 + 0x400*i + 0x2C**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
TableSize	26:16	RO	0x07	System software reads this field to determine the MSI-X Table Size N, which is encoded as N-1. The device supports up to 32 different interrupt vectors per virtual function.
Reserved	29:27	RSV	000b	Reserved.
FunctionMask	30	RO	0b	Function Mask. 0b = Each vector's Mask bit determines whether the vector is masked or not. 1b = All of the vectors associated with the function are masked, regardless of their per-vector Mask bit states. Setting or clearing the MSI-X Function Mask bit has no effect on the state of the per-vector Mask bits.
MSIX_Enable	31	RW	0b	MSIX Enable. 0b = The function is prohibited from using MSI-X to request service. 1b = If set, and the MSI Enable bit in the MSI Message Control register is 0b, the function is permitted to use MSI-X to request service and is prohibited from using its INTx# pin. System configuration software sets this bit to enable MSI-X. A device driver is prohibited from writing this bit to mask a function's service request.

11.30.2.26 PCIE_VF_CFG_MSIX_TOFF[0..63]

Address: **0x130000 + 0x400*i + 0x2D**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
BIR	2:0	RO	010b	Indicates which one of a functions BARs, beginning at 0x10 in the configuration space, is used to map the functions MSI-X table into the memory space. BIR values 0...5 correspond to BARs 0x10...0x 24 respectively.
Offset	31:3	RO	0x0	Used as an offset from the address contained in one of the functions BARs to point to the base of the MSI-X table. The lower three Table BIR bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. ***Note:*** This field is read only.

11.30.2.27 PCIE_VF_CFG_MSIX_PBA[0..63]

Address: **0x130000 + 0x400*i + 0x2E**
Atomicity: **32**
Reset Domains: **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
BIR	2:0	RO	010b	Indicates which one of a functions BARs, beginning at 0x10 in the configuration space, is used to map the functions MSI-X PBA into the memory space. BIR values 0...5 correspond to BARs 0x10...0x 24 respectively.
Offset	31:3	RO	0x0200	Used as an offset from the address contained in one of the functions BARs to point to the base of the MSI-X PBA. The lower three PBA BIR bits are masked off (set to 0b) by software to form a 32-bit Qword-aligned offset. ***Note:*** This field is read only.

11.30.2.28 PCIE_VF_CFG_AER_HDR[0..63]

Address:	**0x130000 + 0x400*i + 0x40**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
ExtCapId	15:0	RO	0x2	Extended Capability ID. PCIe extended capability ID indicating advanced error reporting capability.
Version	19:16	RO	0x1	Version Number. PCIe advanced error reporting extended capability version number.
NextCapPtr	31:20	RO	0x148	Next Capability Pointer.

11.30.2.29 PCIE_VF_CFG_AER_UNERR_STATUS[0..63]

The Uncorrectable Error Status register reports error status of individual uncorrectable error sources on a PCIe device. An individual error status bit that is set to 1b indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit. Register is cleared by LAN_PWR_GOOD.

Address:	**0x130000 + 0x400*i + 0x41**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	CW1	0b	Data Link Protocol Error Status.
SurpriseDownError	5	RO	0b	Surprise Down Error Status.
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	CW1	0b	Poisoned TLP Status.
FlowControlError	13	CW1	0b	Flow Control Protocol Error Status.
CompletionTimeout	14	CW1	0b	Completion Timeout Status.
CompleterAbort	15	CW1	0b	Completer Abort Status.
UnexpecteCompl	16	CW1	0b	Unexpected Completion Status.
RxOverflow	17	CW1	0b	Receiver Overflow Status.
MalformedTLP	18	CW1	0b	Malformed TLP Status.
ECRCError	19	CW1	0b	ECRC Error Status.
UnsupportedRequestError	20	CW1	0b	Unsupported Request Error Status.
ACSViolation	21	RO	0b	ACS Violation Status.
Reserved	31:22	RSV	0x0	Reserved.

11.30.2.30 PCIE_VF_CFG_AER_UNERR_MASK[0..63]

The Uncorrectable Error Mask register controls reporting of individual uncorrectable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. Note that there is a mask bit per bit of the Uncorrectable Error Status register.

Address: 0x130000 + 0x400*i + 0x42
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	RO	0b	Data Link Protocol Error Mask.
SurpriseDownError	5	RO	0b	Surprise Down Error Mask.
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	RO	0b	Poisoned TLP Mask.
FlowControlError	13	RO	0b	Flow Control Protocol Error Mask.
CompletionTimeout	14	RO	0b	Completion Timeout Mask.
CompleterAbort	15	RO	0b	Completer Abort Mask.
UnexpecteCompl	16	RO	0b	Unexpected Completion Mask.
RxOverflow	17	RO	0b	Receiver Overflow Mask.
MalformedTLP	18	RO	0b	Malformed TLP Mask.
ECRCError	19	RO	0b	ECRC Error Mask.
UnsupportedRequestError	20	RO	0b	Unsupported Request Error Mask.
Reserved	21	RSV	0b	Reserved.
UncorrectableInternalError	22	RO	0b	
Reserved	23	RSV	0b	Reserved.
AtomicOpEggressBlocked	24	RO	0b	
TLPPrefixBlockedError	25	RO	0b	
Reserved	31:26	RSV	0x0	Reserved.

11.30.2.31 PCIE_VF_CFG_AER_UNERR_SEVERITY[0..63]

The Uncorrectable Error Severity register controls whether an individual uncorrectable error is reported as a fatal error. An uncorrectable error is reported as fatal when the corresponding error bit in the severity register is set. If the bit is cleared, the corresponding error is considered non-fatal.

Address: 0x130000 + 0x400*i + 0x43
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	3:0	RSV	0x0	Reserved.
DLPError	4	RO	0b	Data Link Protocol Error Severity.
SurpriseDownError	5	RO	0b	Surprise Down Error Severity.
Reserved	11:6	RSV	0x0	Reserved.
PoisonedTLP	12	RO	0b	Poisoned TLP Severity.
FlowControlError	13	RO	0b	Flow Control Protocol Error Severity.
CompletionTimeout	14	RO	0b	Completion Timeout Severity.

Address: **Atomicity:** **Reset Domains:**	**0x130000 + 0x400*i + 0x43** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CompleterAbort	15	RO	0b	Completer Abort Severity.
UnexpecteCompl	16	RO	0b	Unexpected Completion Severity.
RxOverflow	17	RO	0b	Receiver Overflow Severity.
MalformedTLP	18	RO	0b	Malformed TLP Severity.
ECRCError	19	RO	0b	ECRC Error Severity.
UnsupportedRequestError	20	RO	0b	Unsupported Request Error Severity.
Reserved	21	RSV	0b	Reserved.
UncorrectableInternalError	22	RO	0b	
Reserved	23	RSV	0b	Reserved.
AtomicOpEggressBlocked	24	RO	0b	
TLPPrefixBlockedError	25	RO	0b	
Reserved	31:26	RSV	0x0	Reserved.

11.30.2.32 PCIE_VF_CFG_AER_COERR_STATUS[0..63]

The Correctable Error Status register reports error status of individual correctable error sources on a PCIe device. When an individual error status bit is set to 1b, it indicates that a particular error occurred; software can clear an error status by writing a 1b to the respective bit. Register is cleared by LAN_PWR_GOOD.

Address: **Atomicity:** **Reset Domains:**	**0x130000 + 0x400*i + 0x44** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
RxError	0	RO	0b	Receiver Error Status.
Reserved	5:1	RSV	0x0	Reserved.
BadTLP	6	RO	0b	Bad TLP Status.
BadDLLP	7	RO	0b	Bad DLLP Status.
ReplayRollover	8	RO	0b	REPLAY_NUM Rollover Status.
Reserved	11:9	RSV	000b	Reserved.
ReplayTimeout	12	RO	0b	Replay Timer Timeout Status.
NonFatalError	13	RO	0b	Advisory Non-Fatal Error Status.
CorrectableInternalError	14	RO	0b	Corrected Internal Error Status.
Reserved	31:15	RSV	0x0	Reserved.

11.30.2.33 PCIE_VF_CFG_AER_COERR_MASK[0..63]

The Correctable Error Mask register controls reporting of individual correctable errors by device to the host bridge via a PCIe error message. A masked error (respective bit set in mask register) is not reported to the host bridge by an individual device. There is a mask bit per bit in the Correctable Error Status register.

Address: 0x130000 + 0x400*i + 0x45
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
RxError	0	RO	0b	Receiver Error Status.
Reserved	5:1	RSV	0x0	Reserved.
BadTLP	6	RO	0b	Bad TLP Status.
BadDLLP	7	RO	0b	Bad DLLP Status.
ReplayRollover	8	RO	0b	REPLAY_NUM Rollover Status.
Reserved	11:9	RSV	000b	Reserved.
ReplayTimeout	12	RO	0b	Replay Timer Timeout Status.
NonFatalError	13	RO	0b	Advisory Non-Fatal Error Status.
CorrectableInternalError	14	RO	0b	Corrected Internal Error Mask.
Reserved	31:15	RSV	0x0	Reserved.

11.30.2.34 PCIE_VF_CFG_AER_CTRL[0..63]

Address: 0x130000 + 0x400*i + 0x46
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
ErrorIndex	4:0	RO	0x0	Vector pointing to the first recorded error in the Uncorrectable Error Status register. This is a read-only field that identifies the bit position of the first uncorrectable error reported in the Uncorrectable Error Status register.
ECRCGenCap	5	RO	1b	ECRC Generation Capable. If set, this bit indicates that the function is capable of generating ECRC. This bit is loaded from EEPROM.
ECRCGenEn	6	RW	0b	ECRC Generation Enable. When set, ECRC generation is enabled.
ECRCCheckCap	7	RO	1b	ECRC Check Capable. If set, this bit indicates that the function is capable of checking ECRC. This bit is loaded from EEPROM.
ECRCCheckEn	8	RW	0b	ECRC Check Enable. When set Set, ECRC checking is enabled.
Reserved	31:9	RSV	0x0	Reserved.

11.30.2.35 PCIE_VF_CFG_AER_HEADER_LOG0[0..63]

The header log register captures the header for the transaction that generated an error.

Address:	**0x130000 + 0x400*i + 0x47**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.30.2.36 PCIE_VF_CFG_AER_HEADER_LOG1[0..63]

The header log register captures the header for the transaction that generated an error.

Address:	**0x130000 + 0x400*i + 0x48**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.30.2.37 PCIE_VF_CFG_AER_HEADER_LOG2[0..63]

The header log register captures the header for the transaction that generated an error.

Address:	**0x130000 + 0x400*i + 0x49**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.30.2.38 PCIE_VF_CFG_AER_HEADER_LOG3[0..63]

The header log register captures the header for the transaction that generated an error.

Address:	**0x130000 + 0x400*i + 0x4A**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
Header	31:0	RO	0x0	Header of the packet in error (TLP or DLLP).

11.30.2.39 PCIE_VF_CFG_ARI_CAP[0..63]

In order to allow more than eight functions per endpoint without requesting an internal switch, as is usually needed in virtualization scenarios, the PCI-SIG defines a new capability that allows a different interpretation of the Bus, Device, and Function fields.

Address: 0x130000 + 0x400*i + 0x52
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapId	15:0	RO	0x000E	Capability ID.
Version	19:16	RO	0x1	Version.
NextCapPtr	31:20	RO	0x158	Next capability pointer. This field must be set to 0x000 in NVM/BSM by default to not expose the TPH Capabilities.

11.30.2.40 PCIE_VF_CFG_ARI_CTRL[0..63]

Address: 0x130000 + 0x400*i + 0x53
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
Reserved	7:0	RSV	0x0	Reserved.
NFP	15:8	RO	0x0	Next function pointer. Zero, as there is none.
Reserved	31:16	RSV	0x0	Reserved.

11.30.2.41 PCIE_VF_CFG_TPH_HDR[0..63]

Address: 0x130000 + 0x400*i + 0x56
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
CapId	15:0	RO	0x0017	Capability ID.
Version	19:16	RO	0x1	Version.
NextCapPtr	31:20	RO	0x000	Next capability pointer. This field contains the offset to the next PCIe extended capability structure, or 0x000 if no other items exist in the linked list of capabilities. The field is set to 0x1D8 by NVM/BSM to expose the ACS capability to host.

11.30.2.42 PCIE_VF_CFG_TPH_CAP[0..63]

Address: 0x130000 + 0x400*i + 0x57
Atomicity: 32
Reset Domains: DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM

Field Name	Bit(s)	Type	Default	Description
NoStMode	0	RO	1b	The FM10000 does support.
InterruptVector	1	RO	0b	The FM10000 does not support Interrupt Vector Mode of operation.

Address: **Atomicity:** **Reset Domains:**	**0x130000 + 0x400*i + 0x57** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DeviceSpecific	2	RO	0b	the FM10000 does not support Device Specific Mode of operation.
Reserved	7:3	RSV	0x0	Reserved.
ExtendedTPH	8	RO	0b	Cleared to indicate that the function is not capable of generating requests with Extended TPH TLP Prefix.
StTableLocation	10:9	RO	00b	Value indicates if and where the ST Table is located. ST table is not available in the FM10000.
Reserved	15:11	RSV	0x0	Reserved.
StTableSize	31:16	RO	0x0	Sets to 0. Not used.

11.30.2.43 PCIE_VF_CFG_TPH_CTRL[0..63]

Address: **Atomicity:** **Reset Domains:**	**0x130000 + 0x400*i + 0x58** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
StModeSelect	2:0	RW	000b	Indicates the ST mode of operation selected. Defined encodings are: 000b = No Table Mode. 001b = Interrupt Vector Mode (not supported by the FM10000). 010b = Device Specific Mode (not supported by the FM10000). All other values are reserved for future use. Only a value of 000b is supported to indicate No Table mode of operation.
Reserved	7:3	RSV	0x0	Reserved.
TPH_ReqEn	8	RW	0b	Controls the ability to issue Request TLPs using TPH. Defined encodings are: 0b = The FM10000 is not permitted to issue transactions with TPH as Requester. 1b = The FM10000 is permitted to issue transactions with TPH as Requester. The default value of this field is 0b.
Reserved	31:9	RSV	0x0	Reserved.

11.30.2.44 PCIE_VF_CFG_ACS_HDR[0..63]

This capability is not exposed by default, unless TPH next pointer is changed by NVM setting.

Address: **Atomicity:** **Reset Domains:**	**0x130000 + 0x400*i + 0x76** **32** **DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
ID	15:0	RO	0x0D	PCIe Extended Capability ID. PCIe extended capability ID for the SR-IOV capability.
Version	19:16	RO	0x1	Capability Version. This field is a PCI-SIG defined version number that indicates the version of the capability structure present. Must be 0x1 for this version of the specification.
Nextpointer	31:20	RO	0x0	Next Capability Offset. This field contains the offset to the next PCIe extended capability structure or 0x000 if no other items exist in the linked list of capabilities.

11.30.2.45 PCIE_VF_CFG_ACS_CAP[0..63]

Address:	**0x130000 + 0x400*i + 0x77**
Atomicity:	**32**
Reset Domains:	**DEVICE, MASTER, COLD, PCIE_HOT, PCIE_WARM**

Field Name	Bit(s)	Type	Default	Description
SourceValidation	0	RO	0b	The FM10000 does support.
TranslationBlocking	1	RO	0b	The FM10000 does not support.
P2PRequestRedirect	2	RO	0b	The FM10000 does not support.
P2PCompletionRedirect	3	RO	0b	The FM10000 does not support.
UpstreamForwarding	4	RO	0b	The FM10000 does not support.
P2PEgressControl	5	RO	0b	The FM10000 does not support.
DirectTranslatedP2P	6	RO	0b	The FM10000 does not support.
Reserved	7	RSV	0b	Reserved.
EgressControlVectorSize	15:8	RO	0x0	The FM10000 does not support.
SourceValidationEnable	16	RO	0b	Must be hardwired to 0b since the ACS Source Validation functionality is not supported.
TranslationBlockingEnable	17	RO	0b	The FM10000 does not support.
P2PRequestRedirectEnable	18	RO	0b	The FM10000 does not support.
P2PCompletionRedirectEnable	19	RO	0b	The FM10000 does not support.
UpstreamForwardingEnable	20	RO	0b	The FM10000 does not support.
P2PEgressControlEnable	21	RO	0b	The FM10000 does not support.
DirectTranslatedP2PEnable	22	RO	0b	The FM10000 does not support.
Reserved	31:23	RSV	0x0	Reserved.

11.31 TE Registers Description

11.31.1 TE Map

Table 11-43 Tunneling Engine Register Set Map

Name	Address (Base = 0xA00000)	Atomicity	Brief Description
TE_DATA[0..1][0..57343]	Base + 0x100000*j + 0x4*i + 0x0	64	Flow and Tunnel Definition.
TE_LOOKUP[0..1][0..32767]	Base + 0x100000*j + 0x2*i + 0x40000	64	Flow Pointers.
TE_STATS[0..1][0..4095]	Base + 0x100000*j + 0x4*i + 0x50000	64	Frames and Byte Counters.
TE_USED[0..1][0..1024]	Base + 0x100000*j + 0x2*i + 0x54000	64	Track flow usage.
TE_DGLORT_MAP[0..1][0..7]	Base + 0x100000*j + 0x2*i + 0x55000	64	DGLORT Decoder Match.
TE_DGLORT_DEC[0..1][0..7]	Base + 0x100000*j + 0x2*i + 0x55010	64	DGLORT Decoder Configuration.
TE_SGLORT_MAP[0..1][0..7]	Base + 0x100000*j + 0x2*i + 0x55020	64	SGLORT Decoder Match.
TE_SGLORT_DEC[0..1][0..7]	Base + 0x100000*j + 0x2*i + 0x55030	64	SGLORT Decoder Configuration.
TE_DEFAULT_DGLORT[0..1]	Base + 0x100000*i + 0x55040	64	Default DGLORT for each direction.
TE_DEFAULT_SGLORT[0..1]	Base + 0x100000*i + 0x55042	64	Default SGLORT for each direction.
TE_SIP[0..1][0..255]	Base + 0x100000*j + 0x4*i + 0x55400	32	Default SIP Encapsulation.
TE_VNI[0..1][0..255]	Base + 0x100000*j + 0x2*i + 0x55800	64	Default VNI Encapsulation.
TE_DEFAULT_L4DST[0..1]	Base + 0x100000*i + 0x55A00	64	Default UDP Port for NGE and VXLAN Encap.
TE_CFG[0..1]	Base + 0x100000*i + 0x55A02	64	Tunneling Engine Configuration.
TE_PORTS[0..1]	Base + 0x100000*i + 0x55A04	64	UDP Ports for NGE/VXLAN Decap Protocol Detection.
TE_TUN_HEADER_CFG[0..1]	Base + 0x100000*i + 0x55A06	64	Tunnel Header Configuration.
TE_DROP_COUNT[0..1]	Base + 0x100000*i + 0x55A08	64	Frame Drop Counter.
TE_TRAP_DGLORT[0..1]	Base + 0x100000*i + 0x55A0A	64	Trap DGLORT.
TE_DEFAULT_NGE_DATA[0..1][0..15]	Base + 0x100000*j + 0x2*i + 0x55A20	64	Default data for NGE.
TE_DEFAULT_NGE_MASK[0..1]	Base + 0x100000*i + 0x55A40	64	Default Mask.
TE_EXVET[0..1]	Base + 0x100000*i + 0x55A42	64	Extended Ethernet Type Support.

Table 11-43 Tunneling Engine Register Set Map [Continued]

Name	Address (Base = 0xA00000)	Atomicity	Brief Description
TE_FRAMES_IN[0..1]	Base + 0x100000*i + 0x55A44	64	Count frames received by TE.
TE_FRAMES_DONE[0..1]	Base + 0x100000*i + 0x55A46	64	Count frames fully processed by TE.
TE_DMAC[0..1]	Base + 0x100000*i + 0x55A48	64	DMAC to use when adding a tunnel.
TE_SMAC[0..1]	Base + 0x100000*i + 0x55A4A	64	SMAC to use when adding a tunnel.
TE_TRAP_CONFIG[0..1]	Base + 0x100000*i + 0x55A4C	64	Trap control for special cases.
TE_TIMETAG_PREFIX[0..1]	Base + 0x100000*i + 0x55A4E	64	Time tag prefix.
TE_SRAM_CTRL[0..1]	Base + 0x100000*i + 0x55A50	64	Memory controller.
TE_IP[0..1]	Base + 0x100000*i + 0x55A52	64	Interrupt pending.
TE_IM[0..1]	Base + 0x100000*i + 0x55A54	64	Interrupt mask.
TE_SYSTIME[0..1]	Base + 0x100000*i + 0x55A56	64	System Time.
TE_SYSTIME0[0..1]	Base + 0x100000*i + 0x55A58	64	System Time Initial Value.
TE_SYSTIME_CFG[0..1]	Base + 0x100000*i + 0x55A5A	64	System Time Configuration.
TE_ERR_FRAMES_IN[0..1]	Base + 0x100000*i + 0x55A5C	64	Count error frames received by TE.

11.31.2 TE Enumerated Data Types

Table 11-44 TE Enumerated Data Types

Name	Width	Description
TE_TRAP_DISP	3	Disposition for a single trap case. 000b = TRAP_DGLORT0 — Send to TRAP_DGLORT. 001b = TRAP_DGLORT1 — Send to TRAP_DGLORT+1. 010b = TRAP_DGLORT2 — Send to TRAP_DGLORT+2. 011b = TRAP_DGLORT3 — Send to TRAP_DGLORT+3. 100b = PASS — Forward to DGLORT set by TE_DGLORT_DEC (either DGLORT on incoming frame or TE_DEFAULT_DGLORT). 101b = PASS_WITH_ERROR — As in PASS case, but invalidate at end of frame. 110b = DROP — Do not forward the frame. Increment TE_DROP_COUNT, TE_FRAMES_DONE, and USED/STATS only. 111b = Reserved — This value must not be programmed into the TE_TRAP_CONFIG register.

11.31.3 TE Structures

11.31.3.1 TE_KEY_MASK

Table 11-45 TE_KEY_MASK Structure

Name	Bit(s)	Description
KeyID	0	Defines if VSI (encap) or TEP-ID (decap) is used for search.
KeyVNI	1	Defines if VNI (decap VXLAN/NGE) or TNI (decap NVGRE) is used for search. Must set to 0 on encap.
KeyDMAC	2	Defines if DMAC is used for search.
KeySMAC	3	Defines if SMAC is used for search.
KeyVID	4	Defines if VLAN ID is used for search.
KeyIPv6	5	Defines if key is IPV6.
KeyDIP	6	Defines if L3 destination IP is used for search.
KeySIP	7	Defines if L3 source IP address is used for search.
KeyL4SRC	8	Defines if L4 source port is used for search.
KeyL4DST	9	Defines if L4 destination port is used for search.
KeyPROT	10	Defines if PROTOCOL is used for search.
KeyExtra	13:11	An extra 3 bit supplied for match.
isProtocolUDP	14	Defines if PROTOCOL=UDP is a match.
isProtocolTCP	15	Defines if PROTOCOL=TCP is a match.
Reserved	31:16	Reserved.

11.31.3.2 TE_ENCAP_FLOW_DATA_MASK

Table 11-46 TE_ENCAP_FLOW_DATA_MASK Structure

Name	Bit(s)	Description
VNI	0	VNI for this packet.
CounterPtr	1	A pointer to a counter structure is specified. If this is not set, this flow is counted on a default counter.
InnerDMAC	2	New destination MAC address present.
InnerSMAC	3	New source MAC address present.
InnerVID	4	New source MAC address present.
IsIPv6	5	L3 IP address supplied are IPv6.
InnerDIP	6	New L3 IP destination address present.
InnerSIP	7	New L3 IP source address present.
InnerL4SRC	8	New L4 source port present.
InnerL4DST	9	New L4 destination port present.
InnerTTL	10	New TTL present.
NgeData	11	New NGE data present.
Tunneling	12	Indicates if tunneling information is supplied. If not (0b), no tunneling is added, only NAT is performed.

Table 11-46 TE_ENCAP_FLOW_DATA_MASK Structure [Continued]

Name	Bit(s)	Description
TunnelPtr	13	Indicates if the tunnel information following the flow data is a pointer to where the tunnel data is (1b) or if the tunnel data is immediately following the flow data.
LoadTimetag	14	Indicates if the timetag should be loaded into NGE.
Reserved	31:15	Reserved.

11.31.3.3 TE_ENCAP_TUNNEL_DATA_MASK

Table 11-47 TE_ENCAP_TUNNEL_DATA_MASK Structure

Name	Bit(s)	Description
TunnelType	1:0	Defines the tunnel type: 00b = GENERIC 01b = VXLAN 10b = NGE 11b = NVGRE
IsIPv6	2	Indicates the outer header is IPV6.
OuterSIP	3	Tunnel source IP address.
OuterTOS	4	Tunnel TOS.
OuterTTL	5	Tunnel TTL.
OuterL4DST	6	Tunnel L4 destination.
OuterL4SRC	7	Tunnel L4 source.
CounterPtr	8	A pointer to a counter structure is specified. If this is not set, then this flow is counted on a default counter.
NgeData	9	Tunnel NGE data.
LoadTimetag	10	Indicates if the timetag should be loaded into NGE.
Reserved	31:11	Reserved.

11.31.3.4 TE_DECAP_FLOW_DATA_MASK

Table 11-48 TE_DECAP_FLOW_DATA_MASK Structure

Name	Bit(s)	Description
OuterHeaderDisp	2:0	Outer header disposition: 000b = DELETE 001b = LEAVE_AS_IS 010b = MOVE_TO_END All other values are reserved.
DGLORT	3	New DGLORT for packet.
InnerDMAC	4	New destination MAC address present.
InnerSMAC	5	New source MAC address present.
InnerVID	6	New VID present.
IsIPv6	7	L3 IP address supplied are IPv6.
InnerDIP	8	New L3 IP destination address present.

Table 11-48 TE_DECAP_FLOW_DATA_MASK Structure [Continued]

Name	Bit(s)	Description
InnerSIP	9	New L3 IP source address present.
InnerTTL	10	New TTL present.
InnerL4SRC	11	New L4 source port present.
InnerL4DST	12	New L4 destination port present.
CounterPtr	13	A pointer to a counter structure is specified. If this is not set, this flow is counted on a default counter.
Reserved	31:14	Reserved.

11.31.4 TE Registers

11.31.4.1 TE_DATA[0..1][0..57343]

Contains tunnel and data information. The register is 128 bits wide but only readable/writable in 64-bit chunks. To modify while traffic is flowing, use TE_FRAMES_IN/TE_FRAMES_DONE.

Address:	**0xA00000 + 0x100000*j + 0x4*i + 0x0**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DataHigh	63:0	RW	0x0	Lower 64 bits of 128-bit word.
DataHigh	127:64	RW	0x0	Higher 64 bits of 128-bit word.

11.31.4.2 TE_LOOKUP[0..1][0..32767]

To modify while traffic is flowing, use TE_FRAMES_IN/TE_FRAMES_DONE.

Address:	**0xA00000 + 0x100000*j + 0x2*i + 0x40000**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DataPointer	15:0	RW	0x0	Pointer to data structure in TE_DATA (128-bit addressing).
DataLength	22:16	RW	0x1	Length of data structure in TE_DATA (in 128-bit words).
Last	23	RW	1b	Indicates if the pointer points to the last block of data structures or there is a chain.
Reserved	31:24	RSV	0x0	Reserved.

11.31.4.3 TE_STATS[0..1][0..4095]

Address: 0xA00000 + 0x100000*j + 0x4*i + 0x50000
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Frames	47:0	vRW	0x0	Number of frames forwarded to switch.
Reserved	63:48	RSV	0x0	Reserved.
Bytes	119:64	vRW	0x0	Number of bytes transmitted to switch. After frame modification.
Reserved	127:120	RSV	0x0	Reserved.

11.31.4.4 TE_USED[0..1][0..1024]

Address: 0xA00000 + 0x100000*j + 0x2*i + 0x54000
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
Used[0..63]	63:0	vRW	0b	(64 x 1-bit) Tracks if a flow is used. The hardware set the bit TE_USED[flowPtr/64].*Used*[flowPtr%64] each time a flow matches a packet received. The software clears the table periodically and can use this table to detect which flows are still used.

11.31.4.5 TE_DGLORT_MAP[0..1][0..7]

The register set is used for comparing incoming DGLORT,USER from fabric to known patterns. The patterns are in form a *VALUE*/*MASK*. The DGLORT and USER are compared to all patterns, and the highest matching pattern defines how the DGLORT is interpreted. To modify while traffic is flowing, use TE_FRAMES_IN/TE_FRAMES_DONE.

Address: 0xA00000 + 0x100000*j + 0x2*i + 0x55000
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
DGLORT_Value	15:0	RW	0xFFFF	Value to compare.
DGLORT_Mask	31:16	RW	0x0	Selects which bits to compare. • Setting *DGLORT_Mask* to 0xFFFF means all bits are significant for comparison to *DGLORT_Value*. • Setting *DGLORT_Mask* to 0x0000 and *DGLORT_Value* to 0x0000 is always match. • Setting *DGLORT_Mask* to 0x0000 and a *DGLORT_Value* to other than 0x0000 never matches.
UserValue	39:32	RW	0x0	Value to compare.
UserMask	47:40	RW	0x0	Selects which bits to compare. • Setting *UserMask* to 0xFFFF means all bits are significant for comparison to *UserValue*. • Setting *UserMask* to 0x0000 and *UserValue* to 0x0000 is always match. • Setting *UserMask* to 0x0000 and a *UserValue* to other than 0x0000 never matches.
Reserved	63:48	RSV	0x0	Reserved.

11.31.4.6 TE_DGLORT_DEC[0..1][0..7]

To modify while traffic is flowing, use TE_FRAMES_IN/TE_FRAMES_DONE.

Address:	**0xA00000 + 0x100000*j + 0x2*i + 0x55010**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Base	15:0	RW	0x0	Defines the base of the lookup table for this DGLORT decoder in the TE_LOOKUP table.
HashKeyConfig	31:16	RW	0x0	If *Hash* is 1b, this field specifies which keys are included in search. The structure of this field is TE_KEY_MASK (refer to Section 11.31.3.1). The fields *KeyIPv6*, *isProtocolUDP* and *isProtocolTCP* must be set to 0b.
HashSize	35:32	RW	0x0	Defines the hash table size in power of 2, minus 1. Allowed values are 0..14. The size is 2^^(HashSize+1).
Encap	36	RW	0b	Defines if the change is Encap/NAT or Decap. 0b = Decap 1b = Encap/NAT
Hash	37	RW	0b	Defines if the index is extracted from {DGLORT,USER} or a hash key from frame header fields.
IndexStart	42:38	RW	0x0	Defines location of embedded index in {DGLORT[15:0],USER[7:0]} set. If *Hash* is 0b, this is an index used to recover the flow-data. If *Hash* is 1b, this field is used to define the location of the TEP-ID field in the {DGLORT[15:0],USER[7:0]} set, only for decap. This field is not used if *Hash* is 1b and *Encap* is 1b.
IndexLength	46:43	RW	0x0	Defines width of embedded index in {DGLORT[15:0],USER[7:0]} set. If *Hash* is 0b, this is an index used to recover the flow-data. If *Hash* is 1b, this field is used to define width of the TEP-ID field in the {DGLORT[15:0],USER[7:0]} set, only for decap. This field is not used if *Hash* is 1b and *Encap* is 1b.
SetSGLORT	47	RW	0b	Defines if SGLORT is set to TE_DEFAULT_SGLORT or pass-through. 0b = Pass-through 1b = TE_DEFAULT_SGLORT Only applicable if *flowData* does not define a DGLORT to use.
SetDGLORT	48	RW	0b	Defines if SGLORT is set to TE_DEFAULT_DGLORT or pass-through. 0b = Pass-through 1b = TE_DEFAULT_DGLORT Only applicable if *flowData* does not define a DGLORT to use.
Reserved	63:49	RSV	0x0	Reserved.

11.31.4.7 TE_SGLORT_MAP[0..1][0..7]

The register set is used for comparing incoming SGLORT from fabric to known patterns. The patterns are in form a *VALUE*/*MASK*. The SGLORT are compared to all patterns and the highest matching pattern defines how the SGLORT is interpreted. To modify while traffic is flowing, use TE_FRAMES_IN/ TE_FRAMES_DONE.

Address: **0xA00000 + 0x100000*j + 0x2*i + 0x55020**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SGLORT_Value	15:0	RW	0xFFFF	Value to compare.
SGLORT_Mask	31:16	RW	0x0	Selects which bits to compare. • Setting *SGLORT_Mask* to 0xFFFF means all bits are significant for comparison to *SGLORT_Value*. • Setting *SGLORT_Mask* to 0x0000 and *SGLORT_Value* to 0x0000 is always a match. • Setting *SGLORT_Mask* to 0x0000 and *SGLORT_Value* to other than 0x0000 never matches.

11.31.4.8 TE_SGLORT_DEC[0..1][0..7]

To modify while traffic is flowing, use TE_FRAMES_IN/TE_FRAMES_DONE.

Address: **0xA00000 + 0x100000*j + 0x2*i + 0x55030**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
VSI_Start	4:0	RW	0x0	Indicates the location of first bit of VSI in SGLORT.
VSI_Length	9:5	RW	0x0	Indicates length of VSI in SGLORT. Value of 0b means VSI set to VSI_Offset.
VSI_Offset	17:10	RW	0x0	The offset to add to the VSI extracted from SGLORT.
Reserved	31:18	RSV	0x0	Reserved.

11.31.4.9 TE_DEFAULT_DGLORT[0..1]

Address: **0xA00000 + 0x100000*i + 0x55040**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DefaultEncapDGLORT	15:0	RW	0x0	Defines which DGLORT to use during encap if not pass-through.
DefaultDecapDGLORT	31:16	RW	0x0	Defines which DGLORT to use during decap if not pass-through.

11.31.4.10 TE_DEFAULT_SGLORT[0..1]

Address: **0xA00000 + 0x100000*i + 0x55042**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DefaultEncapSGLORT	15:0	RW	0x0	Defines which SGLORT to use during encap if not pass-through.
DefaultDecapSGLORT	31:16	RW	0x0	Defines which SGLORT to use during encap if not pass-through.

11.31.4.11 TE_SIP[0..1][0..255]

Default SIP for each TEP. Specific location is indexed through the VSI extracted from the TE_SGLORT_DEC.

Note: This register is 64-bit atomic. Hardware may see an intermediate value between two 64-bit writes to a 128-bit entry.

Address: **0xA00000 + 0x100000*j + 0x4*i + 0x55400**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
SIP	127:0	RW	0x0	Defines default Source IP address.

11.31.4.12 TE_VNI[0..1][0..255]

Default VNI/TNI value and presence for each TEP. Specific location is indexed through the VSI extracted from the TE_SGLORT_DEC.

Address: **0xA00000 + 0x100000*j + 0x2*i + 0x55800**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
VNI	23:0	RW	0x0	Defines the default L4 destination port for VXLAN encap.
Reserved	31:24	RSV	0x0	Reserved.

11.31.4.13 TE_DEFAULT_L4DST[0..1]

Address: **0xA00000 + 0x100000*i + 0x55A00**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
VXLAN	15:0	RW	0x0	Defines the default L4 destination port for VXLAN encap.
NGE	31:16	RW	0x0	Defines the default L4 destination port for NGE encap.

11.31.4.14 TE_CFG[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A02**
Atomicity:	**64**
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
OuterTTL	7:0	RW	0x0	Defines default TTL for tunnels during encapsulation if none specified with tunnel parameters. If this register is set to 0b, the outer IP header uses the inner TTL received. This parameter is not used during NAT-only operations or during decapsulation.
OuterTOS	15:8	RW	0x0	Defines default TOS for tunnels during encapsulation if none specified with tunnel parameters and *DeriveOuterTOS* is off.
DeriveOuterTOS	16	RW	0b	Defines how the *OuterTOS* is set during encapsulation if none specified with tunnel parameters. Choices are: 0b = Use the default *OuterTOS* in this register. 1b = Copy over the *InnerTOS* from packet.
NotIP	18:17	RW	00b	Defines parser options for non IP packet, applicable to encap only. 00b = TRAP — Trap according to TE_TRAP_CONFIG. 01b = ZERO_CSUM — No need to compute checksum. VXLAN/ NGE tunnel header is set to a zero checksum at exit. 10b = COMPUTE_CSUM — Compute checksum from entire packet starting after FTAG and stopping at end of packet. 11b = Reserved. The packet length is also measured by parser as there is no IP header present. Thus, the packet is in store-and-forward mode for both ZERO_CSUM and COMPUTER_CSUM modes. The packet is in cut-through mode if TRAP is selected.
IPnotTCPnotUDP	20:19	RW	00b	Define options for IP packets that are not TCP or UDP, applicable to encap only. 00b = TRAP — Trap according to TE_TRAP_CONFIG. 01b = ZERO_CSUM — No need to compute checksum. VXLAN/ NGE tunnel header is set to a zero checksum at exit. 10b = COMPUTE_CSUM — Compute VXLAN/NGE checksum on packet payload stopping at length defined in IP header. 11b = Reserved. The length is recovered from IP header length. The packet is store-and-forward if COMPUTE_CSUM is selected, and cut-through otherwise.
IPisTCPorUDP	22:21	RW	11b	Define options for IP packets with TCP or UDP protocols: 00b = TRAP — Trap according to TE_TRAP_CONFIG. 01b = ZERO_CSUM — Set VXLAN/NGE checksum set to 0b. 10b = COMPUTE_CSUM — Compute VXLAN/NGE checksum on packet payload stopping at length defined in IP header. 11b = HEADER_CSUM — Recover checksum from UDP/TCP header. This checksum is used for computing VXLAN/NGE checksum. If this checksum is zero, TE computes the outer checksum walking over the entire IP packet. The length is recovered from IP header length. The packet is store-and-forward if COMPUTE_CSUM is selected or if HEADER_CSUM is selected and the checksum in the incoming UDP or TCP header is zero, and cut-through otherwise.
VerifyDecapCSUM	23	RW	0b	Defines processing of outer VXLAN/NGE UDP checksum. 0b = Does not check the value (cut-through). 1b = The checksum is validated for VXLAN and NGE if and only if the CSUM is not 0b (store-and-forward required). If found invalid, the packet is trapped according to TE_TRAP_CONFIG.

Address: Atomicity: Reset Domains:	0xA00000 + 0x100000*i + 0x55A02 64 DEVICE, MASTER, COLD, SWITCH			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
UpdateOldHeaderInPlaceCSUM	24	RW	1b	Defines value generated for outer VXLAN/NGE UDP checksum in decap old-header-in-place mode. 0b = Overwrite old-header checksum to 0. 1b = Add value changes into old-header checksum.
SwitchLoopbackDisable	25	RW	0b	When cleared, the switch and TE are connected in independent loopbacks. May not be written while traffic is flowing.
Reserved	31:26	RSV	0x0	Reserved.

11.31.4.15 TE_PORTS[0..1]

The TE may read TE_PORTS multiple times per packet. Therefore, this register should not be changed while the TE is receiving traffic.

Address: Atomicity: Reset Domains:	0xA00000 + 0x100000*i + 0x55A04 64 DEVICE, MASTER, COLD, SWITCH			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
VXLAN	15:0	RW	0x0	Defines the L4 destination port used for VXLAN decap. Setting to 0 disables VXLAN detection.
NGE	31:16	RW	0x0	Defines the L4 destination port used for NGE decap. Setting to 0 disables NGE detection.

11.31.4.16 TE_TUN_HEADER_CFG[0..1]

The TE may read TE_TUN_HEADER_CFG multiple times per packet. Therefore, this register should not be changed while the TE is receiving traffic.

Address: Atomicity: Reset Domains:	0xA00000 + 0x100000*i + 0x55A06 64 DEVICE, MASTER, COLD, SWITCH			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
EncapProtocol	15:0	RW	0x6558	Defines the protocol value that TE puts in the NVGRE and NGE and headers when encapsulating. Can also be used to check protocol on decap. When GPE mode (*EncapVersion*[2]) is set, *EncapProtocol* is interpreted as {NSH_PROTOCOL[7:0],L2_PROTOCOL[7:0]}. NVGRE and NGE encapsulation should not be used in GPE mode because *EncapProtocol* is still placed in the header when encapping, but the value is be 0x6558 (as required for NVGRE/NGE). Consider using *TunnelType*=GENERIC exclusively if any GPE decap is being used. *TunnelType*=GENERIC encapsulation does not use *EncapProtocol*. Also consider dedicating one TE exclusively to VXLAN and GPE.
EncapVersion	18:16	RW	000b	Defines the version value that TE puts in the NVGRE and NGE headers when encapsulating. Can also be used to check version on decap. *EncapVersion*[2] enables GPE mode decap. *EncapVersion*[2] is not applied to the NVGRE header version.

Address: **0xA00000 + 0x100000*i + 0x55A06**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
CheckProtocol	19	RW	1b	When set, the TE checks the NVGRE/NGE header protocol on decap and traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN* if the header protocol does not match *EncapProtocol*.
CheckVersion	20	RW	1b	When set, the TE checks the NVGRE/NGE header version on decap and traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN* if the header version does not match *EncapVersion*. When GPE mode (*EncapVersion*[2]) and *CheckVersion* are set, the TE checks both the VXLAN-GPE and NSH header versions. If either header version does not match EncapVersion then TE traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN*.
CheckOam	21	RW	1b	When set, the TE checks the NGE OAM bits on decap and traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN* if this bit is not zero. When GPE mode (*EncapVersion*[2]) and *CheckOam* are set, the TE checks both the VXLAN-GPE and NSH header OAM bits. If either header OAM bit is set then TE traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN*.
CheckC	22	RW	0b	When set, the TE checks the NGE C bit on decap and traps according to TE_TRAP_CONFIG.*DECAP_BAD_TUN* if this bit is not zero.
Reserved	31:23	RSV	0x0	Reserved.

11.31.4.17 TE_DROP_COUNT[0..1]

Address: **0xA00000 + 0x100000*i + 0x55A08**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DropCount	15:0	RW	0x0	Counts number of frames dropped.
Reserved	31:16	RSV	0x0	Reserved.

11.31.4.18 TE_TRAP_DGLORT[0..1]

Address: **0xA00000 + 0x100000*i + 0x55A0A**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
TrapDGLORT	15:0	RW	0x0	Defines the DGLORT to use for frames that are trapped (usually because of no match found).
Reserved	31:16	RSV	0x0	Reserved.

11.31.4.19 TE_DEFAULT_NGE_DATA[0..1][0..15]

Address: **0xA00000 + 0x100000*j + 0x2*i + 0x55A20**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
Data	31:0	RW	0x0	Defines the value.

11.31.4.20 TE_DEFAULT_NGE_MASK[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A40**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Mask	15:0	RW	0x0	Defines which words are valid in TE_DEFAULT_NGE_DATA[0..15]
LoadTimetag	16	RW	0b	Defines if the timetag should be loaded into words[14:15] when TE_DEFAULT_NGE_MASK is used. If this bit is set, Mask[15] must also be set.
Reserved	31:16	RSV	0x0	Reserved.

11.31.4.21 TE_EXVET[0..1]

The TE may read TE_EXVET multiple times per packet, so this register should not be changed while the TE is receiving traffic.

Address:	**0xA00000 + 0x100000*i + 0x55A42**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
EthernetType	15:0	RW	0x8100	Extra Ethernet type. Parser jumps over if detected.
TagSize	17:16	RW	00b	Size in 16-bit quanta. Size supported are 1,2, excluding the 16-bit EthernetType itself. Setting to 0 disables detection of this type. Setting to 3 only works if no VLAN tag present.
AfterVLAN	18	RW	0b	Defines if the extended tag is expected before or after VLAN (0x8100) tag. 0b = Before VLAN tag. 1b = After VLAN tag.
Reserved	31:19	RSV	0x0	Reserved.

11.31.4.22 TE_FRAMES_IN[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A44**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FrameCount	31:0	vRW	0x0	Count number of frame into TE.

11.31.4.23 TE_FRAMES_DONE[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A46**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FrameCount	31:0	vRW	0x0	Counter number of frames done by TE. Includes frames trapped, dropped, pass-through or processed.

11.31.4.24 TE_DMAC[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A48**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
DMAC	47:0	RW	0x0	DMAC to use when adding a tunnel.
Reserved	63:48	RSV	0x0	Reserved.

11.31.4.25 TE_SMAC[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A4A**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
SMAC	47:0	RW	0x0	SMAC to use when adding a tunnel.
Reserved	63:48	RSV	0x0	Reserved.

11.31.4.26 TE_TRAP_CONFIG[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A4C**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
NORMAL	2:0	RW	100b	Normal matching flow. In this case the only legal values are PASS, PASS_WITH_ERROR, and DROP. Behavior is undefined otherwise. Uses type *TE_TRAP_DISP* (see Table 11-44).
NO_DGLORT_MATCH	5:3	RW	100b	Incoming frame's DGLORT did not match any entry in DGLORT_MAP. Uses type *TE_TRAP_DISP* (see Table 11-44).
NO_SGLORT_MATCH	8:6	RW	100b	Incoming frame's SGLORT did not match any entry in SGLORT_MAP. Uses type *TE_TRAP_DISP* (see Table 11-44).
DECAP_NO_OUTER_L3	11:9	RW	000b	No outer L3 header found on frame to be decapped (i.e., decap and outer ether_type != IP). Uses type *TE_TRAP_DISP* (see Table 11-44).
DECAP_NO_OUTER_L4	14:12	RW	000b	No outer L4 header found on frame to be decapped (i.e., decap and outer prot !=UDP/GRE). Uses type *TE_TRAP_DISP* (see Table 11-44).
DECAP_NO_OUTER_TUN	17:15	RW	000b	No outer tunnel header found on frame to be decapped (i.e., VXLAN/NGE specified but UDP port does not match one in TE_PORTS). Uses type *TE_TRAP_DISP* (see Table 11-44).
DECAP_BAD_CSUM	20:18	RW	000b	Bad outer L4 CSUM on decap, and tTE_CFG.*VerifyDecapCSUM* set. Uses type *TE_TRAP_DISP* (see Table 11-44).
DECAP_BAD_TUN	23:21	RW	000b	Bad tunneling version, nonzero OAM/C bits, or encap protocol found in tunnel header, and *DECAP_BAD_TUN* corresponding check enabled in TE_TUN_HEADER_CFG. Uses type *TE_TRAP_DISP* (see Table 11-44).
ENCAP_NO_L3	26:24	RW	000b	No L3 header found on frame to be encapped and TE_CFG.*NotIP*==TRAP. Uses type *TE_TRAP_DISP* (see Table 11-44).

Address: **0xA00000 + 0x100000*i + 0x55A4C**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
ENCAP_NO_L4	29:27	RW	000b	No L4 header found on frame to be encapped and TE_CFG.*IPnotTCPnotUDP*==TRAP. Uses type *TE_TRAP_DISP* (see Table 11-44).
ENCAP_ANY_L4	32:30	RW	000b	Encountered L4 frame and TE_CFG.*IPisTCPorUDP*==TRAP. Uses type *TE_TRAP_DISP* (see Table 11-44).
TRUNCATED_HEADER	35:33	RW	000b	The frame headers were not completely formed. The frame ended during an L2/L3/L4/Tunnel header that was specified in the frame. Uses type *TE_TRAP_DISP* (see Table 11-44).
NO_FLOW_MATCH	38:36	RW	000b	No matching flow record in found in TE_DATA. Uses type *TE_TRAP_DISP* (see Table 11-44).
MISSING_RECORD	41:39	RW	000b	TE_DATA string ended before all expected records were decoded. Uses type *TE_TRAP_DISP* (see Table 11-44).
UC_ERR	44:42	RW	110b	Uncorrectable ECC error encountered in TE_DATA or TE_LOOKUP. Also see TE_DATA_BOUNDS field description in this register. Uses type *TE_TRAP_DISP* (see Table 11-44).
TE_LOOKUP_BOUNDS	47:45	RW	110b	TE_LOOKUP hash/index out of bounds. This could be caused by an invalid hash table size (>32K) or by a smaller table that crosses above the last word implemented (i.e., reaching >=32K). Wrapping back to TE_LOOKUP[0] is not supported. Uses type *TE_TRAP_DISP* (see Table 11-44).
TE_DATA_BOUNDS	50:48	RW	110b	TE_DATA access out of bounds. This trap may also be raised in the case of an uncorrectable error in TE_LOOKUP. Uses type *TE_TRAP_DISP* (see Table 11-44).
TOTAL_HEADER_LIMIT	51:53	RW	110b	Encountered an IPv6 extension header which would cause the total header length (outer plus inner) to exceed 512 bytes. Uses type *TE_TRAP_DISP* (see Table 11-44).
Reserved	63:54	RSV	0x0	Reserved.

11.31.4.27 TE_TIMETAG_PREFIX[0..1]

Address: **0xA00000 + 0x100000*i + 0x55A4E**
Atomicity: **64**
Reset Domains: **DEVICE, MASTER, COLD, SWITCH**

Field Name	Bit(s)	Type	Default	Description
DefaultTag	31:0	RW	0x0	Default value for time tag prefix. One of the byte might be replaced by the top 8 bits of the time tag.
Top8bLocation	35:32	RW	0x0	Defines where the top 8 bits of the time tag is loaded in the NGE prefix. This is a 4-bit mask with 1-hot encoding. If bit 0 is set, then the top 8 bits replaces DefaultTag[7:0]. A value of 0 means the top 8 bits of the time tag is not loaded.
Reserved	63:36	RSV	0x0	Reserved.

11.31.4.28 TE_SRAM_CTRL[0..1]

This register controls the TE memory cores: te_data=0, te_lookup=1, te_sip=2, te_vni=3, te_payload_fifo=4, te_header_fifo=5, te_stats=6, te_used=7

Address: 0xA00000 + 0x100000*i + 0x55A50
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
ErrWrite[0..7]	15:0	RW	00b	(8 x 2-bits)
CErr[0..7]	23:16	vCW1	0b	(8 x 1-bit)
UErr[0..7]	31:24	vCW1	0b	(8 x 1-bit)
BistDonePass[0..7]	39:32	vRO	0b	(8 x 1-bit)
BistDoneFail[0..7]	47:40	vRO	0b	(8 x 1-bit)
Reserved	63:48	RSV	0x0	Reserved.

11.31.4.29 TE_IP[0..1]

Interrupt pending register for TE.

Address: 0xA00000 + 0x100000*i + 0x55A52
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
TeSramErr	1:0	vCW1	00b	SRAM interrupt. There are two bits: Uncorrected Error (MSB), Corrected Error (LSB). The memory core is identified in TE_SRAM_CTRL[0..1].
TePcErr	2	vCW1	0b	Non-quad-mode packet received on portchannel.
Reserved	31:3	RSV	0x0	Reserved.

11.31.4.30 TE_IM[0..1]

The interrupt mask register controls if the corresponding interrupt source is presented to the interrupt hierarchy (bit is set to 0b) or masked out (bit is set to 1b).

Address: 0xA00000 + 0x100000*i + 0x55A54
Atomicity: 64
Reset Domains: DEVICE, MASTER, COLD, SWITCH

Field Name	Bit(s)	Type	Default	Description
TeSramErr	1:0	RW	11b	SRAM interrupt. There are two bits: Uncorrected Error (MSB), Corrected Error (LSB).
TePcErr	2	RW	1b	Non-quad-mode packet received on portchannel.
Reserved	31:3	RSV	0x0	Reserved.

11.31.4.31 TE_SYSTIME[0..1]

Time reference. The time unit is dependent on the system implementation, but should normally represent nanoseconds.

Address:	**0xA00000 + 0x100000*i + 0x55A56**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
CurrentTime	39:0	vRO	0x0	System time.
Reserved	63:40	RSV	0x0	Reserved.

11.31.4.32 TE_SYSTIME0[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A58**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Time0	39:0	RW	0x0	The content of this register is transfered to TE_SYSTIME when a reset command is received on the SYSTIME interface.
Reserved	63:40	RSV	0x0	Reserved.

11.31.4.33 TE_SYSTIME_CFG[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A5A**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
Step	3:0	RW	0xA	The number of time unit per SYSTIME tick. Must be at least 2. Values of 1 or 0 are not supported.
Reserved	31:4	RSV	0x0	Reserved.

11.31.4.34 TE_ERR_FRAMES_IN[0..1]

Address:	**0xA00000 + 0x100000*i + 0x55A5C**			
Atomicity:	**64**			
Reset Domains:	**DEVICE, MASTER, COLD, SWITCH**			
Field Name	**Bit(s)**	**Type**	**Default**	**Description**
FrameCount	31:0	vRW	0x0	Counts the number of frames received from the switch marked as error. Wraps.

12.0 Electrical Specification

12.1 Absolute Maximum Ratings

Parameter	Symbol	Min	Max	Units
Core Supply Voltage	VDDS	-0.3	1.3	V
Core Supply Voltage	VDDF	-0.3	1.3	V
SerDes Analog Supply Voltage	AVDD	-0.3	1.3	V
PLL Analog Supply Voltage	PLLVDD	-0.3	1.3	V
LVCMOS & LVPECL Supply Voltage	VDD25	-0.5	3.3	V
LVPECL Termination Supply Voltage	AVDD25	-0.5	3.3	V
LVCMOS Input PAD Voltage	VI-ABS	-0.5	VDD25+1.4	V
LVPECL REFCLK Analog Supply Voltage	AVDDCLK	-0.5	1.3	V
LVPECL REFCLK Input PAD Voltage	VICLK-ABS	-0.5	2.8	V
Junction Temperature	Tj		115	°C
Storage Temperature		65	150	°C

12.2 ESD

Parameter	Pin Group	Min	Max	Units	Specification
ESD Human Body Model (HBM)	CMOS/LVTTL IO Pins	-1000	1000	V	JEDEC JS-001-2012
	High Speed IO pins	-1000	1000	V	
ESD Charged Device Model (CDM)	CMOS/LVTTL IO Pins	-250	250	V	JESD22-C101F
	High Speed IO pins	-250	250	V	

12.3 Latch-Up

Parameter	Pin Group	I_Min	I_Max	Units	Power Supply	V_Min	V_Max	Units	Spec.	Notes
Latch up[1]	I/O LVCMOS	-100	100	mA	VDD25	-0.50	3.90	V	JESD78D	Compliance voltages are -0.5 V (min) and +3.9 V (max).
	I/O SerDes RX/TX	-100	100	mA	AVDD	-0.50	1.5	V	JESD78D	Compliance voltages are -0.5 V (min) and +1.5 V (max).

[1] Current specification is at compliance limit. If compliance voltage is reached before current limit is reached testing stops and the IO is considered passing.

12.4 Recommended Operating Conditions

12.4.1 Scaled Voltages

Parameter	Symbol	Min	Vnom_min	Vnom_typ	Vnom_max	Max	Units
Core Supply Voltage	VDDS	Vnom – 5%	0.80	0.85	0.95	Vnom + 5%	V
Core Supply Voltage	VDDF	Vnom – 5%	0.80	0.95	1.00	Vnom + 5%	V

Notes:

- Voltages ranges are subject to change and early adopters should plan on supporting higher (or lower) voltages in advance of final qualification.
- The sum of the offset plus voltage ripple must be within the 5% specification.
- VDDF and VDDS voltages must be set to nominal values stored in the on-chip fuse box. These values should be read by the software, and the power supplies adjusted accordingly.
- These values are specific to the SKU number of the device as purchased. SKUs with higher rated cross sectional bandwidth will require higher voltages and will not be power efficient when used in lower bandwidth applications.
- The VDDS and VDDF are specified based as on die values as sensed by the SENSE_VDDS and SENSE_VDDF pins.

12.4.2 Non-Scaled Voltages

Parameter	Symbol	Min	Nom	Max	Units
SerDes Analog Supply Voltage	AVDD	0.95	1.0	1.05	V
LVCMOS, LVTTL Supply Voltage	VDD25	2.375	2.5	2.625	V
LVPECL Termination Voltage	AVDD25	2.375	2.5	2.625	V
PLL Analog Supply Voltage	PLLVDD0	0.95	1.0	1.05	V
PLL Analog Supply Voltage	PLLVDD1	0.95	1.0	1.05	V

12.4.3 Ground Returns

Parameter	Symbol
Digital Supply Ground Return	VSS
SerDes Analog Ground Return	AVSS
PLLVDD0 Analog Ground Return	PLLVSS0
PLLVDD1 Analog Ground Return	PLLVSS1

12.4.4 Environmental

Parameter	Symbol	Min	Nom	Max	Units
Case Temperature	Tc	5		85	°C

12.4.5 Voltage Scaling and Voltage Tolerances

Intel uses voltage scaling to satisfy device performance and power targets. When each part is tested, the required nominal VDDF and VDDS supply voltages are programmed into an on-chip fuse box. The nominal voltages programmed in the device can range from 0.85 V to 1.05 V, with performance guaranteed over a 5% range around the **nominal** programmed voltage in the device.

The ±5% range for both VDDS and VDDF is specified as a **combination** of supply offset + voltage ripple. The sum of the offset plus voltage ripple must be within the 5% specification.

The voltage ripple is considered to be symmetrical about the average supply offset. Thus, the voltage offset is additive to 1/2 the peak-to-peak ripple.

As an example, assume a nominal voltage programmed in the fusebox of 0.90 V on VDDS. In this case a voltage offset of 30 mV with 30 mV of peak-to-peak ripple produces a maximum offset of +45 mV, which is just within spec.

The allowed voltage budget in the case of +30 mV of offset and 30 mV p-p ripple is:

- Vnom = 0.900V ⇒ programmed in the device
- Vmax (average) = 0.930V ⇒ 30 mV of power supply offset
- Vmax-avg + 1/2(Vripple) = 0.945V ⇒ corner for maximum allowed voltage

The allowed voltage budget in the case of -30 mV of offset and 30 mV p-p ripple is:

- Vnom = 0.900V ⇒ programmed in the device
- Vmin (average) = 0.870V ⇒ -30 mV of power supply offset
- Vmin - 1/2(Vripple) = 0.855V ⇒ corner for minimum allowed voltage

The system design must have the capability to adjust both VDDF and VDDS based on the values programmed in the fuse box. If multiple devices are used on the same board, each device must have its VDDF/VDDS adjusted individually to achieve performance and power specified in this document.

12.4.5.1 FM10420 Voltage Scaling

The FM10420 is intended to be operated with VDDS and VDDF set to the same voltage. All tables containing power values with respect to the FM10420 reflect that assumption. Thus, the VDDS and VDDF supplies may be joined at the board level and supplied from a single source with the attendant reduction in system board complexity. If a single supply is joined at the board-level, unifying VDDS and VDDF, the fusebox value for either supply may be read and used as the power supply value. This is because the values for VDDS and VDDF are identical in the case of the FM10420. However, it is still a requirement to use the scaled value.

12.4.6 FM10840/FM10420 Voltage Initial Conditions for Fusebox Read

On first power up, VDDS and VDDF need to be set to a nominal value that guarantees that the fusebox can be read.

Both the FM10840 and the FM10420 are guaranteed to have the fusebox readable with the following conditions:

Parameter	Symbol	Nom	Units
Core Supply Voltage	VDDS	0.9	Volts
Core Supply Voltage	VDDF	0.9	Volts

If the supplies are unified, as can be the case of the FM10420, use 0.9 V for the unified supply values. After reading the fuse values, the part power supplies should be set to the scaled values in the fusebox.

12.4.7 Voltage Tolerances Non-Scaled Voltages

The range of voltage for the non-scaled voltages is specified as a combination of supply offset + voltage ripple. Thus, each supply listed in the Section 12.4.2 must stay within the published Min and Max voltages while summing the ripple and the power supply offset.

12.4.8 Voltage Scaling Fuse Locations and Protocol

The nominal voltage information for VDDS and VDDF are located in the FUSE_DATA register at following locations:

```
FUSE_DATA_0[23:16] => VRM info for VDDS
 0 => apply the default nominal voltage (0.85V)
 1..255 => recommended voltage encoded as an 8-bit VID as per Intel VR12 PWM Specification

FUSE_DATA_0[31:24] => VRM info for VDDF
 0 => apply the default nominal voltage (0.95V)
 1..255 => recommended voltage encoded as an 8-bit VID as per Intel VR12 PWM Specification
```

The register content is only readable after the device is taken out of reset. The recommend process is to first set the voltage to the default nominal listed above, then adjust according to the FUSE_DATA_0 content once this register is accessible by software.

12.4.9 Power On Reset Values

The following table summarizes the voltages at which reset is effectively asserted during power on. Below these values the state of the chip is indeterminate during reset. Fully resetting the chip requires the presence of the external PCIe and Ethernet reference clocks.

Parameter	Symbol	Value	Units
Core Supply Voltage	VDDS	0.80	V
Core Supply Voltage	VDDF	0.80	V
SerDes Analog Supply Voltage	AVDD	0.80	V
LVCMOS, LVTTL Supply Voltage	VDD25	2.00	V
LVPECL Termination Voltage	AVDD25[1]	2.00	V
PLL Analog Supply Voltage	PLLVDD0	N/A[2]	V
PLL Analog Supply Voltage	PLLVDD1	N/A[2]	V

[1] Only required if internal LVPECL termination scheme is selected via reset strapping, such as LED_DATA_[2..0].
[2] PLL clocks are bypassed during power up.

12.5 Power Supply Sequencing

Intel recommends that power supplies be sequenced during power up. The ramp rate of the power supplies must not exceed 50 V/ms. This is to avoid triggering ESD protection circuitry inadvertently. There is no minimum ramp rate.

The sequencing of the supplies is keyed to the power on reset values specified in the previous table. That is, the measurement point for timing the supplies is the point at which the supply in question reaches the power on reset value.

Parameter	Symbol	Min	Max	Units	Description
Rise time of Power Supplies VDDS, VDDF, AVDD	tRise	20		µs	Rise time of low voltage supplies.[1]
Rise time of Power Supply VDD25	tRise	50		µs	Rise time of 2.5 V supply.
Fall time of Power Supplies VDDS, VDDF, AVDD	tFall	20		µs	Fall time of low voltage supplies.[1]
Fall time of Power Supply VDD25	tFall	50		µs	Fall time of 2.5 V supply.
Delay between assertion of Supplies	td	20		µs	Time between reaching POR[2] value of first supply and start of ramp of next supply.
Delay between assertion of Supplies and Reset	tR	20		µs	Time between reaching POR[4] value of all supplies and assertion of RESET.
Ramp rate of power supply	tRamp		50	V/ms	Maximum rate at which a power supply may be ramped on turn on.

[1] Low voltage supplies are considered to have a nominal value of 1 V for this calculation.
[2] POR = Power On Reset.

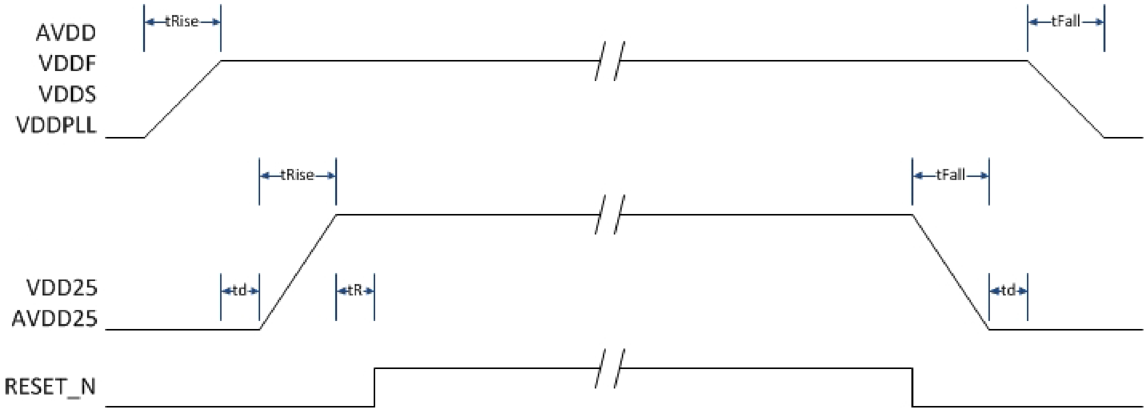


Figure 12-1 Power Supply Sequencing

12.6 Chip Power and Power Supply Currents

12.6.1 Supply Type Definitions

12.6.1.1 High Current Supplies

Defined as power supplies with current requirements greater than 1 Ampere under normal conditions.

Includes power rails: AVDD, VDDF, VDDS

12.6.1.2 Low Current Supplies

Defined as power supplies with current requirements less than 1 Ampere under normal conditions.

Includes power rails: VDD25, AVDD25, PLLVDD0, PLLVDD1

12.6.2 High Current Supply Values

The following tables summarize expected currents for the primary power supplies for each SKU of the FM10000 under typical use cases. The tables also list the expected maximum currents for each SKU.

A recommended Thermal Design Power is given. However, the discussion in Section 12.6.7 should be considered.

12.6.2.1 FM10420

Included in this section are extended discussions regarding the use models of the FM10420 and the expected power for each model.

There are several usage models for the FM10420 SKU. Power dissipation depends on how the device is used, including the number and types of ports used, port utilization and other factors. Power dissipation numbers for three usage models are shown below. Contact your Intel technical representative if you need more information.

12.6.2.1.1 FM10420 Used as a 100 GbE NIC

The primary intended use of the FM10420 is in NIC applications. In the 100 GbE NIC application use model, dual 8-lane PCIe Gen 3 interfaces can only provision up to 100 GbE of Ethernet port bandwidth. As such, the normal use model is with one Ethernet port in use and the other port in standby as a failover port.

The power numbers in the following table are predicated on this use model.

Power Corner	I_{AVDD} (A)	I_{VDDF} (A)	I_{VDDS} (A)	Power (W)
Typical	4.0	2.1	7.6	12.8
Max	4.7	4.8	16.0	24,1
TDP[1]				18.3

[1] See Thermal Design Power discussion in Section 12.6.7.

The basic conditions for the FM10420 used as a 100 GbE NIC are:

Condition	Typical	Maximum	TDP
Traffic	50% 512 B	100% 256 B	66% 512 B
Ethernet Ports	2 x 100 GbE [max 100 GbE Bandwidth]	2 x 100 GbE [max 100 GbE Bandwidth]	2 x 100 GbE [max 100 GbE Bandwidth]
PCIe Ports	2 x (Gen3 x8)	2 x (Gen3 x8)	2 x (Gen3 x8)
Tunneling Engines	2	2	2
Tj (°C)	50	90	87.5

12.6.2.1.2 FM10420 Used as a 2x25 GbE (50 GbE) NIC

The FM10420 also supports dual port 25 GbE NIC applications. In this application, the 8-lane PCIe Gen 3 interface bandwidth can provision up to two 25 GbE ports, so the power specification below assumes both ports are operational.

Power Corner	I_{AVDD} (A)	I_{VDDF} (A)	I_{VDDS} (A)	Power (W)
Typical	1.6	1.2	5.2	7.6
Max	2.2	3.2	11.7	16.1
TDP[1]				12.6

[1] See Thermal Design Power discussion in Section 12.6.7.

The basic conditions for the FM10420 used as a 2x25 GbE NIC are:

Condition	Typical	Maximum	TDP
Traffic	50% 512 B	100% 256 B	66% 512 B
Ethernet Ports	2 x 25 GbE [max 50 GbE Bandwidth]	2 x 25 GbE [max 50 GbE Bandwidth]	2 x 25 GbE [max 50 GbE Bandwidth]
PCIe Ports	1 x (Gen3 x8)	1 x (Gen3 x8)	1 x (Gen3 x8)
Tunneling Engines	2	2	2
Tj (°C)	50	90	87.5

12.6.2.1.3 FM10420 Used in Switch Applications

Although the primary usage model for the FM10420 is in NIC applications, this does not preclude use in small switching or aggregation applications. The available bandwidth in those applications is as much as 400 GbE, and power increases as more bandwidth is used.

Power numbers appropriate for these applications are shown in the following table. Appropriate thermal and power supply design are required for these applications.

Power Corner	I_{AVDD} (A)	I_{VDDF} (A)	I_{VDDS} (A)	Power (W)
Typical	4.1	3.3	8.8	15.1
Max	4.8	6.6	15.8	25.8
TDP[1]				19.4

[1] See Thermal Design Power discussion in Section 12.6.7.

The basic conditions for the FM10420 used in switch applications are:

Condition	Typical	Maximum	TDP
Traffic (desired)	5% 64 B + 75% 128 B	100% 142 B	5% 64 B + 75% 128 B
Ethernet Ports	2 x 100 GbE	2 x 100 GbE	2 x 100 GbE
PCIe Ports	2 x (Gen3 x8)	2 x (Gen3 x8)	2 x (Gen3 x8)
Tunneling Engines	2 @ 60%	2	2 @ 60%
Tj (°C)	50	90	87.5

12.6.2.2 FM10840

Power Corner	I_{AVDD} (A)	I_{VDDF} (A)	I_{VDDS} (A)	Power (W)
Typical	7.8	6.9	14.1	26.8
Max	10.4	15.2	21.9	45.2
TDP[1]				33.8

[1] See Thermal Design Power discussion in Section 12.6.7.

The basic conditions for the FM10840 power numbers are:

Condition	Typical	Maximum	TDP
Traffic	5% 64 B + 75% 128 B	100% 64 B	5% 64 B + 75% 128 B
Ethernet Ports	4 x 100 GbE	1 x 100 GbE + 30 x 10 GbE	4 x 100 GbE
PCIe Ports	4 x (Gen3 x8)	4 x (Gen3 x8)	4 x (Gen3 x8)
Tunneling Engines	2	2	2
Tj (°C)	50	95	90

12.6.2.3 High Current Supply Values Summary

SKU	High Current						High Current Power (W)		
	I_{AVDD} (A)		I_{VDDF} (A)		I_{VDDS} (A)				
	Typical	Max	Typical	Max	Typical	Max	Typical	Max	TDP[1]
FM10840	7.8	10.4	6.9	15.2	14.1	21.9	26.8	45.2	33.8
FM10420 (100 GbE NIC)	4.0	4.7	2.1	4.8	7.6	16.0	12.8	24.1	18.3
FM10420 (2x25 GbE NIC)	1.6	2.2	1.2	3.2	5.2	11.7	7.6	16.1	12.6
FM10420 (Switch)	4.1	4.8	3.3	6.6	8.8	15.8	15.1	25.8	19.4

1. See Thermal Design Power discussion in Section 12.6.7.

12.6.2.4 Supply Conditions for High Current Supply

SKU	Corner	AVDD (V)	VDDF (V)	VDDS (V)	VDD25[1] (V)	Temperature (°C)
FM10840	Typical	1.00	Scaled Nom[2]	Scaled Nom	2.50	50
	Max[3]	1.05	Scaled Nom + 5%	Scaled Nom + 5%	2.75	95
	TDP[4]	1.00	Scaled Nom	Scaled Nom	2.50	90
FM10420	Typical	1.00	Scaled Nom	Scaled Nom	2.50	50
	Max	1.05	Scaled Nom + 5%	Scaled Nom + 5%	2.75	90
	TDP	1.00	Scaled Nom	Scaled Nom	2.50	87.5

1. VDD25 is not listed in “High Current Supply” tables, but is included in Power calculations. Its min/max values are taken from the “Low Current” tables.
2. “Scaled Nom” means that the supply is read from the fusebox and set to the nominal voltage as read.
3. “Max” condition is a calculated “Scaled Nom” voltage value margined high (Vnom + 5%) value on a “worst case” part.
4. Thermal Design Power (TDP) is based on approximately 20% margin over typical at the maximum Tj of the part in question.

12.6.3 Low Current Supply Values

Low Current Supplies have no dependence on traffic and marginal dependence on temperature.

Supply Domain	Symbol	Max Voltage (V)	Typical Current (mA)	Max Current (mA)
PLL Analog Supplies	PLLVDD0	1.05	60	75
PLL Analog Supplies	PLLVDD1	1.05	120	150
LVCMOS and LVPECL Supplies	VDD25	2.75	100	200
LVPECL Termination Supply	AVDD25	2.75	70[1]	70[1]

[1] AVDD25 draw current only when the LVPECL internal terminations are enabled. LVPECL supplies draw approximately 10 mA per active receiver. Values assume all 7 receivers active.

12.6.4 Idle Power

SKU	Idle[1] I_{AVDD} (A)		Idle[1] I_{VDDF} (A)		Idle[1] I_{VDDS} (A)		Idle[1] Power (W)	
	Typical[2]	Max[3]	Typical[2]	Max[3]	Typical[2]	Max[3]	Typical[2]	Max[3]
FM10840	7.8	10.4	1.5	3.6	6.6	15.5	15.3	28.5
FM10420	4.1	4.8	1.0	2.2	5.1	10.0	9.7	16.4

[1] “Idle” is defined as power with the chip out of reset, valid ports up and linked with the clocks running and no traffic.
[2] “Typical” values are calculated for “normal” parts at 50 °C junction temperature.
[3] “Max” values are calculated for “fast” parts at 95 °C junction temperature.

12.6.5 Maximum Current and Dynamic Current Considerations

12.6.5.1 Maximum Currents

Maximum currents are values provided as guidance for system power supply design. These are a combination of leakage currents and dynamic currents. Under worst-case operating conditions and anomalous transient traffic loads, the current needs on the VDDF and VDDS supply domains might include large transients. These transient conditions generally do not arise for longer than a few milliseconds in production environments, if at all. However, to guarantee correct device operation, the system power supply should be designed to deliver the total current represented by these worst-case dynamic currents plus the leakage currents. This is indicated as the “Max” portion of the per-SKU power supply tables.

12.6.5.2 Dynamic Currents

Dynamic currents are the difference transited between the idle current and the maximum current on each part under consistent conditions of temperature and voltage.

Table 12-1 summarizes the expected dynamic currents as "Delta I" shown below on a per-SKU basis.

Table 12-1 Dynamic Currents per SKU

Device	Delta IDDF (A)	Delta IDDS (A)	Worst Case Delta I Total (A)
FM10840	13.3	8.3	18.5[1]
FM10420	4.4	5.9	10.3[2]

[1] In the case of the FM10840, the "Worst Case" Delta I is not the simple sum of the IDDS and IDDF currents. This is because the worst-case conditions for IDDS and IDDF are dependent on traffic conditions, and the worst-case conditions are different for each supply. Thus, the combined worst-case is calculated as the worst-case total current swing possible for the FM10840.

[2] In the case of the FM10420, the supplies are normally tied together. Therefore, the "Worst Case" is the arithmetic sum of the IDDS and IDDF currents

These devices have potentially high dI/dt characteristics. The FM10000 is capable of swinging from idle to maximum current conditions over windows of time as small as 400 ns. For this reason, Intel strongly recommends the use of fast-response, multi-phase power supplies. Best industry design practices should be employed to achieve low impedance power supply networks (below 1 mΩ) and adequate decoupling capacitance. The maximum Delta I expected on both VDDS and VDDF is about 10 A, as can be seen in Table 12-1. The total Delta I on the combined power supplies, in the case of the FM10840, is seen to be about 17 A worst case. These are the numbers that should be used in power supply delivery network (PDN) design. For the FM10420, the numbers are lower due to the lower total cross-sectional bandwidth. For that part, the maximum Delta I is about 4.5 A on VDDS and 6 A on VDDF. The maximum combined Delta I expected on the FM10420 is about 10 A worst case as summarized in the table above.

Based on the Table 12-1, the FM10840 is capable of current variations on the order of 18 amps from idle to full utilization over an interval of approximately 400 ns. The local capacitance needs to support that additional current for the period of time representing the response time of the power supply. Rapid response power supplies (2-5 µs) greatly lower the local decoupling requirements. Power supplies with response times on the order of 30 µs to 60 µs require magnitudes of local bulk capacitance 10X greater than rapid response power supplies.

Over short time spans, instantaneous power might see excursions to the maximum values (previously listed) in response to fluctuations in supply voltage, synchronized bursts of minimum-size packets, and other transitory conditions. However, over thermal time scales, with realistic worst-case traffic and power supply behavior, the steady-state maximum power dissipated by the device is bounded between the "typical" and "maximum" values. It must be noted that power can vary significantly across application types. Furthermore, currents can swing as much as 18 amps from idle to maximum utilization, as stated previously. It is the responsibility of the designer to evaluate the use model of the product under design and install appropriate thermal handling and current handling capabilities.

Customers are encouraged to consult with Intel applications support to ensure that proper consideration is made with regard to transient currents and power supply response times.

12.6.6 Power Versus SKU and Table of Expected Powers

Note: These are preliminary numbers and are derived directly from the expected currents table in Section 12.6.4. It is strongly recommended that early adopters design in ample margin to these numbers.

SKU	Idle[1] Power (W)		Running Power (W)	
	Typical	Max	Typical	Max
FM10840	15.3	28.5	26.8	45.2
FM10420	9.7	16.4	15.1	25.8

[1] "Idle" is defined as power with the chip out of reset, valid ports up and linked with the clocks running and no traffic.

12.6.7 Thermal Design Power

For networking products, the Thermal Design Power (TDP) is not a simple maximum power calculation. This is because the power dissipation of a networking product varies significantly based on the traffic. Unlike a processor, the case where a networking chip goes to the maximum power case and remains at that state for a period of time is extremely unlikely. This is because a networking chip is managing largely uncorrelated data streams. This makes the expected power dissipation dependent on the application at hand. Therefore, a statistical approach to the problem must be taken.

The numbers for TDP in the high current supply tables are based on typical Ethernet and PCIe traffic at the maximum case temperature of 85 °C (see Section 12.7) with a 20% margin. This is sufficient for most cases and is adequate to size the problem. However, users are encouraged to contact Intel with specific application information so that they may have access to the FM10000 Power Calculator, which calculates power based on a specific use model. This allows calculation of powers across the expected use conditions. With this information the design of an appropriately optimized thermal solution can be executed.

12.6.8 Package Power Delivery Network

The FM10000 package simulates to less than 1 mΩ out to 10 MHz. This allows proper power supply management on the scale of the largest voltage transients, which occur on 100 ns time scales.

Figure 12-2 is a graph of the power supply impedance for VDDS and VDDF versus frequency based on HFSS modeling.

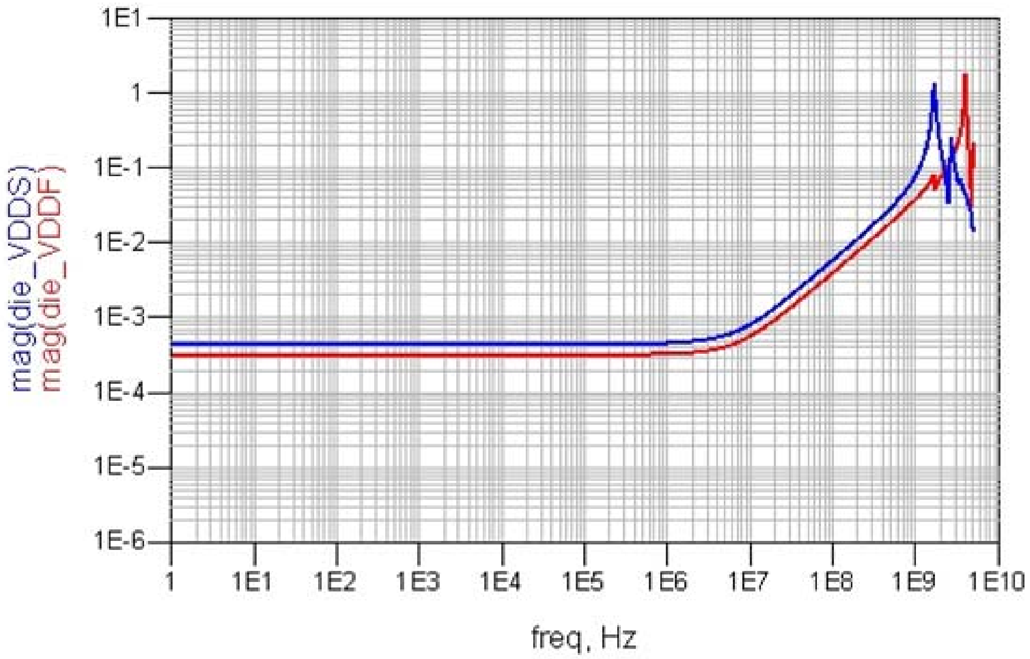


Figure 12-2 Impedance in Ohms vs. Frequency for VDDS and VDDF

12.7 Package Thermal Characteristics

The FM10000 **must** be used with an appropriate thermal solution. In most cases this implies an appropriately designed heat sink.

The FM10000 package is a flip chip cavity up BGA (FCBGA) with a “top hat” style lid. As such, the primary heat path is through the die and then directly through the lid.

The primary thermal path (jc) is a low thermal impedance path. Therefore, the primary thermal solution reduces to an appropriately designed and configured heat sink.

Parameter	Symbol	Value	Units
Junction to Case Thermal Resistance	jc	0.25[1]	°C/W
Junction to Base thermal Resistance (Psi JB)	Psi JB	3.5[2]	°C/W

1. Calculated worst case value based on Flowtherm/Flopack modeling.
2. Informational only. Calculated approximate value based on Flowtherm/Flopack modeling and measurements of similar packages.

12.8 IO Pads

12.8.1 LVCMOS Pads

12.8.1.1 DC Characteristics of LVCMOS PADs

There are three types of LVCMOS PADs: 4 mA, 8 mA, and 12 mA. The characteristics are shown here:

Parameter	Symbol	Min	Typical	Max	Units
Output High Current	IOH		4		mA
			8		mA
			12		mA
Output Low Current	IOL		-4		mA
			-8		mA
			-12		mA
Output High Voltage	VOH	VDD25 - 0.4			V
Output Low Voltage	VOL			0.4	V
Input Current (0V < VPAD < VDD25)	IL	-5		5	µA
Input Current (VDD25 < VPAD < max)		-500		500	µA
Input High Voltage	VIH	1.7		VDD25 + 1.2[1]	V
Input Low Voltage	VIL	-0.3		0.7	V

[1] These IO are compliant with the JESD80 Standard (2.5 V) and the JESD64-A Standard (3.3 V-tolerant with 2.5 V supply)

12.8.1.2 AC Characteristics of LVCMOS Pads

Pad	Parameter	Symbol	Min	Typical	Max	Units
4 mA LVCMOS[1]	Rise Time	Tr	1.5	3.0	4.6	ns
	Fall Time	Tf	1.5	3.0	5.0	ns
8 mA LVCMOS[1]	Rise Time	Tr	1.2	2.0	3.0	ns
	Fall Time	Tf	1.2	2.0	3.2	ns
12 mA LVCMOS[1]	Rise Time	Tr	0.8	1.5	2.5	ns
	Fall Time	Tf	0.8	1.5	2.6	ns

[1] Referenced to a 30 pF load and nominal supply voltages.

12.8.1.3 AC Timing for MDIO Interface

The MDIO electrical timing is shown on the next figure.

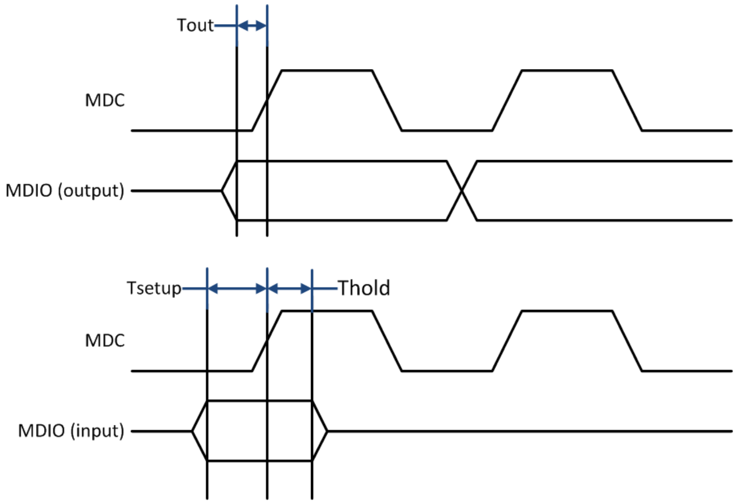


Figure 12-3 MDIO Timing Diagram

Parameter	Symbol	Min	Typical	Max	Units
MDIO output valid to MDC rise	Tout	30			ns
MDIO input setup to MDC rise	Tsetup	30			ns
MDIO input hold to MDC rise	Thold	0			ns

12.8.1.4 LVCMOS Pad Names and Drive Strengths

Pad Name	Direction	Drive	Units
PCIE_RESET_N[3:0]	in	4	mA
1588_RESET_N	in	4	mA
SPI_BOOT	in	4	mA
I2C_SCL	inout	4	mA
I2C_SDA	inout	4	mA
MDC	out	4	mA
MDIO	inout	4	mA
LED_CLK	out	4	mA
LED_EN	out	4	mA
LED_DATA[2:0]	out	4	mA
ADDR[23:0]	in	4	mA
CS_N	in	4	mA
AS_N	in	4	mA

Pad Name	Direction	Drive	Units
RW_N	in	4	mA
EBI_CLK	in	4	mA
1588_PULSE	out	8	mA
INT_N	out	8	mA
GPIO[15:0]	inout	8	mA
SPI_CS_N	out	8	mA
SPI_MOSI	out	8	mA
SPI_MISO	in	8	mA
SPI_IO2	inout	8	mA
SPI_IO3	inout	8	mA
DATA[31:0]	inout	8	mA
DTACK_N	inout	8	mA
1588_SYSTIME[3:0]	out	12	mA
SPI_CLK	out	12	mA
RESET_N	in	N/A	mA
TCK	in	N/A	mA
TMS	in	N/A	mA
TDI	in	N/A	mA
TDO	out	8	mA
TRST_N	in	N/A	mA
TR_TCK	in	N/A	mA
TEST_MODE	in	N/A	mA
TEST_CTRL[2] / TEST_PAD_TRI_L	in	N/A	mA
TEST_CTRL[1] / TEST_ATPG_MODE_L	in	N/A	mA
TEST_CTRL[0] / TEST_CORE_TAP_CTRL_L	in	N/A	mA

12.8.2 LVPECL Pads

12.8.2.1 LVPECL Input Receiver Architecture

The general architecture of the input receiver is illustrated in Figure 12-4:

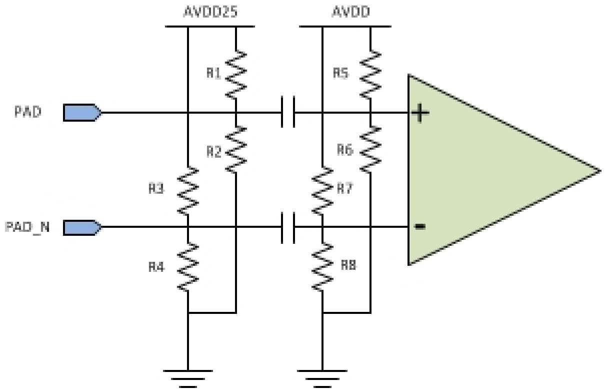


Figure 12-4 Input Receiver

The LVPECL pads have a switchable termination.

- When the input termination is on, there is native DC termination for LVPECL transmitters.
- When the input termination is off, the input impedance is high.

In both cases, the receiver is AC-coupled. However, the input receiver may not be driven with any signal that goes below ground. Thus, zero referenced signals may not be used. On the other hand, a proper source terminated HCSL transmitter may be directly coupled.

12.8.2.2 DC Characteristics of LVPECL Pads

Parameter	Symbol	Min	Typical	Max	Units
Termination Voltage (single-ended)	Vtt	0.4	0.5	0.6	V
Termination Resistance (single-ended)	Rtt	40	50	60	Ohms
Differential Input Voltage	Vdiff	0.4	0.8	1.2	V
Input Common Mode Voltage, Internal Termination Disabled	Vcm	Vdiff/2		AVDD25 - Vdiff/2	V
Input Common Mode Voltage, Internal Termination Enabled	Vcm	0.9	1.2	1.5	V

12.8.2.3 AC Characteristics of LVPECL Pads

Parameter	Symbol	Min	Typical	Max	Units
Input Rise/fall Times (10%-90%)	Trf	40	120	1000	ps
Input Frequency	F	10	-	-	MHz

12.8.2.4 LVPECL Pads

Pad Name	Direction
PCIE_REFCLK	in
PCIE_XREFCLK1	in
PCIE_XREFCLK2	in
PCIE_XREFCLK3	in
PCIE_XREFCLK4	in
1588_REFCLK	in
ETH_REFCLK	in

12.8.3 Clock Observation Pads

These are diagnostic pads not intended to be used in normal operation.

Pad Name	Direction	Load
CLKOUT_A_[P,N]	out	Terminate to 100 ohms differential DC Coupled.
CLKOUT_B_[P,N]	out	Terminate to 100 ohms differential DC Coupled.
CLKOUT_C_[P,N]	out	Terminate to 100 ohms differential DC Coupled.

12.8.4 Thermal Monitoring Diode

Pad Name	Description
THERMAL_IN	Thermal Monitoring Diode - Anode[1]
THERMAL_OUT	Thermal Monitoring Diode - Cathode[1]

[1] Use these two monitoring pins to measure Tj (chip junction temperature).

12.8.5 Thermal Monitoring Diode Characteristics

The thermal monitoring diode behaves as a standard junction diode. It may be connected to a standard series resistance nulling temperature sensing IC, or it may be used directly with appropriate calculations.

The following table can be used to calculate the temperature based on the observed voltage drop at four fixed currents.

Formula:

Temperature = Parameter A - (Parameter B * Voltage)

Current	Parameter A	Parameter B
1 mA	480.5	610.0
200 µA	416.0	558.0
100 µA	392.8	539.3
50 µA	371.2	522.2

Example:

Current = 50 µA, Voltage = 0.72 V

Temperature = (371.2 - (522.2 * 0.72) = -4.8 °C

12.9 Ethernet Specifications

12.9.1 Reference to Electrical Specifications of Supported Modes

The FM10000 is electrically compliant to the following modes as defined in the referenced specifications:

Mode	Lanes	Standard (Notes)	Clause or Section	SerDes Bit Rate (Gb/s)	Encoding
SGMII	1	Cisco		1.25	8b/10b
1000Base-X	1	IEEE Clause 36,37		1.25	8b/10b
1000Base-KX	1	IEEE 802.3ap	Clauses 70, 73	1.25	8b/10b
10GBase-KR	1	IEEE 802.3ap	Clauses 72, 73	10.3125	64b/66b
SFP+ 5M Direct Attach (10GBase-CR)	1	SFF-8431[1]		10.3125	64b/66b
SFP+ SFI Interface (10GBase-SR)	1	SFF-8431[1,2]		10.3125	64b/66b
40GBase-KR4	4	IEEE 802.3ba	Clause 84	10.3125	64b/66b
QSFP 5M Direct Attach (40GBase-CR4)	4	IEEE 802.3ba	Clause 85	10.3125	64b/66b
QSFP PMD Service Interface (40GBase-SR4)	4	IEEE 802.3ba	Clause 86A for XLPPI Interface	10.3125	64b/66b
100GBase-KR4	4	IEEE 802.3bj	Clause 93	25.78125	256b/257b
100GBase-CR4	4	IEEE 802.3bj	Clause 92	25.78125	256b/257b
100GBase-SR4	4	IEEE 802.3bm	CAUI-4 Clause 83D (chip to chip) Clause 83E (chip to module)	25.78125	256b/257b
2.5 GbE non-standard[3]	1	(3)	N/A	3.125	8b/10b
25 GbE non-standard[4]	1	(4)	N/A	25.78125	64b/66b

1. Electrical specifications are SFF-8431. PCS is IEE 802.3 Clause 49
2. The pertinent specifications are host to module electrical specifications.
3. Typically SGMII protocol.
4. Electricals defined as a single lane of corresponding 100 GbE specification (KR, CR, SR).
5. Clause 86 for the optical module. Not relevant to FM10000 Electricals

12.10 EPL RefCLK Specification

All FM10000 reference clock inputs are 2.5-V LVPECL inputs as listed in the previous table. Either internal or external termination may be selected.

To meet required specifications for all Ethernet modes, very low jitter clock sources must be used.

A list of approved sources is being developed. As a guideline, total phase jitter of the clock source should be less than 0.25 ps RMS.

12.11 PCI Express (PCIe)

12.11.1 PCIe Output Specification

FM10000 PCIe transmitters conform to the *PCI Express Base Specification, Revision 3.0, Section 4.3.3*.

12.11.2 PCIe Input Specification

FM10000 PCIe receivers conform to the *PCI Express Base Specification, Revision 3.0, Section 4.3.4*.

12.11.3 PCIe RefCLK Specification

Reference clocks chosen for this application should be compliant to the *PCI Express Base Specification, Revision 3.0, Section 4.3.7*.

The FM10000 does support spread spectrum clocking (SSC) as a single clock domain on all PCIe ports.

12.12 JTAG Interface

The JTAG interface follows standard timing as defined in the *IEEE 1149.1 Standard Test Access Port and Boundary-Scan Architecture, 2001*.

Note: When not using the JTAG interface, either drive the TCK pin with an external clock, or drive the TRST_N pin low. Conversely, when using the JTAG interface assert TRST_N along with chip reset to ensure proper reset of the JTAG interface prior to use.

The FM10000 also supports IEEE 1149.6 (AC JTAG) on all high-speed AC coupled IO pins.

NOTE: ***This page intentionally left blank.***

13.0 Mechanical Specification

13.1 Description

The FM10000 package is a partially depopulated 37.5 mm X 37.5 mm FCBGA package. The lid is a 1 mm thick Cu “Top Hat” style.

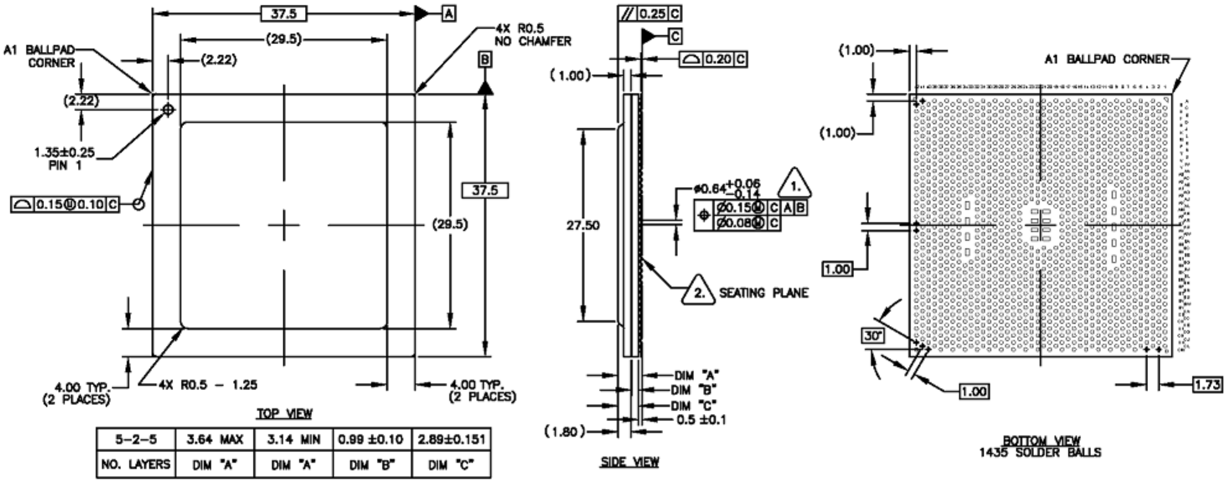
There are top-side and under-side capacitors on the package substrate. Only the bottom side capacitors are exposed. The top side capacitors are beneath the lid.

13.2 Environmental

Specification	Value	Units
Sustained Load[1]	30	PSI (Pounds per square inch)
Recommended PCB Ball Pad	500	um

[1] Sustained Load is pressure to the top of the lid exerted uniformly and perpendicularly to the lid. This is normally the pressure exerted by a socket or heat sink.

13.3 Mechanical Drawings



5-2-5	3.64 MAX	3.14 MIN	0.99 ±0.10	2.89±0.151
NO. LAYERS	DIM "A"	DIM "A"	DIM "B"	DIM "C"

DISCRETE COMPONENTS				
Item No.	X	Y	Z (HEIGHT)	COMPONENT TYPE
1	−10.49	2.85	0.35	0204
2	−10.49	0.61	0.35	0204
3	−10.49	−1.62	0.35	0204
4	−10.49	−3.86	0.35	0204
5	−0.85	1.97	0.35	0204
6	−0.85	0.66	0.35	0204
7	−0.85	−0.66	0.35	0204
8	−0.85	−1.97	0.35	0204
9	0.85	1.97	0.35	0204
10	0.85	0.66	0.35	0204
11	0.85	−0.66	0.35	0204
12	0.85	−1.97	0.35	0204
13	10.49	4.35	0.35	0204
14	10.49	2.05	0.35	0204
15	10.49	−0.25	0.35	0204
16	10.49	−2.55	0.35	0204
17	10.49	−4.85	0.35	0204

NOTES: UNLESS OTHERWISE SPECIFIED

1. DIMENSION IS MEASURED AT THE MAXIMUM SOLDER BALL DIAMETER, PARALLEL TO PRIMARY DATUM C.
2. PRIMARY DATUM C AND SEATING PLANE ARE DEFINED BY THE SPHERICAL CROWNS OF THE SOLDER BALLS.
3. ADDITIONAL PACKAGING, LABELING AND QUALITY REQUIREMENTS ARE AS SPECIFIED ON PART ORDER.

Figure 13-1 Mechanical Drawing

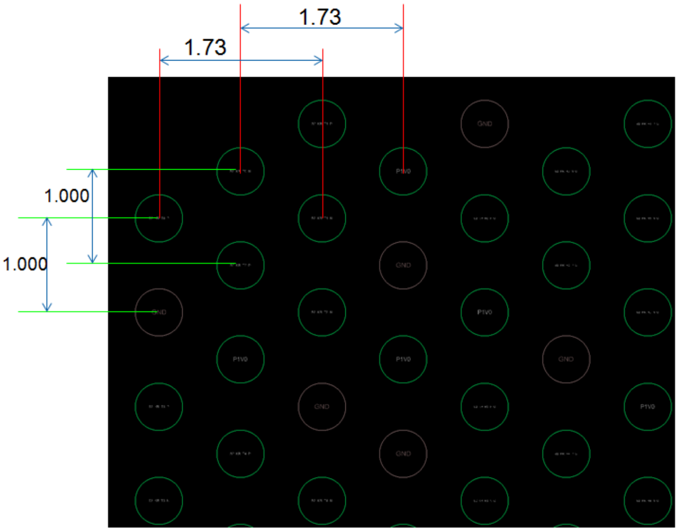


Figure 13-2 Ball Pitch

13.4 Pin Location Color Map

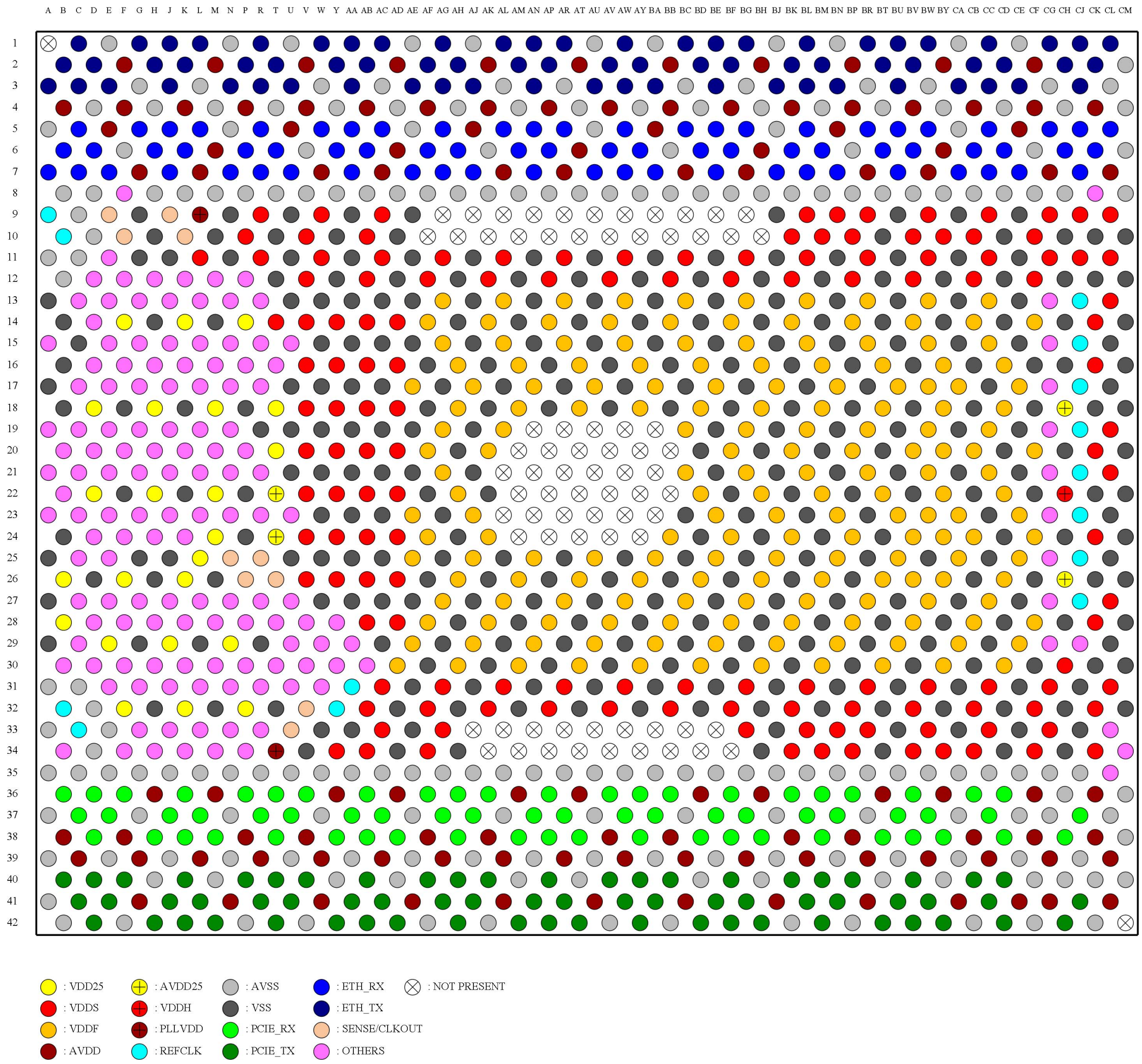


Figure 13-3 Pin Location Color Map (Bottom View)

13.5 Pin Lists

13.5.1 Pin List Ordered by Location

Table 13-1 Pin List Ordered by Location

Pin	Name	Pin	Name	Pin	Name	Pin	Name
A1	—	A3	P0_T1_P	A5	AVSS	A7	P0_R1_P
A9	ETH_REFCLK_P	A11	AVSS	A13	VSS	A15	EBI_CLK
A17	VSS	A19	Reserved-GND	A21	THERMAL_IN	A23	Reserved-NC
A25	VSS	A27	VSS	A29	VSS	A31	AVSS
A33	AVSS	A35	AVSS	A37	AVSS	A39	AVSS
A41	AVSS	B2	P0_T0_N	B4	AVDD	B6	P0_R0_N
B8	AVSS	B10	ETH_REFCLK_N	B12	AVSS	B14	VSS
B16	VSS	B18	VSS	B20	ADDR0	B22	THERMAL_OUT
B24	VSS	B26	VDD25	B28	VDD25	B30	RESET_N
B32	PCIE_REFCLK_P	B34	REF0	B36	PCIE0_R1_P	B38	AVDD
B40	PCIE0_T1_P	B42	AVSS	C1	P0_T0_P	C3	P0_T1_N
C5	P0_R0_P	C7	P0_R1_N	C9	AVSS	C11	AVSS
C13	Reserved-PUVDD25	C15	VSS	C17	DATA19	C19	DATA20
C21	ADDR1	C23	ADDR14	C25	ADDR22	C27	GPIO0
C29	SPI_CLK	C31	AVSS	C33	PCIE_REFCLK_N	C35	AVSS
C37	PCIE0_R0_N	C39	AVDD	C41	PCIE0_T0_N	D2	P0_T2_P
D4	AVSS	D6	P0_R2_P	D8	AVSS	D10	AVSS
D12	Reserved-PUVDD25	D14	Reserved-PUVDD25	D16	DATA8	D18	VDD25
D20	ADDR2	D22	VDD25	D24	ADDR15	D26	VSS
D28	GPIO1	D30	IEEE1588_RESET_N	D32	AVSS	D34	AVSS
D36	PCIE0_R1_N	D38	PCIE0_R0_P	D40	PCIE0_T1_N	D42	PCIE0_T0_P
E1	AVSS	E3	P0_T2_N	E5	AVDD	E7	P0_R2_N
E9	CLKOUT_A_P	E11	Reserved-PD	E13	DATA0	E15	DATA7
E17	DATA21	E19	DATA22	E21	ADDR3	E23	ADDR16
E25	ADDR23	E27	GPIO2	E29	VDD25	E31	IEEE1588_SYSTIME0
E33	AVSS	E35	AVSS	E37	PCIE0_R2_P	E39	AVSS
E41	PCIE0_T2_P	F2	AVDD	F4	AVDD	F6	AVSS
F8	REF2	F10	CLKOUT_A_N	F12	DATA1	F14	VDD25
F16	DATA10	F18	VSS	F20	ADDR4	F22	VSS
F24	ADDR17	F26	VDD25	F28	GPIO3	F30	IEEE1588_SYSTIME1
F32	VDD25	F34	PCIE_RESET_N3	F36	PCIE0_R2_N	F38	AVDD
F40	PCIE0_T2_N	F42	AVSS	G1	P0_T3_P	G3	AVSS
G5	P0_R3_P	G7	AVDD	G9	VSS	G11	VSS

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
G13	DATA2	G15	DATA9	G17	DATA23	G19	DATA24
G21	ADDR5	G23	ADDR18	G25	VSS	G27	GPIO4
G29	VSS	G31	SPI_IO2	G33	SPI_CS_N	G35	AVSS
G37	AVSS	G39	AVDD	G41	AVDD	H2	P1_T0_P
H4	AVSS	H6	P1_R0_P	H8	AVSS	H10	VSS
H12	DATA3	H14	VSS	H16	DATA12	H18	VDD25
H20	ADDR6	H22	VDD25	H24	ADDR19	H26	VSS
H28	GPIO5	H30	IEEE1588_SYSTIME2	H32	VSS	H34	PCIE_RESET_N2
H36	AVDD	H38	PCIE0_R3_P	H40	AVSS	H42	PCIE0_T3_P
J1	P0_T3_N	J3	P1_T0_N	J5	P0_R3_N	J7	P1_R0_N
J9	CLKOUT_B_P	J11	VSS	J13	DATA4	J15	DATA11
J17	DATA25	J19	DATA26	J21	ADDR7	J23	ADDR20
J25	VSS	J27	GPIO6	J29	VDD25	J31	SPI_BOOT
J33	PCIE_RESET_N8	J35	AVSS	J37	PCIE0_R4_P	J39	AVSS
J41	PCIE0_T4_P	K2	P1_T1_N	K4	AVDD	K6	P1_R1_N
K8	AVSS	K10	CLKOUT_B_N	K12	DATA5	K14	VDD25
K16	DATA14	K18	VSS	K20	ADDR8	K22	VSS
K24	ADDR21	K26	VDD25	K28	GPIO7	K30	IEEE1588_SYSTIME3
K32	VDD25	K34	PCIE_RESET_N1	K36	PCIE0_R4_N	K38	PCIE0_R3_N
K40	PCIE0_T4_N	K42	PCIE0_T3_N	L1	P1_T1_P	L3	AVSS
L5	P1_R1_P	L7	AVDD	L9	PLLVDD1	L11	VDDS
L13	DATA6	L15	DATA13	L17	DATA27	L19	DATA28
L21	ADDR9	L23	DTACK_N	L25	VDD25	L27	GPIO8
L29	VSS	L31	SPI_MOSI	L33	PCIE_RESET_N7	L35	AVSS
L37	PCIE0_R5_N	L39	AVDD	L41	PCIE0_T5_N	M2	AVDD
M4	AVSS	M6	AVDD	M8	AVSS	M10	VSS
M12	TCK	M14	VSS	M16	DATA16	M18	VDD25
M20	ADDR10	M22	VDD25	M24	VDD25	M26	VSS
M28	GPIO9	M30	INT_N	M32	VSS	M34	PCIE_RESET_N0
M36	AVDD	M38	PCIE0_R5_P	M40	AVSS	M42	PCIE0_T5_P
N1	AVSS	N3	P1_T3_P	N5	AVSS	N7	P1_R3_P
N9	PLLVSS1	N11	VSS	N13	Reserved-NC	N15	DATA15
N17	DATA29	N19	DATA30	N21	ADDR11	N23	AS_N
N25	SENSE_VDDS	N27	GPIO10	N29	VDD25	N31	SPI_MISO
N33	PCIE_RESET_N6	N35	AVSS	N37	AVSS	N39	AVSS
N41	AVDD	P2	P1_T2_N	P4	AVDD	P6	P1_R2_N
P8	AVSS	P10	VDDS	P12	TMS	P14	VDD25
P16	DATA18	P18	VSS	P20	ADDR12	P22	VSS

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
P24	VSS	P26	SENSE_VSSS	P28	GPIO11	P30	LED_EN
P32	VDD25	P34	PCIE_RESET_N4	P36	PCIE0_R7_P	P38	AVDD
P40	PCIE0_T7_P	P42	AVSS	R1	P1_T2_P	R3	P1_T3_N
R5	P1_R2_P	R7	P1_R3_N	R9	VDDS	R11	VDDS
R13	TDI	R15	DATA17	R17	DATA31	R19	VSS
R21	ADDR13	R23	RW_N	R25	SENSE_VDDF	R27	GPIO12
R29	VSS	R31	SPI_IO3	R33	PCIE_RESET_N5	R35	AVSS
R37	PCIE0_R6_N	R39	AVDD	R41	PCIE0_T6_N	T2	P2_T0_P
T4	AVSS	T6	P2_R0_P	T8	AVSS	T10	VSS
T12	VSS	T14	VDDS	T16	TDO	T18	VDD25
T20	VDD25	T22	AVDD25	T24	AVDD25	T26	SENSE_VSSF
T28	GPIO13	T30	LED_CLK	T32	VSS	T34	PLLVDD0
T36	PCIE0_R7_N	T38	PCIE0_R6_P	T40	PCIE0_T7_N	T42	PCIE0_T6_P
U1	AVSS	U3	P2_T0_N	U5	AVDD	U7	P2_R0_N
U9	VSS	U11	VSS	U13	VSS	U15	TRST_N
U17	VSS	U19	VSS	U21	VSS	U23	CS_N
U25	VSS	U27	GPIO14	U29	I2C_SDA	U31	LED_DATA0
U33	CLKOUT_C_P	U35	AVSS	U37	PCIE1_R0_P	U39	AVSS
U41	PCIE1_T0_P	V2	AVDD	V4	AVDD	V6	AVSS
V8	AVSS	V10	VDDS	V12	VDDS	V14	VDDS
V16	VDDS	V18	VDDS	V20	VDDS	V22	VDDS
V24	VDDS	V26	VDDS	V28	GPIO15	V30	LED_DATA1
V32	CLKOUT_C_N	V34	PLLVSS0	V36	PCIE1_R0_N	V38	AVDD
V40	PCIE1_T0_N	V42	AVSS	W1	P2_T3_P	W3	AVSS
W5	P2_R3_P	W7	AVDD	W9	VDDS	W11	VDDS
W13	VSS	W15	VSS	W17	VSS	W19	VSS
W21	VSS	W23	VSS	W25	VSS	W27	VSS
W29	MDC	W31	LED_DATA2	W33	VSS	W35	AVSS
W37	AVSS	W39	AVDD	W41	AVDD	Y2	P2_T1_P
Y4	AVSS	Y6	P2_R1_P	Y8	AVSS	Y10	VSS
Y12	VSS	Y14	VDDS	Y16	VDDS	Y18	VDDS
Y20	VDDS	Y22	VDDS	Y24	VDDS	Y26	VDDS
Y28	MDIO	Y30	IEEE1588_PULSE0	Y32	IEEE1588_REFCLK_P	Y34	VDDS
Y36	AVDD	Y38	PCIE1_R1_P	Y40	AVSS	Y42	PCIE1_T1_P
AA1	P2_T3_N	AA3	P2_T1_N	AA5	P2_R3_N	AA7	P2_R1_N
AA9	VSS	AA11	VSS	AA13	VSS	AA15	VSS
AA17	VSS	AA19	VSS	AA21	VSS	AA23	VSS
AA25	VSS	AA27	VSS	AA29	I2C_SCL	AA31	IEEE1588_REFCLK_N

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
AA33	VSS	AA35	AVSS	AA37	PCIE1_R2_P	AA39	AVSS
AA41	PCIE1_T2_P	AB2	P2_T2_P	AB4	AVDD	AB6	P2_R2_P
AB8	AVSS	AB10	VDDS	AB12	VDDS	AB14	VDDS
AB16	VDDS	AB18	VDDS	AB20	VDDS	AB22	VDDS
AB24	VDDS	AB26	VDDS	AB28	VDDS	AB30	IEEE1588_PULSE1
AB32	VDDS	AB34	VDDS	AB36	PCIE1_R2_N	AB38	PCIE1_R1_N
AB40	PCIE1_T2_N	AB42	PCIE1_T1_N	AC1	P2_T2_N	AC3	AVSS
AC5	P2_R2_N	AC7	AVDD	AC9	VDDS	AC11	VDDS
AC13	VSS	AC15	VSS	AC17	VSS	AC19	VSS
AC21	VSS	AC23	VSS	AC25	VSS	AC27	VSS
AC29	VSS	AC31	VDDS	AC33	VDDS	AC35	AVSS
AC37	PCIE1_R3_N	AC39	AVDD	AC41	PCIE1_T3_N	AD2	AVDD
AD4	AVSS	AD6	AVDD	AD8	AVSS	AD10	VSS
AD12	VSS	AD14	VDDS	AD16	VDDS	AD18	VDDS
AD20	VDDS	AD22	VDDS	AD24	VDDS	AD26	VDDS
AD28	VDDS	AD30	VDDF	AD32	VSS	AD34	VSS
AD36	AVDD	AD38	PCIE1_R3_P	AD40	AVSS	AD42	PCIE1_T3_P
AE1	AVSS	AE3	P3_T1_P	AE5	AVSS	AE7	P3_R1_P
AE9	VSS	AE11	VSS	AE13	VSS	AE15	VSS
AE17	VDDF	AE19	VSS	AE21	VSS	AE23	VDDF
AE25	VDDF	AE27	VSS	AE29	VDDF	AE31	VSS
AE33	VSS	AE35	AVSS	AE37	AVSS	AE39	AVSS
AE41	AVDD	AF2	P3_T0_N	AF4	AVDD	AF6	P3_R0_N
AF8	AVSS	AF10	—	AF12	VDDS	AF14	VDDF
AF16	VSS	AF18	VSS	AF20	VDDF	AF22	VSS
AF24	VDDF	AF26	VSS	AF28	VDDF	AF30	VSS
AF32	VDDS	AF34	VDDS	AF36	PCIE1_R5_P	AF38	AVDD
AF40	PCIE1_T5_P	AF42	AVSS	AG1	P3_T0_P	AG3	P3_T1_N
AG5	P3_R0_P	AG7	P3_R1_N	AG9	—	AG11	VDDS
AG13	VDDF	AG15	VDDF	AG17	VSS	AG19	VDDF
AG21	VDDF	AG23	VSS	AG25	VSS	AG27	VDDF
AG29	VSS	AG31	VDDS	AG33	VDDS	AG35	AVSS
AG37	PCIE1_R4_N	AG39	AVDD	AG41	PCIE1_T4_N	AH2	P3_T2_P
AH4	AVSS	AH6	P3_R2_P	AH8	AVSS	AH10	—
AH12	VSS	AH14	VSS	AH16	VDDF	AH18	VDDF
AH20	VSS	AH22	VDDF	AH24	VSS	AH26	VDDF
AH28	VSS	AH30	VDDF	AH32	VSS	AH34	VSS
AH36	PCIE1_R5_N	AH38	PCIE1_R4_P	AH40	PCIE1_T5_N	AH42	PCIE1_T4_P

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
AJ1	AVSS	AJ3	P3_T2_N	AJ5	AVDD	AJ7	P3_R2_N
AJ9	—	AJ11	VSS	AJ13	VSS	AJ15	VSS
AJ17	VDDF	AJ19	VSS	AJ21	VSS	AJ23	VDDF
AJ25	VDDF	AJ27	VSS	AJ29	VDDF	AJ31	VSS
AJ33	—	AJ35	AVSS	AJ37	PCIE1_R6_P	AJ39	AVSS
AJ41	PCIE1_T6_P	AK2	AVDD	AK4	AVDD	AK6	AVSS
AK8	AVSS	AK10	—	AK12	VDDS	AK14	VDDF
AK16	VSS	AK18	VSS	AK20	VDDF	AK22	VSS
AK24	VDDF	AK26	VSS	AK28	VDDF	AK30	VSS
AK32	VDDS	AK34	—	AK36	PCIE1_R6_N	AK38	AVDD
AK40	PCIE1_T6_N	AK42	AVSS	AL1	P3_T3_P	AL3	AVSS
AL5	P3_R3_P	AL7	AVDD	AL9	—	AL11	VDDS
AL13	VDDF	AL15	VDDF	AL17	VSS	AL19	VDDF
AL21	—	AL23	—	AL25	VSS	AL27	VDDF
AL29	VSS	AL31	VDDS	AL33	—	AL35	AVSS
AL37	AVSS	AL39	AVDD	AL41	AVDD	AM2	P4_T0_P
AM4	AVSS	AM6	P4_R0_P	AM8	AVSS	AM10	—
AM12	VSS	AM14	VSS	AM16	VDDF	AM18	VDDF
AM20	—	AM22	—	AM24	—	AM26	VDDF
AM28	VSS	AM30	VDDF	AM32	VSS	AM34	—
AM36	AVDD	AM38	PCIE1_R7_P	AM40	AVSS	AM42	PCIE1_T7_P
AN1	P3_T3_N	AN3	P4_T0_N	AN5	P3_R3_N	AN7	P4_R0_N
AN9	—	AN11	VSS	AN13	VSS	AN15	VSS
AN17	VDDF	AN19	—	AN21	—	AN23	—
AN25	VDDF	AN27	VSS	AN29	VDDF	AN31	VSS
AN33	—	AN35	AVSS	AN37	PCIE2_R0_P	AN39	AVSS
AN41	PCIE2_T0_P	AP2	P4_T1_N	AP4	AVDD	AP6	P4_R1_N
AP8	AVSS	AP10	—	AP12	VDDS	AP14	VDDF
AP16	VSS	AP18	VSS	AP20	—	AP22	—
AP24	—	AP26	VSS	AP28	VDDF	AP30	VSS
AP32	VDDS	AP34	—	AP36	PCIE2_R0_N	AP38	PCIE1_R7_N
AP40	PCIE2_T0_N	AP42	PCIE1_T7_N	AR1	P4_T1_P	AR3	AVSS
AR5	P4_R1_P	AR7	AVDD	AR9	—	AR11	VDDS
AR13	VDDF	AR15	VDDF	AR17	VSS	AR19	—
AR21	—	AR23	—	AR25	VSS	AR27	VDDF
AR29	VSS	AR31	VDDS	AR33	—	AR35	AVSS
AR37	PCIE2_R1_N	AR39	AVDD	AR41	PCIE2_T1_N	AT2	AVDD
AT4	AVSS	AT6	AVDD	AT8	AVSS	AT10	—

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
AT12	VSS	AT14	VSS	AT16	VDDF	AT18	VDDF
AT20	—	AT22	—	AT24	—	AT26	VDDF
AT28	VSS	AT30	VDDF	AT32	VSS	AT34	—
AT36	AVDD	AT38	PCIE2_R1_P	AT40	AVSS	AT42	PCIE2_T1_P
AU1	AVSS	AU3	P4_T3_P	AU5	AVSS	AU7	P4_R3_P
AU9	—	AU11	VSS	AU13	VSS	AU15	VSS
AU17	VDDF	AU19	—	AU21	—	AU23	—
AU25	VDDF	AU27	VSS	AU29	VDDF	AU31	VSS
AU33	—	AU35	AVSS	AU37	AVSS	AU39	AVSS
AU41	AVDD	AV2	P4_T2_N	AV4	AVDD	AV6	P4_R2_N
AV8	AVSS	AV10	—	AV12	VDDS	AV14	VDDF
AV16	VSS	AV18	VSS	AV20	—	AV22	—
AV24	—	AV26	VSS	AV28	VDDF	AV30	VSS
AV32	VDDS	AV34	—	AV36	PCIE2_R3_P	AV38	AVDD
AV40	PCIE2_T3_P	AV42	AVSS	AW1	P4_T2_P	AW3	P4_T3_N
AW5	P4_R2_P	AW7	P4_R3_N	AW9	—	AW11	VDDS
AW13	VDDF	AW15	VDDF	AW17	VSS	AW19	—
AW21	—	AW23	—	AW25	VSS	AW27	VDDF
AW29	VSS	AW31	VDDS	AW33	—	AW35	AVSS
AW37	PCIE2_R2_N	AW39	AVDD	AW41	PCIE2_T2_N	AY2	P5_T0_P
AY4	AVSS	AY6	P5_R0_P	AY8	AVSS	AY10	—
AY12	VSS	AY14	VSS	AY16	VDDF	AY18	VDDF
AY20	—	AY22	—	AY24	—	AY26	VDDF
AY28	VSS	AY30	VDDF	AY32	VSS	AY34	—
AY36	PCIE2_R3_N	AY38	PCIE2_R2_P	AY40	PCIE2_T3_N	AY42	PCIE2_T2_P
BA1	AVSS	BA3	P5_T0_N	BA5	AVDD	BA7	P5_R0_N
BA9	—	BA11	VSS	BA13	VSS	BA15	VSS
BA17	VDDF	BA19	—	BA21	—	BA23	—
BA25	VDDF	BA27	VSS	BA29	VDDF	BA31	VSS
BA33	—	BA35	AVSS	BA37	PCIE2_R4_P	BA39	AVSS
BA41	PCIE2_T4_P	BB2	AVDD	BB4	AVDD	BB6	AVSS
BB8	AVSS	BB10	—	BB12	VDDS	BB14	VDDF
BB16	VSS	BB18	VSS	BB20	—	BB22	—
BB24	VDDF	BB26	VSS	BB28	VDDF	BB30	VSS
BB32	VDDS	BB34	—	BB36	PCIE2_R4_N	BB38	AVDD
BB40	PCIE2_T4_N	BB42	AVSS	BC1	P5_T3_P	BC3	AVSS
BC5	P5_R3_P	BC7	AVDD	BC9	—	BC11	VDDS
BC13	VDDF	BC15	VDDF	BC17	VSS	BC19	VDDF

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
BC21	VDDF	BC23	VSS	BC25	VSS	BC27	VDDF
BC29	VSS	BC31	VDDS	BC33	—	BC35	AVSS
BC37	AVSS	BC39	AVDD	BC41	AVDD	BD2	P5_T1_P
BD4	AVSS	BD6	P5_R1_P	BD8	AVSS	BD10	—
BD12	VSS	BD14	VSS	BD16	VDDF	BD18	VDDF
BD20	VSS	BD22	VDDF	BD24	VSS	BD26	VDDF
BD28	VSS	BD30	VDDF	BD32	VSS	BD34	—
BD36	AVDD	BD38	PCIE2_R5_P	BD40	AVSS	BD42	PCIE2_T5_P
BE1	P5_T3_N	BE3	P5_T1_N	BE5	P5_R3_N	BE7	P5_R1_N
BE9	—	BE11	VSS	BE13	VSS	BE15	VSS
BE17	VDDF	BE19	VSS	BE21	VSS	BE23	VDDF
BE25	VDDF	BE27	VSS	BE29	VDDF	BE31	VSS
BE33	—	BE35	AVSS	BE37	PCIE2_R6_P	BE39	AVSS
BE41	PCIE2_T6_P	BF2	P5_T2_P	BF4	AVDD	BF6	P5_R2_P
BF8	AVSS	BF10	—	BF12	VDDS	BF14	VDDF
BF16	VSS	BF18	VSS	BF20	VDDF	BF22	VSS
BF24	VDDF	BF26	VSS	BF28	VDDF	BF30	VSS
BF32	VDDS	BF34	—	BF36	PCIE2_R6_N	BF38	PCIE2_R5_N
BF40	PCIE2_T6_N	BF42	PCIE2_T5_N	BG1	P5_T2_N	BG3	AVSS
BG5	P5_R2_N	BG7	AVDD	BG9	—	BG11	VDDS
BG13	VDDF	BG15	VDDF	BG17	VSS	BG19	VDDF
BG21	VDDF	BG23	VSS	BG25	VSS	BG27	VDDF
BG29	VSS	BG31	VDDS	BG33	VDDS	BG35	AVSS
BG37	PCIE2_R7_N	BG39	AVDD	BG41	PCIE2_T7_N	BH2	AVDD
BH4	AVSS	BH6	AVDD	BH8	AVSS	BH10	—
BH12	VSS	BH14	VSS	BH16	VDDF	BH18	VDDF
BH20	VSS	BH22	VDDF	BH24	VSS	BH26	VDDF
BH28	VSS	BH30	VDDF	BH32	VSS	BH34	VSS
BH36	AVDD	BH38	PCIE2_R7_P	BH40	AVSS	BH42	PCIE2_T7_P
BJ1	AVSS	BJ3	P6_T1_P	BJ5	AVSS	BJ7	P6_R1_P
BJ9	VSS	BJ11	VSS	BJ13	VSS	BJ15	VSS
BJ17	VDDF	BJ19	VSS	BJ21	VSS	BJ23	VDDF
BJ25	VDDF	BJ27	VSS	BJ29	VDDF	BJ31	VSS
BJ33	VSS	BJ35	AVSS	BJ37	AVSS	BJ39	AVSS
BJ41	AVDD	BK2	P6_T0_N	BK4	AVDD	BK6	P6_R0_N
BK8	AVSS	BK10	VDDS	BK12	VDDS	BK14	VDDF
BK16	VSS	BK18	VSS	BK20	VDDF	BK22	VSS
BK24	VDDF	BK26	VSS	BK28	VDDF	BK30	VSS

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
BK32	VDDS	BK34	VDDS	BK36	PCIE3_R1_P	BK38	AVDD
BK40	PCIE3_T1_P	BK42	AVSS	BL1	P6_T0_P	BL3	P6_T1_N
BL5	P6_R0_P	BL7	P6_R1_N	BL9	VDDS	BL11	VDDS
BL13	VDDF	BL15	VDDF	BL17	VSS	BL19	VDDF
BL21	VDDF	BL23	VSS	BL25	VSS	BL27	VDDF
BL29	VSS	BL31	VDDS	BL33	VDDS	BL35	AVSS
BL37	PCIE3_R0_N	BL39	AVDD	BL41	PCIE3_T0_N	BM2	P6_T2_P
BM4	AVSS	BM6	P6_R2_P	BM8	AVSS	BM10	VDDS
BM12	VSS	BM14	VSS	BM16	VDDF	BM18	VDDF
BM20	VSS	BM22	VDDF	BM24	VSS	BM26	VDDF
BM28	VSS	BM30	VDDF	BM32	VSS	BM34	VDDS
BM36	PCIE3_R1_N	BM38	PCIE3_R0_P	BM40	PCIE3_T1_N	BM42	PCIE3_T0_P
BN1	AVSS	BN3	P6_T2_N	BN5	AVDD	BN7	P6_R2_N
BN9	VDDS	BN11	VSS	BN13	VSS	BN15	VSS
BN17	VDDF	BN19	VSS	BN21	VSS	BN23	VDDF
BN25	VDDF	BN27	VSS	BN29	VDDF	BN31	VSS
BN33	VDDS	BN35	AVSS	BN37	PCIE3_R2_P	BN39	AVSS
BN41	PCIE3_T2_P	BP2	AVDD	BP4	AVDD	BP6	AVSS
BP8	AVSS	BP10	VDDS	BP12	VDDS	BP14	VDDF
BP16	VSS	BP18	VSS	BP20	VDDF	BP22	VSS
BP24	VDDF	BP26	VSS	BP28	VDDF	BP30	VSS
BP32	VDDS	BP34	VDDS	BP36	PCIE3_R2_N	BP38	AVDD
BP40	PCIE3_T2_N	BP42	AVSS	BR1	P6_T3_P	BR3	AVSS
BR5	P6_R3_P	BR7	AVDD	BR9	VDDS	BR11	VDDS
BR13	VDDF	BR15	VDDF	BR17	VSS	BR19	VDDF
BR21	VDDF	BR23	VSS	BR25	VSS	BR27	VDDF
BR29	VSS	BR31	VDDS	BR33	VDDS	BR35	AVSS
BR37	AVSS	BR39	AVDD	BR41	AVDD	BT2	P7_T0_P
BT4	AVSS	BT6	P7_R0_P	BT8	AVSS	BT10	VSS
BT12	VSS	BT14	VSS	BT16	VDDF	BT18	VDDF
BT20	VSS	BT22	VDDF	BT24	VSS	BT26	VDDF
BT28	VSS	BT30	VDDF	BT32	VSS	BT34	VSS
BT36	AVDD	BT38	PCIE3_R3_P	BT40	AVSS	BT42	PCIE3_T3_P
BU1	P6_T3_N	BU3	P7_T0_N	BU5	P6_R3_N	BU7	P7_R0_N
BU9	VSS	BU11	VSS	BU13	VSS	BU15	VSS
BU17	VDDF	BU19	VSS	BU21	VSS	BU23	VDDF
BU25	VDDF	BU27	VSS	BU29	VDDF	BU31	VSS
BU33	VSS	BU35	AVSS	BU37	PCIE3_R4_P	BU39	AVSS

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
BU41	PCIE3_T4_P	BV2	P7_T1_N	BV4	AVDD	BV6	P7_R1_N
BV8	AVSS	BV10	VDDS	BV12	VDDS	BV14	VDDF
BV16	VSS	BV18	VSS	BV20	VDDF	BV22	VSS
BV24	VDDF	BV26	VDDF	BV28	VDDF	BV30	VSS
BV32	VDDS	BV34	VDDS	BV36	PCIE3_R4_N	BV38	PCIE3_R3_N
BV40	PCIE3_T4_N	BV42	PCIE3_T3_N	BW1	P7_T1_P	BW3	AVSS
BW5	P7_R1_P	BW7	AVDD	BW9	VDDS	BW11	VDDS
BW13	VDDF	BW15	VDDF	BW17	VDDF	BW19	VDDF
BW21	VDDF	BW23	VDDF	BW25	VSS	BW27	VDDF
BW29	VSS	BW31	VDDS	BW33	VDDS	BW35	AVSS
BW37	PCIE3_R5_N	BW39	AVDD	BW41	PCIE3_T5_N	BY2	AVDD
BY4	AVSS	BY6	AVDD	BY8	AVSS	BY10	VDDS
BY12	VSS	BY14	VSS	BY16	VDDF	BY18	VDDF
BY20	VDDF	BY22	VDDF	BY24	VDDF	BY26	VDDF
BY28	VSS	BY30	VDDF	BY32	VSS	BY34	VDDS
BY36	AVDD	BY38	PCIE3_R5_P	BY40	AVSS	BY42	PCIE3_T5_P
CA1	AVSS	CA3	P7_T3_P	CA5	AVSS	CA7	P7_R3_P
CA9	VSS	CA11	VSS	CA13	VSS	CA15	VSS
CA17	VDDF	CA19	VSS	CA21	VSS	CA23	VDDF
CA25	VDDF	CA27	VSS	CA29	VDDF	CA31	VSS
CA33	VSS	CA35	AVSS	CA37	AVSS	CA39	AVSS
CA41	AVDD	CB2	P7_T2_N	CB4	AVDD	CB6	P7_R2_N
CB8	AVSS	CB10	VDDS	CB12	VDDS	CB14	VDDF
CB16	VSS	CB18	VSS	CB20	VDDF	CB22	VSS
CB24	VDDF	CB26	VSS	CB28	VDDF	CB30	VSS
CB32	VDDS	CB34	VDDS	CB36	PCIE3_R7_P	CB38	AVDD
CB40	PCIE3_T7_P	CB42	AVSS	CC1	P7_T2_P	CC3	P7_T3_N
CC5	P7_R2_P	CC7	P7_R3_N	CC9	VDDS	CC11	VDDS
CC13	VDDF	CC15	VDDF	CC17	VSS	CC19	VDDF
CC21	VDDF	CC23	VSS	CC25	VSS	CC27	VDDF
CC29	VSS	CC31	VDDS	CC33	VDDS	CC35	AVSS
CC37	PCIE3_R6_N	CC39	AVDD	CC41	PCIE3_T6_N	CD2	P8_T0_P
CD4	AVSS	CD6	P8_R0_P	CD8	AVSS	CD10	VSS
CD12	VSS	CD14	VSS	CD16	VDDF	CD18	VDDF
CD20	VSS	CD22	VDDF	CD24	VSS	CD26	VDDF
CD28	VSS	CD30	VDDF	CD32	VSS	CD34	VSS
CD36	PCIE3_R7_N	CD38	PCIE3_R6_P	CD40	PCIE3_T7_N	CD42	PCIE3_T6_P
CE1	AVSS	CE3	P8_T0_N	CE5	AVDD	CE7	P8_R0_N

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
CE9	VSS	CE11	VSS	CE13	VSS	CE15	VSS
CE17	VDDF	CE19	VSS	CE21	VSS	CE23	VDDF
CE25	VDDF	CE27	VSS	CE29	VDDF	CE31	VSS
CE33	VSS	CE35	AVSS	CE37	AVSS	CE39	AVSS
CE41	AVDD	CF2	AVDD	CF4	AVDD	CF6	AVSS
CF8	AVSS	CF10	VDDS	CF12	VDDS	CF14	VDDF
CF16	VSS	CF18	VSS	CF20	VDDF	CF22	VSS
CF24	VDDF	CF26	VSS	CF28	VDDF	CF30	VSS
CF32	VDDS	CF34	VDDS	CF36	AVDD	CF38	AVDD
CF40	AVSS	CF42	AVSS	CG1	P8_T3_P	CG3	AVSS
CG5	P8_R3_P	CG7	AVDD	CG9	VDDS	CG11	VDDS
CG13	Reserved-NC	CG15	Reserved-NC	CG17	Reserved-NC	CG19	Reserved-NC
CG21	Reserved-NC	CG23	Reserved-NC	CG25	Reserved-NC	CG27	Reserved-NC
CG29	Reserved-NC	CG31	VDDS	CG33	VDDS	CG35	AVSS
CG37	AVSS	CG39	AVDD	CG41	AVDD	CH2	P8_T1_P
CH4	AVSS	CH6	P8_R1_P	CH8	AVSS	CH10	VSS
CH12	VSS	CH14	VSS	CH16	VSS	CH18	AVDD25
CH20	VSS	CH22	Reserved-VDDS	CH24	VSS	CH26	AVDD25
CH28	VSS	CH30	VDDS	CH32	VSS	CH34	VSS
CH36	AVSS	CH38	PCIE4_R0_P	CH40	AVSS	CH42	PCIE4_T0_P
CJ1	P8_T3_N	CJ3	P8_T1_N	CJ5	P8_R3_N	CJ7	P8_R1_N
CJ9	VDDS	CJ11	VDDS	CJ13	PCIE_XREFCLK0_P	CJ15	PCIE_XREFCLK0_N
CJ17	PCIE_XREFCLK1_P	CJ19	PCIE_XREFCLK1_N	CJ21	PCIE_XREFCLK2_P	CJ23	PCIE_XREFCLK2_N
CJ25	PCIE_XREFCLK3_P	CJ27	PCIE_XREFCLK3_N	CJ29	Reserved-NC	CJ31	VSS
CJ33	VSS	CJ35	AVSS	CJ37	PCIE4_R0_N	CJ39	AVSS
CJ41	PCIE4_T0_N	CK2	P8_T2_P	CK4	AVDD	CK6	P8_R2_P
CK8	REF3	CK10	VSS	CK12	VSS	CK14	VDDS
CK16	VDDS	CK18	VSS	CK20	VDDS	CK22	VSS
CK24	VDDS	CK26	VSS	CK28	VDDS	CK30	VSS
CK32	VDDS	CK34	VDDS	CK36	AVDD	CK38	AVDD
CK40	AVSS	CK42	AVSS	CL1	P8_T2_N	CL3	AVSS
CL5	P8_R2_N	CL7	AVDD	CL9	VDDS	CL11	VDDS
CL13	VDDS	CL15	VSS	CL17	VSS	CL19	VDDS
CL21	VDDS	CL23	VSS	CL25	VSS	CL27	VDDS
CL29	VSS	CL31	VDDS	CL33	Reserved-GND	CL35	REF1
CL37	AVSS	CL39	AVDD	CL41	AVDD	CM2	AVSS
CM4	AVSS	CM6	AVSS	CM8	AVSS	CM10	VSS
CM12	VSS	CM14	VSS	CM16	VSS	CM18	VSS

Table 13-1 Pin List Ordered by Location [Continued]

Pin	Name	Pin	Name	Pin	Name	Pin	Name
CM20	VSS	CM22	VSS	CM24	VSS	CM26	VSS
CM28	VSS	CM30	VSS	CM32	VSS	CM34	Reserved-GND
CM36	AVSS	CM38	AVSS	CM40	AVSS	CM42	—

13.5.2 Pin List Ordered by Name

Table 13-2 Pin List Ordered by Name

Name	Pin	Name	Pin	Name	Pin	Name	Pin
ADDR0	B20	ADDR1	C21	ADDR2	D20	ADDR3	E21
ADDR4	F20	ADDR5	G21	ADDR6	H20	ADDR7	J21
ADDR8	K20	ADDR9	L21	ADDR10	M20	ADDR11	N21
ADDR12	P20	ADDR13	R21	ADDR14	C23	ADDR15	D24
ADDR16	E23	ADDR17	F24	ADDR18	G23	ADDR19	H24
ADDR20	J23	ADDR21	K24	ADDR22	C25	ADDR23	E25
AS_N	N23	AVDD	B4	AVDD	B38	AVDD	C39
AVDD	E5	AVDD	F2	AVDD	F4	AVDD	F38
AVDD	G7	AVDD	G39	AVDD	G41	AVDD	H36
AVDD	K4	AVDD	L7	AVDD	L39	AVDD	M2
AVDD	M6	AVDD	M36	AVDD	N41	AVDD	P4
AVDD	P38	AVDD	R39	AVDD	U5	AVDD	V2
AVDD	V4	AVDD	V38	AVDD	W7	AVDD	W39
AVDD	W41	AVDD	Y36	AVDD	AB4	AVDD	AC7
AVDD	AC39	AVDD	AD2	AVDD	AD6	AVDD	AD36
AVDD	AE41	AVDD	AF4	AVDD	AF38	AVDD	AG39
AVDD	AJ5	AVDD	AK2	AVDD	AK4	AVDD	AK38
AVDD	AL7	AVDD	AL39	AVDD	AL41	AVDD	AM36
AVDD	AP4	AVDD	AR7	AVDD	AR39	AVDD	AT2
AVDD	AT6	AVDD	AT36	AVDD	AU41	AVDD	AV4
AVDD	AV38	AVDD	AW39	AVDD	BA5	AVDD	BB2
AVDD	BB4	AVDD	BB38	AVDD	BC7	AVDD	BC39
AVDD	BC41	AVDD	BD36	AVDD	BF4	AVDD	BG7
AVDD	BG39	AVDD	BH2	AVDD	BH6	AVDD	BH36
AVDD	BJ41	AVDD	BK4	AVDD	BK38	AVDD	BL39
AVDD	BN5	AVDD	BP2	AVDD	BP4	AVDD	BP38
AVDD	BR7	AVDD	BR39	AVDD	BR41	AVDD	BT36
AVDD	BV4	AVDD	BW7	AVDD	BW39	AVDD	BY2
AVDD	BY6	AVDD	BY36	AVDD	CA41	AVDD	CB4

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
AVDD	CB38	AVDD	CC39	AVDD	CE5	AVDD	CE41
AVDD	CF2	AVDD	CF4	AVDD	CF36	AVDD	CF38
AVDD	CG7	AVDD	CG39	AVDD	CG41	AVDD	CK4
AVDD	CK36	AVDD	CK38	AVDD	CL7	AVDD	CL39
AVDD	CL41	AVDD25	CH18	AVDD25	CH26	AVDD25	T22
AVDD25	T24	AVSS	A5	AVSS	A11	AVSS	A31
AVSS	A33	AVSS	A35	AVSS	A37	AVSS	A39
AVSS	A41	AVSS	B8	AVSS	B12	AVSS	B42
AVSS	C9	AVSS	C11	AVSS	C31	AVSS	C35
AVSS	D4	AVSS	D8	AVSS	D10	AVSS	D32
AVSS	D34	AVSS	E1	AVSS	E33	AVSS	E35
AVSS	E39	AVSS	F6	AVSS	F42	AVSS	G3
AVSS	G35	AVSS	G37	AVSS	H4	AVSS	H8
AVSS	H40	AVSS	J35	AVSS	J39	AVSS	K8
AVSS	L3	AVSS	L35	AVSS	M4	AVSS	M8
AVSS	M40	AVSS	N1	AVSS	N5	AVSS	N35
AVSS	N37	AVSS	N39	AVSS	P8	AVSS	P42
AVSS	R35	AVSS	T4	AVSS	T8	AVSS	U1
AVSS	U35	AVSS	U39	AVSS	V6	AVSS	V8
AVSS	V42	AVSS	W3	AVSS	W35	AVSS	W37
AVSS	Y4	AVSS	Y8	AVSS	Y40	AVSS	AA35
AVSS	AA39	AVSS	AB8	AVSS	AC3	AVSS	AC35
AVSS	AD4	AVSS	AD8	AVSS	AD40	AVSS	AE1
AVSS	AE5	AVSS	AE35	AVSS	AE37	AVSS	AE39
AVSS	AF8	AVSS	AF42	AVSS	AG35	AVSS	AH4
AVSS	AH8	AVSS	AJ1	AVSS	AJ35	AVSS	AJ39
AVSS	AK6	AVSS	AK8	AVSS	AK42	AVSS	AL3
AVSS	AL35	AVSS	AL37	AVSS	AM4	AVSS	AM8
AVSS	AM40	AVSS	AN35	AVSS	AN39	AVSS	AP8
AVSS	AR3	AVSS	AR35	AVSS	AT4	AVSS	AT8
AVSS	AT40	AVSS	AU1	AVSS	AU5	AVSS	AU35
AVSS	AU37	AVSS	AU39	AVSS	AV8	AVSS	AV42
AVSS	AW35	AVSS	AY4	AVSS	AY8	AVSS	BA1
AVSS	BA35	AVSS	BA39	AVSS	BB6	AVSS	BB8
AVSS	BB42	AVSS	BC3	AVSS	BC35	AVSS	BC37
AVSS	BD4	AVSS	BD8	AVSS	BD40	AVSS	BE35
AVSS	BE39	AVSS	BF8	AVSS	BG3	AVSS	BG35
AVSS	BH4	AVSS	BH8	AVSS	BH40	AVSS	BJ1

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
AVSS	BJ5	AVSS	BJ35	AVSS	BJ37	AVSS	BJ39
AVSS	BK8	AVSS	BK42	AVSS	BL35	AVSS	BM4
AVSS	BM8	AVSS	BN1	AVSS	BN35	AVSS	BN39
AVSS	BP6	AVSS	BP8	AVSS	BP42	AVSS	BR3
AVSS	BR35	AVSS	BR37	AVSS	BT4	AVSS	BT8
AVSS	BT40	AVSS	BU35	AVSS	BU39	AVSS	BV8
AVSS	BW3	AVSS	BW35	AVSS	BY4	AVSS	BY8
AVSS	BY40	AVSS	CA1	AVSS	CA5	AVSS	CA35
AVSS	CA37	AVSS	CA39	AVSS	CB8	AVSS	CB42
AVSS	CC35	AVSS	CD4	AVSS	CD8	AVSS	CE1
AVSS	CE35	AVSS	CE37	AVSS	CE39	AVSS	CF6
AVSS	CF8	AVSS	CF40	AVSS	CF42	AVSS	CG3
AVSS	CG35	AVSS	CG37	AVSS	CH4	AVSS	CH8
AVSS	CH36	AVSS	CH40	AVSS	CJ35	AVSS	CJ39
AVSS	CK40	AVSS	CK42	AVSS	CL3	AVSS	CL37
AVSS	CM2	AVSS	CM4	AVSS	CM6	AVSS	CM8
AVSS	CM36	AVSS	CM38	AVSS	CM40	CLKOUT_A_N	F10
CLKOUT_A_P	E9	CLKOUT_B_N	K10	CLKOUT_B_P	J9	CLKOUT_C_N	V32
CLKOUT_C_P	U33	CS_N	U23	DATA0	E13	DATA1	F12
DATA2	G13	DATA3	H12	DATA4	J13	DATA5	K12
DATA6	L13	DATA7	E15	DATA8	D16	DATA9	G15
DATA10	F16	DATA11	J15	DATA12	H16	DATA13	L15
DATA14	K16	DATA15	N15	DATA16	M16	DATA17	R15
DATA18	P16	DATA19	C17	DATA20	C19	DATA21	E17
DATA22	E19	DATA23	G17	DATA24	G19	DATA25	J17
DATA26	J19	DATA27	L17	DATA28	L19	DATA29	N17
DATA30	N19	DATA31	R17	DTACK_N	L23	EBI_CLK	A15
ETH_REFCLK_N	B10	ETH_REFCLK_P	A9	GPIO0	C27	GPIO1	D28
GPIO2	E27	GPIO3	F28	GPIO4	G27	GPIO5	H28
GPIO6	J27	GPIO7	K28	GPIO8	L27	GPIO9	M28
GPIO10	N27	GPIO11	P28	GPIO12	R27	GPIO13	T28
GPIO14	U27	GPIO15	V28	I2C_SCL	AA29	I2C_SDA	U29
IEEE1588_PULSE0	Y30	IEEE1588_PULSE1	AB30	IEEE1588_REFCLK_N	AA31	IEEE1588_REFCLK_P	Y32
IEEE1588_RESET_N	D30	IEEE1588_SYSTIME0	E31	IEEE1588_SYSTIME1	F30	IEEE1588_SYSTIME2	H30
IEEE1588_SYSTIME3	K30	INT_N	M30	LED_CLK	T30	LED_DATA0	U31
LED_DATA1	V30	LED_DATA2	W31	LED_EN	P30	MDC	W29
MDIO	Y28	P0_R0_N	B6	P0_R0_P	C5	P0_R1_N	C7
P0_R1_P	A7	P0_R2_N	E7	P0_R2_P	D6	P0_R3_N	J5

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
P0_R3_P	G5	P0_T0_N	B2	P0_T0_P	C1	P0_T1_N	C3
P0_T1_P	A3	P0_T2_N	E3	P0_T2_P	D2	P0_T3_N	J1
P0_T3_P	G1	P1_R0_N	J7	P1_R0_P	H6	P1_R1_N	K6
P1_R1_P	L5	P1_R2_N	P6	P1_R2_P	R5	P1_R3_N	R7
P1_R3_P	N7	P1_T0_N	J3	P1_T0_P	H2	P1_T1_N	K2
P1_T1_P	L1	P1_T2_N	P2	P1_T2_P	R1	P1_T3_N	R3
P1_T3_P	N3	P2_R0_N	U7	P2_R0_P	T6	P2_R1_N	AA7
P2_R1_P	Y6	P2_R2_N	AC5	P2_R2_P	AB6	P2_R3_N	AA5
P2_R3_P	W5	P2_T0_N	U3	P2_T0_P	T2	P2_T1_N	AA3
P2_T1_P	Y2	P2_T2_N	AC1	P2_T2_P	AB2	P2_T3_N	AA1
P2_T3_P	W1	P3_R0_N	AF6	P3_R0_P	AG5	P3_R1_N	AG7
P3_R1_P	AE7	P3_R2_N	AJ7	P3_R2_P	AH6	P3_R3_N	AN5
P3_R3_P	AL5	P3_T0_N	AF2	P3_T0_P	AG1	P3_T1_N	AG3
P3_T1_P	AE3	P3_T2_N	AJ3	P3_T2_P	AH2	P3_T3_N	AN1
P3_T3_P	AL1	P4_R0_N	AN7	P4_R0_P	AM6	P4_R1_N	AP6
P4_R1_P	AR5	P4_R2_N	AV6	P4_R2_P	AW5	P4_R3_N	AW7
P4_R3_P	AU7	P4_T0_N	AN3	P4_T0_P	AM2	P4_T1_N	AP2
P4_T1_P	AR1	P4_T2_N	AV2	P4_T2_P	AW1	P4_T3_N	AW3
P4_T3_P	AU3	P5_R0_N	BA7	P5_R0_P	AY6	P5_R1_N	BE7
P5_R1_P	BD6	P5_R2_N	BG5	P5_R2_P	BF6	P5_R3_N	BE5
P5_R3_P	BC5	P5_T0_N	BA3	P5_T0_P	AY2	P5_T1_N	BE3
P5_T1_P	BD2	P5_T2_N	BG1	P5_T2_P	BF2	P5_T3_N	BE1
P5_T3_P	BC1	P6_R0_N	BK6	P6_R0_P	BL5	P6_R1_N	BL7
P6_R1_P	BJ7	P6_R2_N	BN7	P6_R2_P	BM6	P6_R3_N	BU5
P6_R3_P	BR5	P6_T0_N	BK2	P6_T0_P	BL1	P6_T1_N	BL3
P6_T1_P	BJ3	P6_T2_N	BN3	P6_T2_P	BM2	P6_T3_N	BU1
P6_T3_P	BR1	P7_R0_N	BU7	P7_R0_P	BT6	P7_R1_N	BV6
P7_R1_P	BW5	P7_R2_N	CB6	P7_R2_P	CC5	P7_R3_N	CC7
P7_R3_P	CA7	P7_T0_N	BU3	P7_T0_P	BT2	P7_T1_N	BV2
P7_T1_P	BW1	P7_T2_N	CB2	P7_T2_P	CC1	P7_T3_N	CC3
P7_T3_P	CA3	P8_R0_N	CE7	P8_R0_P	CD6	P8_R1_N	CJ7
P8_R1_P	CH6	P8_R2_N	CL5	P8_R2_P	CK6	P8_R3_N	CJ5
P8_R3_P	CG5	P8_T0_N	CE3	P8_T0_P	CD2	P8_T1_N	CJ3
P8_T1_P	CH2	P8_T2_N	CL1	P8_T2_P	CK2	P8_T3_N	CJ1
P8_T3_P	CG1	PCIE_REFCLK_N	C33	PCIE_REFCLK_P	B32	PCIE_RESET_N0	M34
PCIE_RESET_N1	K34	PCIE_RESET_N2	H34	PCIE_RESET_N3	F34	PCIE_RESET_N4	P34
PCIE_RESET_N5	R33	PCIE_RESET_N6	N33	PCIE_RESET_N7	L33	PCIE_RESET_N8	J33
PCIE_XREFCLK0_N	CJ15	PCIE_XREFCLK0_P	CJ13	PCIE_XREFCLK1_N	CJ19	PCIE_XREFCLK1_P	CJ17

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
PCIE_XREFCLK2_N	CJ23	PCIE_XREFCLK2_P	CJ21	PCIE_XREFCLK3_N	CJ27	PCIE_XREFCLK3_P	CJ25
PCIE0_R0_N	C37	PCIE0_R0_P	D38	PCIE0_R1_N	D36	PCIE0_R1_P	B36
PCIE0_R2_N	F36	PCIE0_R2_P	E37	PCIE0_R3_N	K38	PCIE0_R3_P	H38
PCIE0_R4_N	K36	PCIE0_R4_P	J37	PCIE0_R5_N	L37	PCIE0_R5_P	M38
PCIE0_R6_N	R37	PCIE0_R6_P	T38	PCIE0_R7_N	T36	PCIE0_R7_P	P36
PCIE0_T0_N	C41	PCIE0_T0_P	D42	PCIE0_T1_N	D40	PCIE0_T1_P	B40
PCIE0_T2_N	F40	PCIE0_T2_P	E41	PCIE0_T3_N	K42	PCIE0_T3_P	H42
PCIE0_T4_N	K40	PCIE0_T4_P	J41	PCIE0_T5_N	L41	PCIE0_T5_P	M42
PCIE0_T6_N	R41	PCIE0_T6_P	T42	PCIE0_T7_N	T40	PCIE0_T7_P	P40
PCIE1_R0_N	V36	PCIE1_R0_P	U37	PCIE1_R1_N	AB38	PCIE1_R1_P	Y38
PCIE1_R2_N	AB36	PCIE1_R2_P	AA37	PCIE1_R3_N	AC37	PCIE1_R3_P	AD38
PCIE1_R4_N	AG37	PCIE1_R4_P	AH38	PCIE1_R5_N	AH36	PCIE1_R5_P	AF36
PCIE1_R6_N	AK36	PCIE1_R6_P	AJ37	PCIE1_R7_N	AP38	PCIE1_R7_P	AM38
PCIE1_T0_N	V40	PCIE1_T0_P	U41	PCIE1_T1_N	AB42	PCIE1_T1_P	Y42
PCIE1_T2_N	AB40	PCIE1_T2_P	AA41	PCIE1_T3_N	AC41	PCIE1_T3_P	AD42
PCIE1_T4_N	AG41	PCIE1_T4_P	AH42	PCIE1_T5_N	AH40	PCIE1_T5_P	AF40
PCIE1_T6_N	AK40	PCIE1_T6_P	AJ41	PCIE1_T7_N	AP42	PCIE1_T7_P	AM42
PCIE2_R0_N	AP36	PCIE2_R0_P	AN37	PCIE2_R1_N	AR37	PCIE2_R1_P	AT38
PCIE2_R2_N	AW37	PCIE2_R2_P	AY38	PCIE2_R3_N	AY36	PCIE2_R3_P	AV36
PCIE2_R4_N	BB36	PCIE2_R4_P	BA37	PCIE2_R5_N	BF38	PCIE2_R5_P	BD38
PCIE2_R6_N	BF36	PCIE2_R6_P	BE37	PCIE2_R7_N	BG37	PCIE2_R7_P	BH38
PCIE2_T0_N	AP40	PCIE2_T0_P	AN41	PCIE2_T1_N	AR41	PCIE2_T1_P	AT42
PCIE2_T2_N	AW41	PCIE2_T2_P	AY42	PCIE2_T3_N	AY40	PCIE2_T3_P	AV40
PCIE2_T4_N	BB40	PCIE2_T4_P	BA41	PCIE2_T5_N	BF42	PCIE2_T5_P	BD42
PCIE2_T6_N	BF40	PCIE2_T6_P	BE41	PCIE2_T7_N	BG41	PCIE2_T7_P	BH42
PCIE3_R0_N	BL37	PCIE3_R0_P	BM38	PCIE3_R1_N	BM36	PCIE3_R1_P	BK36
PCIE3_R2_N	BP36	PCIE3_R2_P	BN37	PCIE3_R3_N	BV38	PCIE3_R3_P	BT38
PCIE3_R4_N	BV36	PCIE3_R4_P	BU37	PCIE3_R5_N	BW37	PCIE3_R5_P	BY38
PCIE3_R6_N	CC37	PCIE3_R6_P	CD38	PCIE3_R7_N	CD36	PCIE3_R7_P	CB36
PCIE3_T0_N	BL41	PCIE3_T0_P	BM42	PCIE3_T1_N	BM40	PCIE3_T1_P	BK40
PCIE3_T2_N	BP40	PCIE3_T2_P	BN41	PCIE3_T3_N	BV42	PCIE3_T3_P	BT42
PCIE3_T4_N	BV40	PCIE3_T4_P	BU41	PCIE3_T5_N	BW41	PCIE3_T5_P	BY42
PCIE3_T6_N	CC41	PCIE3_T6_P	CD42	PCIE3_T7_N	CD40	PCIE3_T7_P	CB40
PCIE4_R0_N	CJ37	PCIE4_R0_P	CH38	PCIE4_T0_N	CJ41	PCIE4_T0_P	CH42
PLLVDD0	T34	PLLVDD1	L9	PLLVSS0	V34	PLLVSS1	N9
REF0	B34	REF1	CL35	REF2	F8	REF3	CK8
Reserved-GND	A19	Reserved-GND	CL33	Reserved-GND	CM34	Reserved-NC	A23
Reserved-NC	N13	Reserved-NC	CG13	Reserved-NC	CG15	Reserved-NC	CG17

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
Reserved-NC	CG19	Reserved-NC	CG21	Reserved-NC	CG23	Reserved-NC	CG25
Reserved-NC	CG27	Reserved-NC	CG29	Reserved-NC	CJ29	Reserved-PD	E11
Reserved-PUVDD25	C13	Reserved-PUVDD25	D12	Reserved-PUVDD25	D14	Reserved-VDDS	CH22
RESET_N	B30	RW_N	R23	SENSE_VDDF	R25	SENSE_VDDS	N25
SENSE_VSSF	T26	SENSE_VSSS	P26	SPI_BOOT	J31	SPI_CLK	C29
SPI_CS_N	G33	SPI_IO2	G31	SPI_IO3	R31	SPI_MISO	N31
SPI_MOSI	L31	TCK	M12	TDI	R13	TDO	T16
THERMAL_IN	A21	THERMAL_OUT	B22	TMS	P12	TRST_N	U15
VDD25	B26	VDD25	B28	VDD25	D18	VDD25	D22
VDD25	E29	VDD25	F14	VDD25	F26	VDD25	F32
VDD25	H18	VDD25	H22	VDD25	J29	VDD25	K14
VDD25	K26	VDD25	K32	VDD25	L25	VDD25	M18
VDD25	M22	VDD25	M24	VDD25	N29	VDD25	P14
VDD25	P32	VDD25	T18	VDD25	T20	VDDF	AD30
VDDF	AE17	VDDF	AE23	VDDF	AE25	VDDF	AE29
VDDF	AF14	VDDF	AF20	VDDF	AF24	VDDF	AF28
VDDF	AG13	VDDF	AG15	VDDF	AG19	VDDF	AG21
VDDF	AG27	VDDF	AH16	VDDF	AH18	VDDF	AH22
VDDF	AH26	VDDF	AH30	VDDF	AJ17	VDDF	AJ23
VDDF	AJ25	VDDF	AJ29	VDDF	AK14	VDDF	AK20
VDDF	AK24	VDDF	AK28	VDDF	AL13	VDDF	AL15
VDDF	AL19	VDDF	AL27	VDDF	AM16	VDDF	AM18
VDDF	AM26	VDDF	AM30	VDDF	AN17	VDDF	AN25
VDDF	AN29	VDDF	AP14	VDDF	AP28	VDDF	AR13
VDDF	AR15	VDDF	AR27	VDDF	AT16	VDDF	AT18
VDDF	AT26	VDDF	AT30	VDDF	AU17	VDDF	AU25
VDDF	AU29	VDDF	AV14	VDDF	AV28	VDDF	AW13
VDDF	AW15	VDDF	AW27	VDDF	AY16	VDDF	AY18
VDDF	AY26	VDDF	AY30	VDDF	BA17	VDDF	BA25
VDDF	BA29	VDDF	BB14	VDDF	BB24	VDDF	BB28
VDDF	BC13	VDDF	BC15	VDDF	BC19	VDDF	BC21
VDDF	BC27	VDDF	BD16	VDDF	BD18	VDDF	BD22
VDDF	BD26	VDDF	BD30	VDDF	BE17	VDDF	BE23
VDDF	BE25	VDDF	BE29	VDDF	BF14	VDDF	BF20
VDDF	BF24	VDDF	BF28	VDDF	BG13	VDDF	BG15
VDDF	BG19	VDDF	BG21	VDDF	BG27	VDDF	BH16
VDDF	BH18	VDDF	BH22	VDDF	BH26	VDDF	BH30
VDDF	BJ17	VDDF	BJ23	VDDF	BJ25	VDDF	BJ29

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
VDDF	BK14	VDDF	BK20	VDDF	BK24	VDDF	BK28
VDDF	BL13	VDDF	BL15	VDDF	BL19	VDDF	BL21
VDDF	BL27	VDDF	BM16	VDDF	BM18	VDDF	BM22
VDDF	BM26	VDDF	BM30	VDDF	BN17	VDDF	BN23
VDDF	BN25	VDDF	BN29	VDDF	BP14	VDDF	BP20
VDDF	BP24	VDDF	BP28	VDDF	BR13	VDDF	BR15
VDDF	BR19	VDDF	BR21	VDDF	BR27	VDDF	BT16
VDDF	BT18	VDDF	BT22	VDDF	BT26	VDDF	BT30
VDDF	BU17	VDDF	BU23	VDDF	BU25	VDDF	BU29
VDDF	BV14	VDDF	BV20	VDDF	BV24	VDDF	BV26
VDDF	BV28	VDDF	BW13	VDDF	BW15	VDDF	BW17
VDDF	BW19	VDDF	BW21	VDDF	BW23	VDDF	BW27
VDDF	BY16	VDDF	BY18	VDDF	BY20	VDDF	BY22
VDDF	BY24	VDDF	BY26	VDDF	BY30	VDDF	CA17
VDDF	CA23	VDDF	CA25	VDDF	CA29	VDDF	CB14
VDDF	CB20	VDDF	CB24	VDDF	CB28	VDDF	CC13
VDDF	CC15	VDDF	CC19	VDDF	CC21	VDDF	CC27
VDDF	CD16	VDDF	CD18	VDDF	CD22	VDDF	CD26
VDDF	CD30	VDDF	CE17	VDDF	CE23	VDDF	CE25
VDDF	CE29	VDDF	CF14	VDDF	CF20	VDDF	CF24
VDDF	CF28	VDDS	L11	VDDS	P10	VDDS	R9
VDDS	R11	VDDS	T14	VDDS	V10	VDDS	V12
VDDS	V14	VDDS	V16	VDDS	V18	VDDS	V20
VDDS	V22	VDDS	V24	VDDS	V26	VDDS	W9
VDDS	W11	VDDS	Y14	VDDS	Y16	VDDS	Y18
VDDS	Y20	VDDS	Y22	VDDS	Y24	VDDS	Y26
VDDS	Y34	VDDS	AB10	VDDS	AB12	VDDS	AB14
VDDS	AB16	VDDS	AB18	VDDS	AB20	VDDS	AB22
VDDS	AB24	VDDS	AB26	VDDS	AB28	VDDS	AB32
VDDS	AB34	VDDS	AC9	VDDS	AC11	VDDS	AC31
VDDS	AC33	VDDS	AD14	VDDS	AD16	VDDS	AD18
VDDS	AD20	VDDS	AD22	VDDS	AD24	VDDS	AD26
VDDS	AD28	VDDS	AF12	VDDS	AF32	VDDS	AF34
VDDS	AG11	VDDS	AG31	VDDS	AG33	VDDS	AK12
VDDS	AK32	VDDS	AL11	VDDS	AL31	VDDS	AP12
VDDS	AP32	VDDS	AR11	VDDS	AR31	VDDS	AV12
VDDS	AV32	VDDS	AW11	VDDS	AW31	VDDS	BB12
VDDS	BB32	VDDS	BC11	VDDS	BC31	VDDS	BF12

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
VDDS	BF32	VDDS	BG11	VDDS	BG31	VDDS	BG33
VDDS	BK10	VDDS	BK12	VDDS	BK32	VDDS	BK34
VDDS	BL9	VDDS	BL11	VDDS	BL31	VDDS	BL33
VDDS	BM10	VDDS	BM34	VDDS	BN9	VDDS	BN33
VDDS	BP10	VDDS	BP12	VDDS	BP32	VDDS	BP34
VDDS	BR9	VDDS	BR11	VDDS	BR31	VDDS	BR33
VDDS	BV10	VDDS	BV12	VDDS	BV32	VDDS	BV34
VDDS	BW9	VDDS	BW11	VDDS	BW31	VDDS	BW33
VDDS	BY10	VDDS	BY34	VDDS	CB10	VDDS	CB12
VDDS	CB32	VDDS	CB34	VDDS	CC9	VDDS	CC11
VDDS	CC31	VDDS	CC33	VDDS	CF10	VDDS	CF12
VDDS	CF32	VDDS	CF34	VDDS	CG9	VDDS	CG11
VDDS	CG31	VDDS	CG33	VDDS	CH30	VDDS	CJ9
VDDS	CJ11	VDDS	CK14	VDDS	CK16	VDDS	CK20
VDDS	CK24	VDDS	CK28	VDDS	CK32	VDDS	CK34
VDDS	CL9	VDDS	CL11	VDDS	CL13	VDDS	CL19
VDDS	CL21	VDDS	CL27	VDDS	CL31	VSS	A13
VSS	A17	VSS	A25	VSS	A27	VSS	A29
VSS	B14	VSS	B16	VSS	B18	VSS	B24
VSS	C15	VSS	D26	VSS	F18	VSS	F22
VSS	G9	VSS	G11	VSS	G25	VSS	G29
VSS	H10	VSS	H14	VSS	H26	VSS	H32
VSS	J11	VSS	J25	VSS	K18	VSS	K22
VSS	L29	VSS	M10	VSS	M14	VSS	M26
VSS	M32	VSS	N11	VSS	P18	VSS	P22
VSS	P24	VSS	R19	VSS	R29	VSS	T10
VSS	T12	VSS	T32	VSS	U9	VSS	U11
VSS	U13	VSS	U17	VSS	U19	VSS	U21
VSS	U25	VSS	W13	VSS	W15	VSS	W17
VSS	W19	VSS	W21	VSS	W23	VSS	W25
VSS	W27	VSS	W33	VSS	Y10	VSS	Y12
VSS	AA9	VSS	AA11	VSS	AA13	VSS	AA15
VSS	AA17	VSS	AA19	VSS	AA21	VSS	AA23
VSS	AA25	VSS	AA27	VSS	AA33	VSS	AC13
VSS	AC15	VSS	AC17	VSS	AC19	VSS	AC21
VSS	AC23	VSS	AC25	VSS	AC27	VSS	AC29
VSS	AD10	VSS	AD12	VSS	AD32	VSS	AD34
VSS	AE9	VSS	AE11	VSS	AE13	VSS	AE15

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
VSS	AE19	VSS	AE21	VSS	AE27	VSS	AE31
VSS	AE33	VSS	AF16	VSS	AF18	VSS	AF22
VSS	AF26	VSS	AF30	VSS	AG17	VSS	AG23
VSS	AG25	VSS	AG29	VSS	AH12	VSS	AH14
VSS	AH20	VSS	AH24	VSS	AH28	VSS	AH32
VSS	AH34	VSS	AJ11	VSS	AJ13	VSS	AJ15
VSS	AJ19	VSS	AJ21	VSS	AJ27	VSS	AJ31
VSS	AK16	VSS	AK18	VSS	AK22	VSS	AK26
VSS	AK30	VSS	AL17	VSS	AL25	VSS	AL29
VSS	AM12	VSS	AM14	VSS	AM28	VSS	AM32
VSS	AN11	VSS	AN13	VSS	AN15	VSS	AN27
VSS	AN31	VSS	AP16	VSS	AP18	VSS	AP26
VSS	AP30	VSS	AR17	VSS	AR25	VSS	AR29
VSS	AT12	VSS	AT14	VSS	AT28	VSS	AT32
VSS	AU11	VSS	AU13	VSS	AU15	VSS	AU27
VSS	AU31	VSS	AV16	VSS	AV18	VSS	AV26
VSS	AV30	VSS	AW17	VSS	AW25	VSS	AW29
VSS	AY12	VSS	AY14	VSS	AY28	VSS	AY32
VSS	BA11	VSS	BA13	VSS	BA15	VSS	BA27
VSS	BA31	VSS	BB16	VSS	BB18	VSS	BB26
VSS	BB30	VSS	BC17	VSS	BC23	VSS	BC25
VSS	BC29	VSS	BD12	VSS	BD14	VSS	BD20
VSS	BD24	VSS	BD28	VSS	BD32	VSS	BE11
VSS	BE13	VSS	BE15	VSS	BE19	VSS	BE21
VSS	BE27	VSS	BE31	VSS	BF16	VSS	BF18
VSS	BF22	VSS	BF26	VSS	BF30	VSS	BG17
VSS	BG23	VSS	BG25	VSS	BG29	VSS	BH12
VSS	BH14	VSS	BH20	VSS	BH24	VSS	BH28
VSS	BH32	VSS	BH34	VSS	BJ9	VSS	BJ11
VSS	BJ13	VSS	BJ15	VSS	BJ19	VSS	BJ21
VSS	BJ27	VSS	BJ31	VSS	BJ33	VSS	BK16
VSS	BK18	VSS	BK22	VSS	BK26	VSS	BK30
VSS	BL17	VSS	BL23	VSS	BL25	VSS	BL29
VSS	BM12	VSS	BM14	VSS	BM20	VSS	BM24
VSS	BM28	VSS	BM32	VSS	BN11	VSS	BN13
VSS	BN15	VSS	BN19	VSS	BN21	VSS	BN27
VSS	BN31	VSS	BP16	VSS	BP18	VSS	BP22
VSS	BP26	VSS	BP30	VSS	BR17	VSS	BR23

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
VSS	BR25	VSS	BR29	VSS	BT10	VSS	BT12
VSS	BT14	VSS	BT20	VSS	BT24	VSS	BT28
VSS	BT32	VSS	BT34	VSS	BU9	VSS	BU11
VSS	BU13	VSS	BU15	VSS	BU19	VSS	BU21
VSS	BU27	VSS	BU31	VSS	BU33	VSS	BV16
VSS	BV18	VSS	BV22	VSS	BV30	VSS	BW25
VSS	BW29	VSS	BY12	VSS	BY14	VSS	BY28
VSS	BY32	VSS	CA9	VSS	CA11	VSS	CA13
VSS	CA15	VSS	CA19	VSS	CA21	VSS	CA27
VSS	CA31	VSS	CA33	VSS	CB16	VSS	CB18
VSS	CB22	VSS	CB26	VSS	CB30	VSS	CC17
VSS	CC23	VSS	CC25	VSS	CC29	VSS	CD10
VSS	CD12	VSS	CD14	VSS	CD20	VSS	CD24
VSS	CD28	VSS	CD32	VSS	CD34	VSS	CE9
VSS	CE11	VSS	CE13	VSS	CE15	VSS	CE19
VSS	CE21	VSS	CE27	VSS	CE31	VSS	CE33
VSS	CF16	VSS	CF18	VSS	CF22	VSS	CF26
VSS	CF30	VSS	CH10	VSS	CH12	VSS	CH14
VSS	CH16	VSS	CH20	VSS	CH24	VSS	CH28
VSS	CH32	VSS	CH34	VSS	CJ31	VSS	CJ33
VSS	CK10	VSS	CK12	VSS	CK18	VSS	CK22
VSS	CK26	VSS	CK30	VSS	CL15	VSS	CL17
VSS	CL23	VSS	CL25	VSS	CL29	VSS	CM10
VSS	CM12	VSS	CM14	VSS	CM16	VSS	CM18
VSS	CM20	VSS	CM22	VSS	CM24	VSS	CM26
VSS	CM28	VSS	CM30	VSS	CM32	—	A1
—	AF10	—	AG9	—	AH10	—	AJ9
—	AJ33	—	AK10	—	AK34	—	AL9
—	AL21	—	AL23	—	AL33	—	AM10
—	AM20	—	AM22	—	AM24	—	AM34
—	AN9	—	AN19	—	AN21	—	AN23
—	AN33	—	AP10	—	AP20	—	AP22
—	AP24	—	AP34	—	AR9	—	AR19
—	AR21	—	AR23	—	AR33	—	AT10
—	AT20	—	AT22	—	AT24	—	AT34
—	AU9	—	AU19	—	AU21	—	AU23
—	AU33	—	AV10	—	AV20	—	AV22
—	AV24	—	AV34	—	AW9	—	AW19

Table 13-2 Pin List Ordered by Name [Continued]

Name	Pin	Name	Pin	Name	Pin	Name	Pin
—	AW21	—	AW23	—	AW33	—	AY10
—	AY20	—	AY22	—	AY24	—	AY34
—	BA9	—	BA19	—	BA21	—	BA23
—	BA33	—	BB10	—	BB20	—	BB22
—	BB34	—	BC9	—	BC33	—	BD10
—	BD34	—	BE9	—	BE33	—	BF10
—	BF34	—	BG9	—	BH10	—	CM42

13.5.3 Disabled Pins on Reduced-Functionality SKUs

Certain pins are disabled on the FM10420 and FM10064 devices based on the internal fusing. The electrical differences between the SKUs is discussed in Section 1.0, “Introduction” and Section 2.0, “Architecture Overview”.

The logical outcome of these differences is that certain pins are disabled as the relevant internal circuitry is disabled. Disabled pins may be allowed to float.

Disabled pins are listed in Table 13-3. Accessible pins are labeled with “X”, while unavailable pins are marked with “N/A”.

Note: Only potentially affected pins are listed in Table 13-3.

Table 13-3 Disabled Pins List

Pin	Name	SKU	
		FM10840	FM10420
M34	PCIE_RESET_N0	X	X
K34	PCIE_RESET_N1	X	X
H34	PCIE_RESET_N2	X	X
F34	PCIE_RESET_N3	X	X
P34	PCIE_RESET_N4	X	X
R33	PCIE_RESET_N5	X	N/A
N33	PCIE_RESET_N6	X	X
L33	PCIE_RESET_N7	X	N/A
J33	PCIE_RESET_N8	X	X
CJ15	PCIE_XREFCLK0_N	X	X
CJ13	PCIE_XREFCLK0_P	X	X
CJ19	PCIE_XREFCLK1_N	X	X
CJ17	PCIE_XREFCLK1_P	X	X
Ch23	PCIE_XREFCLK2_N	X	X
CJ21	PCIE_XREFCLK2_P	X	X
K36	PCIE0_R4_N	X	X

Table 13-3 Disabled Pins List [Continued]

Pin	Name	SKU	
		FM10840	FM10420
J37	PCIE0_R4_P	X	X
L37	PCIE0_R5_N	X	X
M38	PCIE0_R5_P	X	X
R37	PCIE0_R6_N	X	X
T38	PCIE0_R6_P	X	X
T36	PCIE0_R7_N	X	X
P36	PCIE0_R7_P	X	X
K40	PCIE0_T4_N	X	X
J41	PCIE0_T4_P	X	X
L41	PCIE0_T5_N	X	X
M42	PCIE0_T5_P	X	X
R41	PCIE0_T6_N	X	X
T42	PCIE0_T6_P	X	X
T40	PCIE0_T7_N	X	X
P40	PCIE0_T7_P	X	X
V36	PCIE1_R0_N	X	X
U37	PCIE1_R0_P	X	X
AB38	PCIE1_R1_N	X	X
Y38	PCIE1_R1_P	X	X
AB36	PCIE1_R2_N	X	X
AA37	PCIE1_R2_P	X	X
AC37	PCIE1_R3_N	X	X
AD38	PCIE1_R3_P	X	X
AG37	PCIE1_R4_N	X	X
AH38	PCIE1_R4_P	X	X
AH36	PCIE1_R5_N	X	X
AF36	PCIE1_R5_P	X	X
AK36	PCIE1_R6_N	X	X
AJ37	PCIE1_R6_P	X	X
AP38	PCIE1_R7_N	X	X
AM38	PCIE1_R7_P	X	X
V40	PCIE1_T0_N	X	X
U41	PCIE1_T0_P	X	X
AB42	PCIE1_T1_N	X	X
Y42	PCIE1_T1_P	X	X
AB40	PCIE1_T2_N	X	X

Table 13-3 Disabled Pins List [Continued]

Pin	Name	SKU	
		FM10840	FM10420
AA41	PCIE1_T2_P	X	X
AC41	PCIE1_T3_N	X	X
AD42	PCIE1_T3_P	X	X
AG41	PCIE1_T4_N	X	X
AH42	PCIE1_T4_P	X	X
AH40	PCIE1_T5_N	X	X
AF40	PCIE1_T5_P	X	X
AK40	PCIE1_T6_N	X	X
AJ41	PCIE1_T6_P	X	X
AP42	PCIE1_T7_N	X	X
AM42	PCIE1_T7_P	X	X
AP36	PCIE2_R0_N	X	X
AN37	PCIE2_R0_P	X	X
AR37	PCIE2_R1_N	X	X
AT38	PCIE2_R1_P	X	X
AW37	PCIE2_R2_N	X	X
AY38	PCIE2_R2_P	X	X
AY36	PCIE2_R3_N	X	X
AV36	PCIE2_R3_P	X	X
BB36	PCIE2_R4_N	X	N/A
BA37	PCIE2_R4_P	X	N/A
BF38	PCIE2_R5_N	X	N/A
BD38	PCIE2_R5_P	X	N/A
BF36	PCIE2_R6_N	X	N/A
BE37	PCIE2_R6_P	X	N/A
BG37	PCIE2_R7_N	X	N/A
BH38	PCIE2_R7_P	X	N/A
AP40	PCIE2_T0_N	X	X
AN41	PCIE2_T0_P	X	X
AR41	PCIE2_T1_N	X	X
AT42	PCIE2_T1_P	X	X
AW41	PCIE2_T2_N	X	X
AY42	PCIE2_T2_P	X	X
AY40	PCIE2_T3_N	X	X
AV40	PCIE2_T3_P	X	X
BB40	PCIE2_T4_N	X	N/A

Table 13-3 Disabled Pins List [Continued]

Pin	Name	SKU	
		FM10840	FM10420
BA41	PCIE2_T4_P	X	N/A
BF42	PCIE2_T5_N	X	N/A
BD42	PCIE2_T5_P	X	N/A
BF40	PCIE2_T6_N	X	N/A
BE41	PCIE2_T6_P	X	N/A
BG41	PCIE2_T7_N	X	N/A
BH42	PCIE2_T7_P	X	N/A
BL37	PCIE3_R0_N	X	X
BM38	PCIE3_R0_P	X	X
BM36	PCIE3_R1_N	X	X
BK36	PCIE3_R1_P	X	X
BP36	PCIE3_R2_N	X	X
BN37	PCIE3_R2_P	X	X
BV38	PCIE3_R3_N	X	X
BT38	PCIE3_R3_P	X	X
BV36	PCIE3_R4_N	X	N/A
BU37	PCIE3_R4_P	X	N/A
BW37	PCIE3_R5_N	X	N/A
BY38	PCIE3_R5_P	X	N/A
CC37	PCIE3_R6_N	X	N/A
CD38	PCIE3_R6_P	X	N/A
CD36	PCIE3_R7_N	X	N/A
CB36	PCIE3_R7_P	X	N/A
BL41	PCIE3_T0_N	X	X
BM42	PCIE3_T0_P	X	X
BM40	PCIE3_T1_N	X	X
BK40	PCIE3_T1_P	X	X
BP40	PCIE3_T2_N	X	X
BN41	PCIE3_T2_P	X	X
BV42	PCIE3_T3_N	X	X
BT42	PCIE3_T3_P	X	X
BV40	PCIE3_T4_N	X	N/A
BU41	PCIE3_T4_P	X	N/A
BW41	PCIE3_T5_N	X	N/A
BY42	PCIE3_T5_P	X	N/A
CC41	PCIE3_T6_N	X	N/A

Table 13-3 Disabled Pins List [Continued]

Pin	Name	SKU	
		FM10840	FM10420
CD42	PCIE3_T6_P	X	N/A
CD40	PCIE3_T7_N	X	N/A
CB40	PCIE3_T7_P	X	N/A

13.5.4 Pin Mapping to PEP Ports per SKU

SKU	Available PEP Port#	PCIe Mode	Pin/ Port#	PCIE0_T/ R[0..7]	PCIE1_T/ R[0..7]	PCIE2_T/ R[0..7]	PCIE3_T/ R[0..7]	PCIE4_T/ R[0]
FM10840	PEP [0..8]	PCIe x8 Mode	Pin	PCIE0_T/ R[0..7]	PCIE1_T/ R[0..7]	PCIE2_T/ R[0..7]	PCIE3_T/ R[0..7]	PCIE4_T/R[0]
			Port#	PEP[0]	PEP[2]	PEP[4]	PEP[6]	PEP[8][1]
		PCIe x4 Mode	Pin	PCIE0_T/ R[0..7]	PCIE1_T/ R[0..7]	PCIE2_T/ R[0..7]	PCIE3_T/ R[0..7]	PCIE4_T/R[0]
			Port#	PEP[0,1]	PEP[2,3]	PEP[4,5]	PEP[6,7]	PEP[8][1]
FM10420	PEP [0,2]	PCIe x8 Mode	Pin	PCIE0_T/ R[0..7]	PCIE1_T/ R[0..7]	Not Available	Not Available	Not Available
			Port#	PEP[0]	PEP[2]	N/A	N/A	N/A
	PEP [0,1,2,3]	PCIe x4 Continuous Mode	Pin	PCIE0_T/ R[0..3],[4..7]	PCIE1_T/ R[0..3],[4..7]	Not Available	Not Available	Not Available
			Port#	PEP[0,1]	PEP[2,3]	N/A	N/A	N/A
	PEP [0,2,4,6]	PCIe x4 Staggered Mode	Pin	PCIE0_T/ R[0..3]	PCIE1_T/ R[0..3]	PCIE2_T/ R[0..3]	PCIE3_T/ R[0..3]	Not Available
			Port#	PEP[0]	PEP[2]	PEP[4]	PEP[6]	N/A

[1] PCIe x1 mode only for CPP switch management.

NOTE: This page intentionally left blank.

LEGAL

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors which may cause deviations from published specifications.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.